МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ИНСТИТУТ СТРАТЕГИИ РАЗВИТИЯ ОБРАЗОВАНИЯ
федеральное государственное бюджетное научное учреждение

ИНФОРМАТИКА
(углубленный уровень)

Реализация требований ФГОС среднего общего образования

Методическое пособие для учителя

Москва
2023

УДК 372.8
ББК 74.263.2
И74

Автор:
Н.Н. Самылкина, доктор педагогических наук, доцент, профессор кафедры теории и методики обучения математике и информатике Института математики и информатики МПГУ

Рецензенты:
Л.Л. Босова, член-корреспондент РАО, доктор педагогических наук
И.В. Левченко, доктор педагогических наук, профессор

И74 **Информатика (углубленный уровень). Реализация требований ФГОС среднего общего образования:** методическое пособие для учителя / [Н.Н. Самылкина]. – М. : ФГБНУ «Институт стратегии развития образования», 2023. – 226 с. : ил.
ISBN 978-5-6049296-3-6

Методические рекомендации по учебному предмету «Информатика» (углубленный уровень) включают общую характеристику требований обновленного ФГОС СОО и федеральной рабочей программы (ФРП) по информатике, обзор активных методов обучения и образовательных технологий, актуальных для их реализации. В пособии представлены элементы нового содержания обучения, отсутствующие в существующих учебниках, с примерами заданий и практических работ, а также даны рекомендации по использованию резервного времени по информатике на углубленном уровне.

Материалы предназначены учителям информатики и методистам по информатике, преподавателям методических дисциплин педагогических вузов и колледжей.

Методическое пособие разработано в рамках государственного задания ФГБНУ «Институт стратегии развития образования» на 2023 год «Обновление содержания общего образования».

УДК 372.8
ББК 74.263.2

ISBN 978-5-6049296-3-6

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Приказ Министерства просвещения Российской Федерации от 12 августа 2022 г. № 732 «О внесении изменений в федеральный государственный образовательный стандарт среднего общего образования, утвержденный приказом Министерства образования и науки Российской Федерации от 17 мая 2012 г. № 413» преимущественно вносит изменения в главу II «Требования к результатам освоения основной образовательной программы» во все три группы результатов: личностные, метапредметные и предметные. Предлагаемые изменения личностных результатов обусловлены потребностью более детального описания формируемых в единстве обучения и воспитания системы ценностных ориентаций, соответствующих традиционным ценностям российского общества.

Личностные результаты разбиты на следующие блоки: гражданское, патриотическое, духовно-нравственное, эстетическое, физическое, трудовое, экологическое воспитание и ценности научного познания.

Метапредметные результаты развернуто описаны в трех блоках: универсальные познавательные действия, универсальные коммуникативные действия и универсальные регулятивные действия. Универсальные познавательные действия подразделяются на базовые, логические, исследовательские и работу с информацией. Универсальные коммуникативные действия описаны для сферы общения и совместной деятельности. Универсальные регулятивные действия раскрываются через приемы самоорганизации, самоконтроля и самооценки.

Предметные результаты изучения информатики на уровне среднего общего образования ориентированы на получение компетентностей для последующей профессиональной деятельности как в рамках данной предметной области, так и в смежных с ней областях.

Основная цель изучения учебного предмета «Информатика» на углубленном уровне среднего общего образования – обеспечение дальнейшего развития информационных компетенций выпускника, его готовности к жизни в условиях развивающегося информационного общества и возрастающей конкуренции на рынке труда.

В рамках углубленного уровня изучения информатики обеспечивается целенаправленная подготовка выпускников средней школы к продолжению образования в высших учебных заведениях по специальностям, непосредственно связанным с цифровыми технологиями, таким как программная инженерия; информационная безопасность; информационные системы и технологии; мобильные системы и сети; большие данные и машинное обучение; промышленный интернет вещей; искусственный интеллект; технологии беспроводной связи; робототехника; квантовые технологии; системы распределенного реестра; технологии виртуальной и дополненной реальностей.

Углубленный уровень изучения информатики рекомендуется для технологического профиля, ориентированного на инженерную и информационную сферы деятельности. Углубленный уровень изучения информатики обеспечивает подготовку обучающихся, ориентированных на специальности в области информационных технологий и инженерные специальности; участие в проектной и исследовательской деятельности, связанной с современными направлениями отрасли информационных технологий; подготовку к участию в олимпиадах и сдаче ЕГЭ по информатике.

Всего выделяется 280 часов на информатику углубленного уровня за два года обучения. По 4 часа в неделю по расписанию. Сюда не входят проектная и исследовательская деятельность, а также внеурочная деятельность по информатике. Следовательно, возможно увеличение часов на информатику за счет часов части основной образовательной программы, формируемой участниками образовательных отношений и для реализации междисциплинарных проектов, предусмотренных выбранным профилем обучения. Вместе с тем, 24 сентября 2022 г. в Федеральный закон «Об образовании в Российской Федерации» в статью 12 часть 6.2 внесены изменения, согласно которым за образовательной организацией предусматривается «право перераспределения времени на изучение учебных предметов, по которым не проводится государственная итоговая аттестация, в пользу изучения иных учебных предметов, в том числе на организацию углубленного изучения отдельных учебных предметов и профильное обучение» [12].

Образовательным организациям рекомендован переход на обновленный ФГОС СОО с 1 сентября 2023 г. Сложность такого перехода в том, что учителям

и обучающимся необходимо формировать предметные компетенции в соответствии с новыми тематическими разделами курса обновленного ФГОС СОО, используя при этом существующее учебно-методическое обеспечение курса информатики.

Цель настоящего методического пособия – помочь учителю в организации учебного процесса по информатике в 10 и далее в 11 классах в условиях введения обновленных ФГОС СОО. В методическом пособии даны рекомендации по распределению резервного времени по информатике на углубленном уровне; представлены элементы нового содержания обучения с примерами заданий и практических работ, отсутствующие в существующих учебниках; предложены варианты организации образовательного процесса в условиях цифровой информационно-образовательной среды.

ОБЩАЯ ХАРАКТЕРИСТИКА ТРЕБОВАНИЙ ФГОС СОО И ФРП ПО УЧЕБНОМУ ПРЕДМЕТУ «ИНФОРМАТИКА» НА УРОВНЕ СРЕДНЕГО ОБЩЕГО ОБРАЗОВАНИЯ

Содержание курса информатики на двух уровнях общего образования сгруппировано в четыре тематических раздела: **«Цифровая грамотность»**, **«Теоретические основы информатики»**, **«Алгоритмы и программирование»** и **«Информационные технологии»**.

Рассмотрим изменения предметных результатов освоения углубленного курса информатики на уровне среднего общего образования в их группировке по тематическим разделам. В таблице 1 выделены базовый и углубленный уровни изучения информатики. При этом углубленный уровень включает в себя предметные результаты базового уровня и дополнен результатами для углубленного уровня; предметные результаты углубленного уровня выделены полужирным шрифтом для быстрого визуального отличия от результатов базового уровня.

Таблица 1

Тематический раздел «Цифровая грамотность» (базовый уровень)	**Тематический раздел «Цифровая грамотность» (углубленный уровень)**
10 класс (6 ч); 11 класс (8 ч)	*10 класс (24 ч); 11 класс (0 ч, можно задействовать резервное время)*
– Понимание основных принципов устройства и функционирования современных стационарных и мобильных компьютеров; тенденций развития компьютерных технологий; владение навыками работы с операционными системами, основными видами программного обеспечения для решения учебных задач по выбранной специализации	– Понимание основных принципов устройства и функционирования современных стационарных и мобильных компьютеров; тенденций развития компьютерных технологий; владение навыками работы с операционными системами, основными видами программного обеспечения для решения учебных задач по выбранной специализации

– Наличие представлений о компьютерных сетях и их роли в современном мире; об общих принципах разработки и функционирования интернет-приложений	– Наличие представлений о компьютерных сетях и их роли в современном мире; ***о базовых принципах организации и функционирования компьютерных сетей***; об общих принципах разработки и функционирования интернет-приложений
– Понимание угроз информационной безопасности, использование методов и средств противодействия этим угрозам, соблюдение мер безопасности, предотвращающих незаконное распространения персональных данных; соблюдение требований техники безопасности и гигиены при работе с компьютерами и другими компонентами цифрового окружения; понимание правовых основ использования компьютерных программ, баз данных и материалов, размещенных в сети Интернет	– Понимание угроз информационной безопасности, использование методов и средств противодействия этим угрозам, соблюдение мер безопасности, предотвращающих незаконное распространение персональных данных; соблюдение требований техники безопасности и гигиены при работе с компьютерами и другими компонентами цифрового окружения; понимание правовых основ использования компьютерных программ, баз данных и работы в сети Интернет
– Умение организовывать личное информационное пространство с использованием различных цифровых технологий; понимание возможностей цифровых сервисов государственных услуг, цифровых образовательных сервисов; понимание возможностей и ограничений технологий искусственного интеллекта	– Умение организовывать личное информационное пространство с использованием различных средств цифровых технологий; понимание возможностей цифровых сервисов государственных услуг, цифровых образовательных сервисов; ***понимание основных принципов работы,*** возможностей и ограничения применения технологий искусственного интеллекта

в различных областях; наличие представлений об использовании информационных технологий в различных профессиональных сферах	в различных областях, ***наличие представлений о круге решаемых задач машинного обучения (распознавания, классификации и прогнозирования)***; наличие представлений об использовании информационных технологий в различных профессиональных сферах
Тематический раздел «Теоретические основы информатики» (базовый уровень)	**Тематический раздел «Теоретические основы информатики» (углубленный уровень)**
10 класс (20 ч); *11 класс (4 ч)*	*10 класс (40 ч);* *11 класс (18 ч)*
– Владение представлениями о роли информации и связанных с ней процессов в природе, технике и обществе; понятиями «информация», «информационный процесс», «система», «компоненты системы», «системный эффект», «информационная система», «система управления»; владение методами поиска информации в сети Интернет; умение критически оценивать информацию, полученную из сети Интернет; умение характеризовать большие данные, приводить примеры источников их получения и направления использования	– Владение представлениями о роли информации и связанных с ней процессов в природе, технике и обществе; понятиями «информация», «информационный процесс», «система», «компоненты системы», «системный эффект», «информационная система», «система управления»; владение методами поиска информации в сети Интернет; умение критически оценивать информацию, полученную из сети Интернет; умение характеризовать большие данные, приводить примеры источников их получения и направления использования; – ***умение классифицировать основные задачи анализа данных (прогнозирование, классификация, кластеризация, анализ отклонений); понимать последовательность решения задач анализа данных: сбор первичных данных, очистка и оценка***

	качества данных, выбор и/или построение модели, преобразование данных, визуализация данных, интерпретация результатов
– Понимание основных принципов дискретизации различных видов информации; умение определять информационный объем текстовых, графических и звуковых данных при заданных параметрах дискретизации	– Понимание основных принципов дискретизации различных видов информации; умение определять информационный объем текстовых, графических и звуковых данных при заданных параметрах дискретизации; – ***умение определять среднюю скорость передачи данных, оценивать изменение времени передачи при изменении информационного объема данных и характеристик канала связи***
– Умение строить неравномерные коды, допускающие однозначное декодирование сообщений (префиксные коды)	– Умение строить неравномерные коды, допускающие однозначное декодирование сообщений (префиксные коды); – ***использовать простейшие коды, которые позволяют обнаруживать и исправлять ошибки при передаче данных; строить код, обеспечивающий наименьшую возможную среднюю длину сообщения при известной частоте символов; пояснять принципы работы простых алгоритмов сжатия данных***
– Владение теоретическим аппаратом, позволяющим осуществлять представление заданного натурального числа в различных системах счисления;	– Умение ***использовать при решении задач свойства позиционной записи чисел, алгоритма построения записи числа в позиционной системе счисления с заданным основанием и***

выполнять преобразования логических выражений, используя законы алгебры логики; определять кратчайший путь во взвешенном графе и количество путей между вершинами ориентированного ациклического графа	***построения числа по строке, содержащей запись этого числа в позиционной системе счисления с заданным основанием;*** – умение выполнять арифметические операции в позиционных системах счисления; умение выполнять преобразования логических выражений, используя законы алгебры логики; – ***умение строить логическое выражение в дизъюнктивной и конъюнктивной нормальных формах по заданной таблице истинности; исследовать область истинности высказывания, содержащего переменные; решать несложные логические уравнения и системы уравнений;*** – умение решать алгоритмические задачи, связанные с анализом графов (задачи построения оптимального пути между вершинами графа, определения количества различных путей между вершинами ориентированного ациклического графа); – ***умение использовать деревья при анализе и построении кодов и для представления арифметических выражений, при решении задач поиска и сортировки; умение строить дерево игры по заданному алгоритму; разрабатывать и обосновывать выигрышную стратегию игры***

– Умение использовать компьютерно-математические модели для анализа объектов и процессов: формулировать цель моделирования, выполнять анализ результатов, полученных в ходе моделирования; оценивать адекватность модели моделируемому объекту или процессу; представлять результаты моделирования в наглядном виде	– Умение использовать компьютерно-математические модели для анализа объектов и процессов: формулировать цель моделирования, выполнять анализ результатов, полученных в ходе моделирования; оценивать адекватность модели моделируемому объекту или процессу; представлять результаты моделирования в наглядном виде
Тематический раздел «Алгоритмы и программирование» (базовый уровень)	**Тематический раздел «Алгоритмы и программирование» (углубленный уровень)**
10 класс (20 ч); *11 класс (10 ч)*	*10 класс (44 ч);* *11 класс (50 ч)*
– Умение читать и понимать программы, реализующие несложные алгоритмы обработки числовых и текстовых данных (в том числе массивов и символьных строк) на выбранном для изучения универсальном языке программирования высокого уровня (Паскаль, Python, Java, C++, C#); анализировать алгоритмы с использованием таблиц трассировки; определять без использования компьютера результаты выполнения несложных программ, включающих циклы, ветвления и подпрограммы, при заданных	– ***Понимание базовых алгоритмов обработки числовой и текстовой информации (запись чисел в позиционной системе счисления; нахождение всех простых чисел в заданном диапазоне; обработка многоразрядных целых чисел; анализ символьных строк и др.), алгоритмов поиска и сортировки; умение определять сложность изучаемых в курсе базовых алгоритмов (суммирование элементов массива, сортировка массива, переборные алгоритмы, двоичный поиск) и приводить примеры нескольких алгоритмов разной сложности для решения одной задачи***

исходных данных; модифицировать готовые программы для решения новых задач, использовать их в своих программах в качестве подпрограмм (процедур, функций)	
– Умение реализовывать на выбранном для изучения языке программирования высокого уровня (Паскаль, Python, Java, C++, C#) типовые алгоритмы обработки чисел, числовых последовательностей и массивов: представление числа в виде набора простых сомножителей; нахождение максимальной (минимальной) цифры натурального числа, записанного в системе счисления с основанием, не превышающим 10; вычисление обобщенных характеристик элементов массива или числовой последовательности (суммы, произведения, среднего арифметического, минимального и максимального элементов; количества элементов, удовлетворяющих заданному условию); сортировку элементов массива	– ***Владение универсальным языком программирования высокого уровня (Python, Java, C++, C#), представлениями о базовых типах данных и структурах данных; умение использовать основные управляющие конструкции; умение осуществлять анализ предложенной программы: определять результаты работы программы при заданных исходных данных; определять, при каких исходных данных возможно получение указанных результатов; выявлять данные, которые могут привести к ошибке в работе программы; формулировать предложения по улучшению программного кода***
	– ***Умение разрабатывать и реализовывать в виде программ базовые алгоритмы; умение использовать***

	в программах данные различных типов с учетом ограничений на диапазон их возможных значений, применять при решении задач структуры данных (списки, словари, стеки, очереди, деревья), использовать базовые операции со структурами данных; применять стандартные и собственные подпрограммы для обработки числовых данных и символьных строк; использовать при разработке программ библиотеки подпрограмм; знать функциональные возможности инструментальных средств среды разработки; умение использовать средства отладки программ в среде программирования; умение документировать программы
Тематический раздел «Информационные технологии» (базовый уровень)	**Тематический раздел «Информационные технологии» (углубленный уровень)**
10 класс (6 ч); *11 класс (10 ч)*	*10 класс (14 ч);* *11 класс (48 ч)*
– Умение создавать структурированные текстовые документы и демонстрационные материалы с использованием возможностей современных программных средств и облачных сервисов; умение использовать табличные (реляционные) базы данных, в частности, составлять запросы к базам данных (в том числе запросы с вычисляемыми полями),	– Умение создавать структурированные текстовые документы и демонстрационные материалы с использованием возможностей современных программных средств и облачных сервисов; умение создавать web-страницы; умение использовать электронные таблицы для анализа, представления и обработки данных (включая вычисление суммы, среднего арифметического, наибольшего и наименьшего значений, решение

выполнять сортировку и поиск записей в базе данных; наполнять разработанную базу данных; умение использовать электронные таблицы для анализа, представления и обработки данных (включая вычисление суммы, среднего арифметического, наибольшего и наименьшего значений, решение уравнений)	уравнений, ***выбор оптимального решения, подбор линии тренда, решение задач прогнозирования); владение основными сведениями о базах данных, их структуре, средствах создания и работы с ними; умение использовать табличные (реляционные) базы данных (составлять запросы в базах данных, выполнять сортировку и поиск записей в базе данных, наполнять разработанную базу данных) и справочные системы***
– Умение организовывать личное информационное пространство с использованием различных цифровых технологий; понимание возможностей цифровых сервисов государственных услуг, цифровых образовательных сервисов; понимание возможностей и ограничений технологий искусственного интеллекта в различных областях; наличие представлений об использовании информационных технологий в различных профессиональных сферах	– Умение организовывать личное информационное пространство с использованием различных средств цифровых технологий; понимание возможностей цифровых сервисов государственных услуг, цифровых образовательных сервисов; понимание основных принципов работы, возможностей и ограничения применения технологий искусственного интеллекта в различных областях, ***наличие представлений о круге решаемых задач машинного обучения (распознавания, классификации и прогнозирования)***; наличие представлений об использовании информационных технологий в различных профессиональных сферах
Резервное время – 3 ч	*Резервное время:* в 10 классе – 18 ч; в 11 классе – 24 ч

Предметные результаты по тематическому разделу **«Цифровая грамотность»** на двух уровнях изучения базовом и углубленном различаются очень мало. На углубленном уровне надо хорошо знать материал о работе компьютерных сетей и задачах, решаемых с использованием искусственного интеллекта. В целом, это обязательные цифровые компетенции для каждого выпускника школы в условиях построения цифровой экономики в Российской Федерации. По предлагаемым рабочим программам по информатике базового и углубленного уровней изучения содержания раздела разнесено по разным классам и скорректировано по объему отведенного времени. Фактически на уровне среднего общего образования систематизируются и немного наращиваются цифровые навыки, сформированные в основной школе и внешкольной жизни. Разница по объему учебного времени между базовым и углубленным уровнем всего 10 часов, которая будет использована на изучение нового материала по основам шифрования в контексте темы информационной безопасности при работе в компьютерной сети.

Предметные результаты по тематическому разделу **«Теоретические основы информатики»** на углубленном уровне подразумевают выход на актуальную тему *«Искусственный интеллект и анализ больших данных»*, в которой делается акцент на аналитические и практические умения решать задачи по этой тематике. Далее в этом разделе темы *«Дискретизация различных видов информации»* и *«Компьютерные сети»* представлены интегративно с акцентом на формирование вычислительных навыков. Тема *«Кодирование данных»* помимо устоявшегося в старшей школе кода Хаффмана на углубленном уровне расширена рассмотрением кода Хемминга и переходом к различным алгоритмам сжатия. Но знание этих алгоритмов и их применение пока не проверяются в едином государственном экзамене. Традиционные умения решать задачи с числами в позиционных системах счисления, на использование логических операций и построение простых графов усложнены аспектом применения этих умений для более сложных вычислительных задач.

При этом, остаются равноценными на двух уровнях изучения (базовом и углубленном) предметные результаты в части информационного моделирования.

Предметные результаты по тематическому разделу **«Алгоритмы и программирование»** не просто наращиваются на углубленном уровне, они

отличаются подходом к изучению языка программирования. На базовом уровне язык программирования изучается для различения основных алгоритмических структур при решении типовых заданий, которые могут встретиться на едином государственном экзамене по информатике. На углубленном уровне изучение возможностей профессионального языка программирования является основной целью для участия в олимпиадах по программированию и продолжения образования в области информационных технологий. Поэтому описание результатов по данному разделу на двух уровнях изучения информатики различаются по формулировкам. В углубленном уровне отсутствует структурный язык программирования Паскаль. Если сравнивать предметные результаты углубленного уровня и рекомендуемое к изучению содержание, то содержание позволяет углубиться в изучение программирования по имеющимся возможностям обучающихся для их подготовки к участию в олимпиадах по программированию.

Предметные результаты по тематическому разделу **«Информационные технологии»** на углубленном уровне нацелены на решение более сложных в вычислительном плане задач с использованием современных программных пакетов и также выходом на задачи по теме *«Искусственный интеллект и анализ больших данных»*. При этом тема искусственного интеллекта должна быть отработана на решении реальных задач этой области.

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ДЛЯ РЕАЛИЗАЦИИ ПРЕДМЕТНЫХ РЕЗУЛЬТАТОВ ПО ИНФОРМАТИКЕ УГЛУБЛЕННОГО УРОВНЯ ОБНОВЛЕННОГО ФГОС СОО

Обзор активных методов обучения и образовательных технологий, актуальных для реализации требований ФГОС СОО и ФРП по информатике углубленного уровня

Как видно из таблицы 1 углубленный уровень изучения информатики характеризуется усложнением аналитического и практического компонентов образовательной деятельности обучающихся. Следовательно, учителю информатики необходимо использовать современные образовательные технологии на основе активных методов обучения, специализированные программные продукты отечественного производства или содержащиеся в реестре разрешенного ПО, а также выбрать для изучения один из языков программирования, который отличался бы многофункциональностью и мог применяться при изучении всех тематических разделов курса информатики углубленного уровня. Более того, углубленная подготовка по информатике должна позволить обучающимся не только сдать успешно ЕГЭ по информатике как предмет по выбору, но попробовать свои силы в олимпиадных соревнованиях, которые дают существенные преимущества или являются альтернативной формой поступления в выбранный вуз.

Активные методы обучения характеризуются высокой степенью включенности обучаемого в учебный процесс, т. е. деятельность обучаемого носит продуктивный, творческий, поисковый характер. Это происходит за счет целенаправленной активизация мышления обучающегося, когда он вынужден быть активным участником образовательного процесса длительное время (в течение всего занятия или выполнения проекта).

Существует множество классификаций активных методов обучения. Наиболее часто используемая классификация по ведущей деятельности обучающихся предполагает разделение активных методов на четыре группы: **дискуссионные, игровые, рейтинговые и тренинговые.** На рисунке 1 можно увидеть, какие активные методы включает в себя каждая из четырех представленных выше групп.

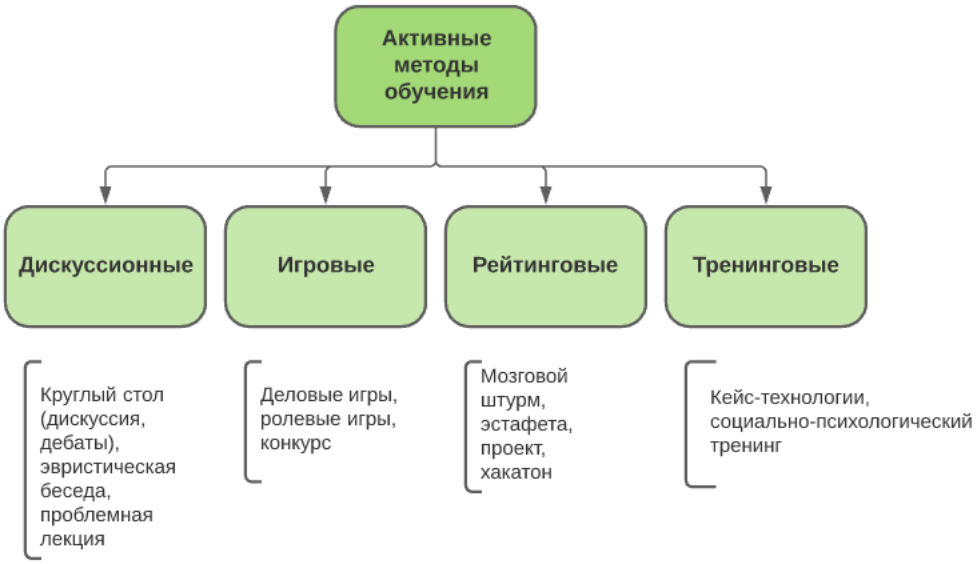


Рис. 1. Классификация активных методов обучения по ведущей деятельности

Рассмотрим их подробнее с примерами использования в курсе информатики.

Дискуссионные (проблемные, эвристические) методы обучения предполагают непосредственный обмен мнениями участников по озвученной проблеме, при необходимости управление процессами выработки и принятия группового решения. К ним относят: проблемную лекцию с элементами дискуссии, эвристическую беседу, поисковый диспут, круглый стол в виде форсайт-сессии или дебатов.

Дискуссионные методы обучения на уровне среднего общего образования необходимо использовать намного чаще, чем это происходит в настоящее время. Это связано с *психологической готовностью* старшеклассников к такому формату обучения, активным формированием их самооценки, вниманием к мнению сверстников. Мыслительная деятельность и интеллектуальные процессы старшеклассников изменились в сторону перехода к мышлению в понятиях (абстрактному), которое глубже и всесторонне отражает взаимосвязь между явлениями окружающей действительности. Поэтому если урок подразумевает освоение большого объема нового материала, то это должен быть не монологический формат объяснения темы, а диалог или дискуссия. Лучше всего использовать эвристическую беседу, проблемную дискуссию или поисковый диспут. Для этого учителю надо выстроить последовательность вопросов, которые он будет задавать в процессе объяснения и наброски выводов (к чему нужно подойти) по каждому вопросу. До урока для обучающихся можно

предусмотреть опережающее домашнее задание. Такое инициирование учителем обсуждение нового материала формирует также «мягкие» навыки обучающихся. Это навыки деловой коммуникации, анализа ситуации, эмоционального интеллекта и настрой на выработку коллективного решения проблемы.

К использованию дискуссионных методов изучения нового материала есть *педагогические предпосылки.* На уровне среднего общего образования продолжается изучение информатики, т. е. происходит расширение и углубление уже знакомого теоретического материала, а также расширяются варианты его практического применения. Например, по тематическому разделу «Цифровая грамотность», рассчитанного на 24 часа учебного времени, распределение тем и часов следующее. На изучение компьютера как устройства обработки информации – 6 часов, программное обеспечение изучается также 6 часов, компьютерные сети – 5 часов, темы информационной безопасности – 7 часов. Эти же темы изучались на уровне основного общего образования, в старшей школе идет их расширение и углубление, поэтому, опираясь на имеющиеся у обучающихся знания, актуализировав их, можно в дискуссионном формате изучить тему на более высоком уровне. Примерные вопросы и выводы по теме «Компьютер» предложены в таблице 2.

Таблица 2

Примерные вопросы для изучения темы «Устройство компьютера» с использованием проблемного метода обучения

Проблемные вопросы	**Предполагаемые ответы/ комментарии**	**Содержание нового материала**	**Примерные выводы**
Почему (за счет каких решений) размеры компьютеров уменьшаются, а функциональные возможности увеличиваются?	Мотивационный вопрос, задающий направление обсуждения. Ответ будет получен ближе к концу урока	Из каких устройств состоит компьютер? Повторение изученного материала в основной школе	При повторении целесообразно построить ментальную карту по теме, чтобы при систематизации знаний было понятно, что нарастили по теме

Проблемные вопросы	Предполагаемые ответы/ комментарии	Содержание нового материала	Примерные выводы
Какое из перечисленных устройств является основным, без которого не существует компьютер?	Процессор	Повторяем или рассматриваем заново последовательно: логический элемент (инвертор, конъюнктор, дизъюнктор) – комбинированные элементы (вентили) – функциональные элементы (сумматор, регистры) – чипсеты. Какие элементы есть в процессоре?	Как устроен и как взаимодействует с другими устройствами процессор? Выход на технологию производства микросхем и архитектурные решения
Может ли быть у компьютера много процессоров и как они будут взаимодействовать между собой?	Вопрос предполагает переход на обсуждение возможностей суперкомпьютеров	Сущность параллельных вычислений. Распределенные вычислительные системы. Суперкомпьютеры и задачи, которые они решают	Повторили фон Неймановскую архитектуру и сравнили ее с Гарвардской (материал по теме есть в учебнике: И.А. Калинин, Н.Н. Самылкина «Информатика.

Проблемные вопросы	Предполагаемые ответы/ комментарии	Содержание нового материала	Примерные выводы
			10 класс: углубленный уровень»)
К какой архитектуре можно отнести мобильные устройства?	Приводятся аргументы в пользу отнесения к какой-либо архитектуре и выдвигается предположение, что есть специальное решение для мобильных устройств	Об архитектуре для процессоров мобильных телефонов (новый материал)	Роботы и другие устройства в различных промышленных производствах, характеристики (новый материал)
Какими вы видите цифровые устройства через 20 лет?	Форсайт-вопрос. Анализ прогнозов ИТ-компаний из сети (опережающее домашнее задание)	Прогнозируем будущее в индивидуальном или командном формате (новый материал)	Получим список новых характеристик цифровых устройств (новый материал)

Круглый стол в виде форсайт-сессии или дебатов, может быть отдельным уроком или его частью, если предусматривается решение задач или практическая работа в конце урока или новый материал в начале урока. Различаются они тем, что форсайт-сессия предполагает взгляд в будущее, его прогнозирование с опорой на лучшие решения / практики, а в дебатах отстаивают противоположные точки зрения на возможные варианты развития событий. Дебаты и форсайт-сессия могут быть индивидуальными или групповыми.

Для организации дебатов по информатике в разделе «Цифровая грамотность» подходят следующие темы / вопросы:

1. Какую операционную систему выбрать на свой компьютер? (Что мы знаем о преимуществах и недостатках различных классов ОС?)

2. Облачные технологии – наше будущее или вынужденное решение крупных компаний?

3. Техногенные или экономические угрозы, связанные с использованием ИКТ, принесут больше вреда обществу?

Игровые методы обучения основаны на имитации ситуаций реальной учебной или профессиональной деятельности, формируют навыки решения проблем в совместной деятельности участников, когда по-другому их сформировать невозможно. Выделяют наиболее популярные: деловые игры, конкурсы-инсценировки, сторителлинг. Игровые методы требуют большой подготовительной работы всех участников образовательного процесса, поэтому используются нечасто, особенно для старшеклассников. Но игровые элементы, входящие в технологию геймификации (см. далее) стали довольно популярными в образовании. Традиционно через деловые игры по информатике реализуются межпредметные связи, например в электронных таблицах решаются задачи с экономическим содержанием. В ФРП по информатике стало больше математических тем и значительно расширились темы искусственного интеллекта и анализа больших данных, которые также можно будет использовать в межпредметных деловых играх. Например, по интегрированным темам информатики – программирование, экспертные системы, правовые вопросы информационной безопасности и обществознания или экономики.

Рейтинговые методы обучения активизируют деятельность обучающихся за счет эффекта соревнования. Наиболее часто используемые методы: мозговой штурм, эстафета, мини-проект, хакатон. Соревновательные преимущества можно реализовать при решении задач разными способами, разборе наиболее трудных задач на уроках. Если для структурирования изученного материала строится общая ментальная карта или осуществляется коллективная работа с документом, изображениями, сайтом, можно использовать эстафету, которая поможет удержать темп урока и внимание всех его участников до конца работы.

Тренинговые методы обучения направлены на оказание стимулирующего, корректирующего или развивающего воздействия на личность и поведение участников. Сюда относятся текстовые (ситуационные) задачи, кейсы, социально-адаптивные тренинги и проч. Подробнее работу с кейсами рассмотрим на примере новой для информатики темы «Основы шифрования». В приложении 1 даны два несложных кейса на шифры перестановки. При изучении темы в классе кейсы могут быть диагностическим инструментом, с их помощью можно проверить усвоение темы, даже у тех обучающихся, которые отсутствовали на предыдущем уроке. В кейсе описана ситуация, требующая решения, и приведены подробные примеры способов решения заданий по теме, что позволит по образцу выполнить кейс. Можно использовать альтернативные решения, например, используя программный код. Кейсы предполагают разные способы решения заданий. Из предлагаемых кейсов легко сделать фасетные задания, получатся индивидуальные кейсы.

На основе приведенных активных методов обучения выделяют следующие образовательные технологии, применение которых наиболее актуально на уровне среднего общего образования:

1) **технологию геймификации** (использование игровых элементов вне игрового контекста);

2) **кейс-технологию** (анализ конкретных ситуаций и выработка различных стратегий решения проблемы);

3) **проектную технологию** (способ достижения цели через детальную разработку проблемы, которая должна завершиться практическим результатом);

4) **смешанное обучение** (сочетание традиционного и дистанционного обучения, гибкое изменение форматов деятельности обучающихся).

Когда активные методы обучения в совокупности с ИКТ-средствами и наиболее рациональными организационными формами обучения составляют целостную образовательную технологию, позволяющую достичь планируемые результаты в виде цифровых компетенций, необходимо продумать, какое предметное содержание лучше всего реализуется выбранными образовательными технологиями. Рассмотрим наиболее популярные образовательные технологии в применении к обучению информатике.

Технологию геймификации лучше применять для освоения таких тем, где нужна повышенная мотивация для изучения. Именно игровые элементы

(правила игрового пространства, баллы, бейджи, рейтинг-листы) позволят поддерживать высокую мотивацию длительный период, пока тема не будет освоена. Повышенная мотивация необходима для наиболее трудных тем курса информатики. Традиционно такой темой считается программирование, следовательно, и подготовка к различным олимпиадам, где ведущей является деятельность по программированию.

Поэтому для достижения высоких результатов в программировании необходимо как можно раньше начинать программировать в визуальных средах программирования, где используются игровые элементы в программировании. Создается специальное игровое пространство, накапливаются очки, подсчитываются рейтинги участников. Совсем не обязательно конечной целью ставить создание готового программного продукта, целью является вовлечение в деятельность по созданию кода для решения задачи доступными средствами. На уровне среднего общего образования программирование является почти профессиональным инструментом, программистские навыки необходимо оттачивать именно как программистам, участвуя в различных индивидуальных и групповых соревнованиях (хакатонах) по программированию, не забывая, что в профессиональных сообществах также существуют рейтинги (индивидуальные и командные), есть правила участия, похожие на правила игрового пространства, где можно набирать или терять баллы (очки). В качестве командных соревнований и тренингов внутри класса, школы или города наиболее приемлемым является хакатон. Для организации хакатона нет возрастных ограничений. Необходимо учитывать возраст обучающихся при постановке командного задания, выборе программного пакета и времени работы. Чем ниже возраст, тем меньше времени на работу и тем более интерактивна должна быть среда программирования.

Кейс-технологии все шире входят в образовательную практику. Кейсом называют ситуацию, которая подразумевает наличие нескольких способов решения проблемы. Описание ситуации в виде кейса содержит все необходимые данные и способы их обработки для поиска решений. Любую «закрытую» задачу можно сделать «открытой» или кейсом. Кейсы можно использовать по любой теме курса информатики, предварительно подготовив необходимый материал. Предлагаем воспользоваться набором кейсов для изучения новой темы

по информатике «Основы шифрования» (см. Приложение 1) и социальной инженерии (правовые основы информационной безопасности) (Приложение 2). Кейсы могут быть хорошим мотивационным или диагностическим инструментом. В тематических кейсах по основам криптографии и правовым основам информационной безопасности содержаться все необходимые данные для решения проблемной ситуации и демонстрируются примеры решения или комментируются подходы к решению.

Проекты вошли в систему образования как метод активного обучения. В настоящее время проектный подход к решению проблем в любой сфере деятельности можно отнести к проектной технологии. Те темы курса информатики углубленного уровня, которые могут «вырасти» из практических работ, т. е. можно сформулировать новые задания к выполненной работе, ввести и использовать дополнительные параметры в заданиях, подключать или перепрограммировать дополнительные устройства или среды, требуют развитого аналитического мышления обучающихся (креативности), можно реализовать через учебные или исследовательские проекты. Например, по тематическому разделу **«Цифровая грамотность»**, исходя из обновленного содержания ФРП углубленного уровня, можно выделить следующие темы проектов:

1. *Гарвардская архитектура микропроцессоров и ее применение* (учебное исследование).

2. *Многопроцессорные системы. Суперкомпьютеры* (учебное исследование).

3. *Система контроля хранения продуктов (на плате Arduino)* (практико-ориентированный учебный проект).

4. *Новогодняя гирлянда (на плате Arduino)* (практико-ориентированный учебный проект).

5. *Простые методы шифрования* (учебный вычислительный проект).

6. *Цифровая подпись (реализация алгоритма RSA)* (учебный вычислительный проект).

7. *Технология блокчейн* (учебный вычислительный проект).

Для тем 3–7 можно использовать учебное пособие: Самылкина Н. Н., Калинин И. А., Тарапата В. В., Салахова А. А. Информатика: 8–11-е классы: практикум. – М. : Просвещение, 2023. – 157 с.: ил. – URL: https://lbz.ru/books/1171/

По тематическому разделу **«Теоретические основы информатики»** можно обучающимся предложить такие темы проектов:

1. *Двухразрядный последовательный сумматор* (практико-ориентированный учебный проект) [6].

2. *Асинхронный RS-триггер* (практико-ориентированный учебный проект).

3. *Синхронный RS-триггер* (практико-ориентированный учебный проект).

4. *Автоматизация работы склада в среде имитационного моделирования AnyLogic* [6].

5. *Обеспечение безопасности обучающихся в школе на примере реализации агентной модели в среде имитационного моделирования AnyLogic»* [4].

6. *Оптимизация работы поликлиники с использованием среды имитационного моделирования AnyLogic* [4].

7. *Исследование модели распространения эпидемии в среде имитационного моделирования AnyLogic* [4].

8. *Исследование системно-динамической модели работы сотовой компании среде имитационного моделирования AnyLogic* [4].

По тематическому разделу **«Алгоритмы и программирование»** обучающимся будет интересно выполнить проекты по темам:

1. *Создание чат-ботов в Telegram.*

2. *Разработка 2D-игры на платформе Unity.*

3. *Эксперименты по микроэлектронике на JavaScript*.

4. *Реализация алгоритма CART в углубленном курсе информатики* [3].

По тематическому разделу **«Информационные технологии»** будут востребованы прикладные проектные работы по темам:

1. *Чем занимаются инженеры?* (о специфике инженерной деятельности и профессии будущего).

2. *Большие данные. Откуда они берутся и как могут помочь?*

3. *Почему программист – инженер? Как мы делаем программы*.

4. *Что такое интеллектуальные задачи и интеллектуальные методы решения задач?*

5. *Нейронные сети – что они могут?*

6. *Безопасность информационных систем как инженерная задача*.

7. *Трехмерное моделирование и прототипирование в программе T-FLEX CAD»*.

8. *Разработка экспертных систем (например, виртуальный доктор).*

Для тем 2, 4, 5, 8 можно использовать учебное пособие: Калинин И. А., Самылкина Н. Н., Салахова А. А. Искусственный интеллект: 10–11 классы: учебное пособие. – М.: Просвещение, 2023. – 144 с. [5].

Для темы 7, связанной с трехмерным моделированием и прототипированием, в Приложении 3 к данному методическому пособию подготовлена практическая работа.

Смешанное обучение позволяет часть тем вывести на дистанционное изучение. К таким темам можно отнести темы, которые выделены курсивом в Рабочей программе по информатике. Дистанционное обучение можно использовать для консультаций и обсуждения различных этапов проектной деятельности. Электронное обучение подразумевает использование цифровых образовательных ресурсов по различным темам, предварительно подготовленных для обучающихся определенной возрастной категории. Дистанционное и электронное обучение являются обязательными компонентами цифровой образовательной среды школы. Смешанное обучение считается наиболее эффективным в обучении информатике, так как требует высокого уровня владения средствами ИКТ, навыками создания цифровых образовательных ресурсов, экспертного уровня методической подготовки для гибкого подхода при выборе моделей (перевернутый класс, ротация станций, личный выбор, ротация лабораторий и проч.) реализации смешанного обучения.

Методические рекомендации по изучению тематического раздела «Цифровая грамотность»

Тематический раздел **«Цифровая грамотность»** на углубленном уровне представлен следующими темами:

1. Компьютер – универсальное устройство обработки данных (6 ч).
2. Программное обеспечение (6 ч).
3. Компьютерные сети (5 ч).
4. Информационная безопасность (7 ч).

В ФРП предусмотрено четыре практических работы. При этом можно использовать короткий и длинный форматы, т. е. практическую работу могут составлять несколько коротких взаимосвязанных упражнений по теме или полностью разработанная практическая работа, интегрирующая несколько тем.

Предметные результаты на двух уровнях изучения данного раздела почти не отличаются и представлены в таблице 1.

По тематическому разделу «Цифровая грамотность» сложными темами являются темы «Компьютерные сети» и «Информационная безопасность». В разделе появилась новая, ранее не рассматриваемая тема «Шифрование данных» в контексте основной темы «Информационная безопасность».

Сложность указанных тем связана с сильным упрощением рассматриваемых понятий темы на уровне основного общего образования, возникновением в этом случае разрыва в понимании между тем, как функционируют сети в реальности и как об этом написано в школьном учебнике. На углубленном уровне изучения темы для обучающихся, которые планируют работать в отрасли информационных технологий, нужно рассказывать без существенных упрощений, как работают компьютерные сети. Адаптация отобранного содержания для старшеклассников должна быть минимальна и реализована за счет акцентов при разборе основных определений и многократных повторений материала, а также систематизации изученного материала с использованием вопросов и заданий. Содержание темы «Компьютерные сети» выбрано для подробного рассмотрения в связи еще и с тем, что она отсутствует в тематическом разделе «Информационные технологии» и полностью реализуется в разделе «Цифровая грамотность», хотя

акценты при объяснении обучающимся необходимо делать именно на сетевые информационные технологии.

Рассмотрим содержание темы «Компьютерные сети», которую необходимо освоить за пять уроков. Учителю информатики необходимо также предусмотреть практическую работу или несколько практических упражнений в процессе изучения предлагаемого содержания.

Общие понятия и структура компьютерных сетей

Содержание темы урока

Современные решения в области компьютерных телекоммуникационных сетей – одно из основных направлений развития информационных технологий, своеобразная визитная карточка отрасли.

Компьютерная телекоммуникационная сеть – программно-аппаратный комплекс, обеспечивающий обмен данными между компьютерами по каналам связи в автоматическом режиме.

Ключевыми характеристиками такого комплекса будут:

– время передачи сообщений – чем меньше это время, тем лучше;

– надежность передачи – чем меньше вероятность сбоя, тем больше задач можно решить;

– автоматизация обмена данными;

– разнообразие решаемых задач, т. е. степень универсальности.

Обратим внимание на несколько особенностей:

– обмен ведется по некоторому *каналу связи*. Чтобы учесть требования скорости, мы рассматриваем только те каналы, которые достаточно быстры, поэтому чаще всего речь идет об электромагнитных или световых каналах;

– сеть – это комплекс, поэтому нельзя исключить из него программы или аппаратуру;

– обмен данными – единственная задача, которую решает сеть. Все возможные применения полученной по сети информации – задача программ и пользователей этих программ.

С общей точки зрения сеть состоит из *станций* (*узлов* сети, участников обмена) и *линий* взаимодействия, т. е. *каналов связи*.

При проектировании сети (в подавляющем большинстве случаев – при подборе и сборке элементов типового решения) необходимо в первую очередь определить ее назначение.

При этом часто пользуются классификацией сетей (точнее – технологий, заложенных в их основу) по основным критериям: скорость обмена данными, возможное количество узлов и дальность связи.

По этим параметрам выделяют два основных класса сетей:

1) *локальные* – сети с ограниченными расстоянием для обмена и количеством узлов. За счет этих ограничений такие сети, как правило, имеют высокую скорость и низкую стоимость. Следует обратить внимание, что технологии этих сетей ограничивают количество узлов и обязательное время ответа, т. е. нельзя «просто увеличить мощность» и получить сеть любого размера;

2) *территориальные* – сети, которые могут быть развернуты на большой территории, т. е. для них нет технологического ограничения на длину каналов и количество узлов, в результате скорость передачи данных по таким сетям может быть ниже, но зато сеть может быть гораздо крупнее.

Если территория – несколько континентов, то такая сеть может называться *глобальной*. В современных условиях выделить одну глобальную сеть очень трудно. Фактически такие сети объединены в общую среду, которую и называют Глобальной сетью.

Вне зависимости от размера территории в основе современных компьютерных сетей лежит несколько общих принципов:

1) *пакетная передача данных*; суть этого принципа в том, что сообщение передается независимыми частями – пакетами. То есть перед началом отправки сообщение разбивают на пакеты, а в конечной точке собирают снова;

2) *модульная архитектура*; современные сети чаще всего создаются из отдельных элементов, которые должны соответствовать общеизвестным стандартам, – это позволяет при создании сетей использовать разные модули (например, от разных производителей), подбирая оптимальную конфигурацию и изменяя ее по мере необходимости;

3) *открытая архитектура*, которая дает возможность разрабатывать всевозможные дополнения, позволяющие модернизировать уже имеющиеся компоненты и разрабатывать новые компоненты самого разного назначения.

Соблюдение данных принципов обеспечивается в первую очередь использованием общих стандартов. Большинство этих стандартов опубликовано и широко известно. Материалы этих уроков основаны на этих стандартах.

Для описания компонентов сетей и их взаимодействия разработаны общие структуры, получившие название *модели сетевого обмена*. Наиболее известны две из них – DOD и ISO/OSI.

Обе эти модели разделяют одну и ту же процедуру приема и передачи данных между специализированными программами на этапы – *уровни*. Уровни взаимодействуют между собой строго по порядку, на каждом уровне решаются некоторые типовые задачи (см. табл. 3).

Модели различаются между собой количеством уровней, но решаемые задачи, конечно, одни и те же.

Таблица 3

Модели сетевого обмена

DOD	ISO/OSI
Прикладных задач	Прикладной
	Представлений
	Сеансовый
Транспортный	Транспортный
Сетевой	Сетевой
Доступа к среде	Канальный (каналов множественного доступа)
	Физический

Модель DOD была использована при проектировании и разработке самого известного набора сетевых протоколов – TCP/IP. Со временем при проектировании и описании сетевых технологий и протоколов стали использовать более общую и подробную модель ISO/OSI, но ни одного полностью основанного на ней законченного решения создано не было. Тем не менее взаимодействие и назначение отдельных компонентов легче показывать с ее помощью.

В обеих моделях уровни выделяются от «человеческих» задач (вверху) до задачи непосредственного обмена сигналами (внизу). Опишем уровни модели DOD:

1. *Уровень доступа к среде* – аппаратура (приемники, передатчики, микроэлектронные компоненты приема и передачи, кодирования сигнала и т. п.) и средства (например, кабели) передачи сигнала. Этот уровень описывает одну сеть на базе взаимодействующих компонентов. Пакеты, выделенные на этом уровне, как правило, называют кадрами, потому что речь идет об участке сигнала.

2. *Сетевой уровень.* На этом уровне собраны средства объединения сетей – правила их взаимодействия, пересылки данных и т. д. Поскольку уровень объединяет разнородные сети в единое целое, то и называется «сетевым». Непосредственной передачей занимается уровень доступа к среде, которому передаются *датаграммы* (часть данных, которая передается без предварительных действий – установления соединения).

3. *Транспортный уровень.* Это компоненты, обеспечивающие обмен данными между отдельными приложениями, т. е. транспорт данных. Именно здесь формируется *пакет* данных, который передается программам сетевого уровня для передачи.

4. *Прикладной уровень* – объединение всех программ, с которыми непосредственно работает пользователь для решения прикладных задач (просмотра страниц, чтения почты и т. д.). На этом уровне мы говорим об осмысленных с точки зрения пользователя объектах приема и передачи – сообщениях, файлах и т. д.

Большое значение для любого сетевого обмена имеет **протокол** – *общие правила передачи и приема информации, включая ее представление для этого и служебные данные.* Соблюдение протокола – необходимое условие правильного обмена, особенно тогда, когда участники обмена работают через цепь посредников и не имеют возможности согласовывать что-либо напрямую. Поэтому подавляющее большинство современных сетевых протоколов разработано коллегиально и широко опубликовано для того, чтобы все участники имели возможность верно организовать работу.

Если протокол получает данные для дальнейшей обработки от другого протокола, то выполняется процедура инкапсуляции – добавляются (как правило, в начале) служебные данные протокола, а все полученное «сверху» воспринимается как готовые данные и передается в неизменном виде

(т. е. данные помещаются «внутрь»). Служебные данные таким образом чаще всего передаются в виде ***заголовка***. Структура и правила формирования и обработки заголовков определяются протоколом.

Как мы уже упоминали, реализуется сетевая модель тем или иным набором взаимодействующих протоколов. Поскольку сетевые модели используют принцип строгой последовательности, то набор протоколов называют ***стеком.***

Поставим в соответствие уровням названия частей данных и протоколы в самом распространенном стеке (табл. 4).

Таблица 4

Уровень	**Название части данных**	**Стек TCP/IP**
Прикладных задач	Сообщение	DNS, HTTP, SMTP, POP3 и др.
Транспортный	Пакет / датаграмма	TCP, UDP
Сетевой	Датаграмма / Фрагмент	IP, ICMP, ARP, BGP и др.
Доступа к среде	Кадр	Нет компонентов

Поскольку большая часть нашего описания будет относиться к стеку протоколов TCP/IP, то мы будем для структурирования материала опираться на модель DOD.

Вопросы и задания для закрепления теоретического материала

1. Приведите примеры использования локальных и территориальных сетей.

2. Почему стек TCP/IP не имеет протоколов на уровне доступа к среде?

3. К какому уровню модели DOD будет относиться протокол передачи звука, видео?

4. Почему передача данных в современных сетях почти всегда пакетная?

5. Мы знаем, что сейчас пакеты даже через океан передаются очень быстро и каналы связи там очень мощные. Тем не менее это не локальные сети. Почему?

6. Зачем придумана инкапсуляция? Передавали бы данные сразу сообщением или сразу подготовленным набором пакетов.

7. Если уже существует технология, которая позволяет передавать данные через океан, почему бы не использовать ее везде?

Ответы на вопросы

1. Технология, оптимальная для сети школьного здания, класса, рядом расположенных зданий с количеством абонентов до 1000, жилого здания – локальная. Технология для построения сети связи в масштабах города, крупного предприятия с разнесенными зданиями – территориальная.

2. Потому что он предполагает работу по любым каналам связи, обеспечивающим доставку датаграмм / фрагментов за оговоренное в настройках время. Это позволяет совершенствовать технологии связи и передавать данные между разными сетями (в том числе используя сети только для транспорта).

3. К прикладному. Остальные уровни никак не различают передаваемые через них данные.

4. Потому что это позволяет множеству пользователей использовать общий канал связи, например при пересылке данных через другие сети, определять повреждения и реагировать на потерю не всего сообщения, а отдельных пакетов.

5. Потому что локальные сети строят исходя из предположения, что количество узлов в такой сети ограничено сравнительно небольшим числом, и при этом есть обязательные параметры (например, пауза для проверки свободен канал, или нет), которые ограничивают дальность и количество узлов. Например, в технологии GigabitEthernet стандарт определяет максимальную длину линии (оптический кабель) в 5 км. Если линия будет длиннее, то устойчивую работу гарантировать нельзя. В трансконтинентальных каналах линии ОЧЕНЬ длинные и сложные, так что там все эти параметры настраиваются.

6. Сети сильно отличаются и создаются в разное время, поэтому они могут предъявлять к пакетам разные требования. Стандарты с «переупаковкой» пакетов и датаграмм позволяют нам передавать данные, адаптируя их к конкретным сетям или задачам их взаимодействия – не переделывая сами сети.

7. Во-первых, это дорого – такие сети предъявляют высокие требования к каналам, требуют сложного и дорогого оборудования. Во-вторых, это избыточно сложно – для задач передачи данных внутри квартиры почти ничего из этих возможностей не нужно. В-третьих, к ним нельзя подключить, например, смартфон – вам ведь нужно с ним перемещаться.

Доступ к среде или сетевые технологии

Содержание темы урока

Уровень доступа к среде – это средства физического обмена данными между двумя произвольными компьютерами в сети. Очевидно, что обмен потребует наличия:

– среды, которая будет меняться;

– аппаратуры, которая эти изменения будет фиксировать;

– аппаратуры, обеспечивающей доставку данных от одного компьютера до другого;

– программ, которые будут переводить числовую информацию в эти изменения (передатчика) и изменения назад в информацию (приемника).

Средства, которые реализуют эти задачи, называются ***сетевой технологией***. Именно сетевая технология будет определять возможности конкретной сети (скорость передачи данных, расстояния, количество станций), поэтому она выбирается исходя из предполагаемого назначения сети, т. е. ее класса – локальная или территориальная.

Чтобы описать конкретную сеть, обычно описывают и принцип создания физических каналов связи между узлами. Такие общие принципы называют ***топологией сети***.

Чаще всего в локальных сетях используют три основных топологии, которые уже рассматривались в основной школе:

1) *«общая шина»*, когда все станции подключены к общему каналу и обмениваются данными через него;

2) *«звезда»,* когда все подключены к общему центральному устройству, которое и передает данные каждому участнику;

3) *«кольцо»,* когда все станции передают данные строго по порядку от одной к другой, образуя кольцо.

О простой типовой топологии можно говорить только для небольшой локальной сети. В крупных сетях часто сочетают несколько разных топологий, причем они могут меняться с течением времени. Поэтому топологию крупной сети нельзя описать коротко.

Часто в крупных сетях реализуют так называемую *ячеистую топологию* – схему соединений, при которой каждая станция соединяется минимум с двумя другими. В такой топологии отказ любого канала не приведет к прекращению работы.

Сетевых технологий существует множество, наиболее популярные и доступные мы рассмотрим подробнее.

Ethernet – самая распространенная технология создания локальных сетей. Под этим названием скрывается несколько семейств стандартов с общими принципами, разработанных в разное время. Современные варианты этой технологии позволяют достичь скорости передачи до 25 Гбит/с (и собрать канал из четырех таких линий, т. е. передавать до 100 Гбит/с). Ее роль велика еще и потому, что именно этот стандарт чаще всего используют как стандартный стык, т. е. самые разные устройства (например, работающие с другими сетями) имеют возможность подключения к такой сети.

В основе всех сетей Ethernet лежит общий принцип, получивший название «сети с контролем несущей, множественным доступом и определением коллизий» (Carrier Sense Multiple Accesswith Collision Detection, CSMA/CD).

Суть этого принципа проще всего пояснить с помощью аналогии: разговор нескольких человек в одной комнате. Все участники «слушают» несущую волну, т. е. слушают все звуки в комнате, имеют равные возможности начать передачу (не запрашивая разрешения) для любой другой станции. Каждая станция «услышит» переданные пакеты и выберет те, которые предназначены ей.

Перед началом передачи каждая станция в течение строго заданного времени «прислушивается» к среде и начинает передачу, только если в течение этого времени никто другой ничего не пытается передавать.

Если передачи двух станций все-таки «наложатся» друг на друга, то такая ситуация называется *коллизией*, все станции считают такие данные испорченными (определяют коллизию), а передававшие станции прекращают передачу, ждут случайное время (небольшое, но разное для обеих станций) и снова пытаются передать данные (прослушав канал).

На этом уровне передаваемые фрагменты сообщений фактически представляют собой фрагменты изменяющейся в течение заданного времени волны-сигнала и поэтому называются *кадрами*.

Для идентификации (т. е. направления данных конкретной станции, а не всем станциям вообще) каждая станция при доступе к среде передачи имеет адрес (Media Access Control, MAC).

В модели ISO/OSI задачи передачи сигналов через каналы и взаимодействия отдельных станций разделены на два уровня – физический

(непосредственная передача сигнала) и среды общего доступа (Multi Access Chanel, MAC-уровень).

Подключение узла к сети выполняется с помощью специального устройства, которое называется ***сетевым адаптером***. Сетевой адаптер может быть выполнен в виде отдельной платы, USB-устройства, встроенного устройства материнской платы и т. д. Его функция – обеспечивать взаимодействие компьютера с цифровой сетью. Сетевой адаптер выполняет все функции по непосредственному приему и передаче кадров.

Первоначально технология Ethernet использовала для передачи данных общий кабель, т. е. сети имели топологию «общая шина». Со временем это стало препятствием в создании крупных сетей, так как длина сегмента кабеля ограничена, ограничивало скорость передачи (частота передачи должна быть общей) и при эксплуатации порождало много проблем (любое повреждение кабеля или контактов прекращало работу всей сети).

Сейчас сети Ethernet строятся с использованием топологии «звезда» – все станции подключаются к общему устройству – ***коммутатору*** (Switch), а оно обеспечивает логическую общую шину на MAC-уровне. Ранее применяли и более простой тип – концентратор, но их осталось мало. Коммутатор повторяет кадр только для той линии, на которой зарегистрирована станция, что позволяет существенно уменьшить количество коллизий и создавать значительно более сложные сети (например, логически разделенные на несколько сегментов).

С технической точки зрения современные реализации этой технологии чаще всего используют как среду передачи 8-жильный медный кабель с попарно свитыми жилами (так называемую неэкранированную витую пару) или двухжильный оптический кабель. Медный кабель может иметь длину до 100 м, оптический – до 5 км без специального оборудования, а со специальным – до 80 км.

Коммутаторы и концентраторы можно подсоединять друг к другу, образуя сложную и разветвленную структуру.

Количество станций в сети Ethernet не может превышать 1024, поэтому даже с применением оптического оборудования эти сети остаются локальными и предназначены для организации работы в помещениях.

Сети стандарта Ethernet обеспечивают скорость передачи данных 9,76 Мбит/с (Ethernet), 97,6 Мбит/с (Fast Ethernet) и 1000 Мбит/с (Gigabit Ethernet).

Достоинствами сетей Ethernet различных стандартов являются простота монтажа и обслуживания, доступность большого количества разнообразного оборудования от самых разных производителей. К основным недостаткам нужно отнести падение производительности при возрастании нагрузки (из-за коллизий), трудности с обеспечением гарантированной скорости обмена.

С удешевлением оптического оборудования и появлением провайдеров, предоставляющих в аренду высокоскоростные каналы связи, Ethernet стали использовать и для построения территориальных сетей небольшого размера в тех случаях, когда передача компьютерных данных является основной (а иногда и единственной) задачей такой сети.

Сети **WiFi** (от англ. Wireless Fidelity – беспроводная точность) – самые распространенные беспроводные сети. В качестве средства передачи данных они используют радиосигнал и образуют сеть с топологией «общая шина».

В их основе лежит принцип, похожий на принцип сетей Ethernet, но вместо определения коллизий для снижения издержек применяется метод предупреждения коллизий (Carrier Sense Multiple Accesswith Collision Avoidance – CSMA/CA). В отличие от CSMA/CD перед началом отправки кадра данных станция должна послать сигнал затора и некоторое время ждать положительного ответа от всех станций, которые его получат. Если сигнал занятости будет получен во время передачи (станцией, вовремя не получившей предупреждение), то передача будет прекращена. Таким образом, коллизии данных практически исключаются.

Чаще всего такие сети строятся на базе точек доступа (AccessPoint), которые выполняют функцию коммутаторов и соединяются либо с опорной проводной сетью, либо с другими точками доступа. Очень часто современные точки доступа входят в состав более сложных и универсальных устройств – маршрутизаторов, о которых мы будем говорить ниже.

Сети WiFi имеют несколько стандартов, скорость передачи данных в которых варьируется от 1 до 9,6 Гбит/с. Дальность связи в помещениях без специального оборудования редко превышает 40–50 м. Мощность сигнала сильно снижают металлические препятствия, бытовая техника, зеркала, железобетонные стены и перекрытия и т. д. В таких сетях невозможно гарантировать скорость или время доставки пакетов, оно будет зависеть от препятствий, количества пользователей и других условий.

Беспроводные сети позволяют организовать доступ там, где прокладка кабеля невозможна или его подключение затруднено (например, к мобильному телефону или фотоаппарату – размер разъема Ethernet для них велик).

Поскольку для обмена данными используются радиосигналы, то в таких сетях с самого начала предусматривается защита данных от перехвата и подмены с помощью шифрования. Современные и надежные методы шифрования вычислительно трудоемки, поэтому это один из факторов, ограничивающих количество клиентов, одновременно подключенных к одной точке доступа.

Сети WiFi из-за ограниченных полосы пропускания, дальности связи и количества подключенных точек применяются, как правило, для организации сравнительно небольших сетей (или сегментов сетей) в помещениях – домах, гостиницах, кафе и т. д.

Мы не рассматриваем здесь специализированные сетевые технологии, которые применяются либо в специфических условиях (например, для передачи данных по электрическим проводам – «Интернет через розетку») или для особых задач. Технологий довольно много, и они постоянно изменяются – но все сети могут взаимодействовать между собой.

Магистральные сети. Перечисленные нами технологии «близки» к пользователям. Между тем они не очень хорошо подходят для построения магистральных сетей, предназначенных для использования на больших расстояниях и передачи очень больших объемов данных. Помимо этих общих требований, магистральные сети должны быть универсальными, т. е. обеспечивать передачу разных типов данных (например, телефонных разговоров, интернет-данных и т. п.) и взаимодействие с уже существующими сетями.

Построение таких сетей – отдельная, чрезвычайно сложная задача. В современных условиях используется несколько конкурирующих и взаимодействующих технологий[1], позволяющих строить сети любой топологии и дальности, обеспечивая огромную пропускную способность.

Ключевые требования к таким сетям:

– высокая скорость;

– возможность создания и использования дублирующих каналов связи;

[1] Несколько сокращений: ATM, SDH/SONET, DWDM, MILPS.

– возможность передавать потоки данных разного назначения;

– использование существующих каналов связи с максимальной эффективностью;

– возможность диагностики и контроля их работы.

Практически все современные магистральные сети построены на базе волоконно-оптических каналов связи. Примерную схему таких каналов, создаваемую ежегодно, можно увидеть на сайте http://www.telegeography.com. Там вы можете увидеть, как выросла пропускная способность каналов связи за 10 лет. Определить их общую пропускную способность трудно (простым суммированием это сделать нельзя). Чаще всего сейчас используются каналы SDH[1] STM-16, обеспечивающие скорость 2,5 Гбит/с.

Вопросы и задания для закрепления изученного материала

1. Подготовьте таблицу сопоставления возможностей часто используемых сетевых технологий, например по среде передачи сигнала, скорости передачи, расстоянию связи, возможному количеству узлов.

2. Для каких целей при разработке сетевой технологии может понадобиться уменьшение расстояния возможного установления связи?

3. Какие ограничения технологий Ethernet не дают возможности использовать их для построения территориальной сети?

Ответы на вопросы

1. Задание можно использовать в качестве домашней практической работы.

2. Для обеспечения работы в условиях множественных подключений по этой технологии – для обеспечения безопасности и удобства. Например, Bluetooth – предназначена для подключения носимых устройств. Таким образом, вероятна работа нескольких таких сетей рядом (несколько человек в транспорте и т. д.).

3. Обязательный межкадровый интервал (IPG) – для Gigabit Ethernet 96 нс. Если размер сегмента сети (т. е. расстояние) превышает определенные (зависит от среды передачи), то абонент не успевает за это время получить подтверждение свободы канала – и в сети падает производительность. Максимальное количество станций в одной сети, определенное стандартом, – 1024.

[1] Synchronous Digital Hierarchy— синхронная цифровая иерархия (*англ.*).

Сетевой уровень, или Протокол IP и адресация в сети

Содержание темы урока

Самые разные технологии уровня доступа к среде позволяют создать общую среду обмена, т. е. одну сеть. Очевидно, что универсальной такая сеть не будет – слишком разные задачи ставятся при создании. В одних случаях необходимо обеспечить легкость подключения и отсутствие проводов, но допустимо падение производительности и перерывы связи, в других – необходима универсальность, расширяемость и надежность, но не имеет значения стоимость подключения одной точки и т. д.

Тем не менее решаемые задачи очень часто требуют взаимодействия между абонентами разных сетей.

Объединение и взаимодействие сетей обеспечивается с помощью программных протоколов сетевого уровня, за это и получившего свое название. Задача протоколов этого уровня – управление потоками данных, т. е. обеспечение перемещения данных между двумя узлами, возможно расположенными в разных сетях, связанных между собой.

Самое распространенное и универсальное решение этой задачи – использование протоколов, входящих в стек TCP/IP, в частности основного протокола сетевого уровня Internet Protocol (IP). Название этого протокола исчерпывающе говорит о его назначении – это протокол межсетевого обмена.

Стоит обратить внимание на то, что в таблице соответствия уровней и протоколов стека TCP/IP на уровне доступа к среде протоколов не было. Причина этого – универсальность стека протоколов TCP/IP, который изначально должен был объединять любые сети, обеспечивающие пакетный обмен данными.

Протокол IP. Адресация

С точки зрения протокола IP обмен данными ведется между адресами. Группа идущих подряд адресов (не любая, а соответствующая ряду условий) объединяется в *IP-сеть*. IP-сеть (или подсеть) – это понятие на уровне именно этого протокола, и физической сети оно соответствует не всегда. С точки зрения протокола IP, данные между узлами одной сети передаются напрямую, а между разными сетями – через пересылку. Процесс такой пересылки называется *маршрутизацией*.

Протокол IP предусматривает, что каждый узел сети (участник обмена данными) получает аппаратно-независимый адрес (т. е. MAC-адрес в рамках конкретной сети может быть любым, от него IP-адрес не зависит). Действующая сейчас четвертая версия протокола IP предусматривает, что на адрес выделяется 32 бита (4 байта – октета).

Адрес, таким образом, делится на две части: номер сети (первые биты) и номер узла (последние биты). Количество битов на номер сети и номер узла меняется и зависит от предполагаемого количества узлов в сети.

Для технических целей предусмотрено, что:

– адрес, в котором все биты номера сети 0, – это адрес в своей собственной сети;

– адрес, в котором все биты номера узла 0, – это указание на всю сеть целиком;

– адрес, в котором все биты номера узла 1, – это отправка данных любому узлу в этой сети.

Изначально было предусмотрено четыре класса сетей.

Класс А. Крупные сети. На номер сети отводился один байт, причем первый бит обязательно 0. Таким образом, с учетом правил, это номера от 1 до 126[1]. Номер узла – три байта, т. е. возможно примерно 16,7 млн узлов. По очевидным причинам таких сетей мало, и номерная емкость существующих выбрана далеко не полностью.

Класс В. Средние сети. На номер сети отводится два байта, начинающихся с 1 и 0 (т. е. первый байт в диапазоне от 128 до 191), на номер узла – два байта. В такой сети может быть 65 536 – 2 = 65 534 узла. Этот класс сетей – самый популярный у интернет-провайдеров.

Класс С. Маленькие сети. На номер сети отводится три байта, с началом 1,1,0 (первый байт от 192 до 223) и 1 байт на номер узла в сети, т. е. это сети размером до 254 рабочих станций.

[1] Внимательный слушатель, хорошо знающий двоичную систему счисления, в этом месте должен спросить: а куда делся адрес 127? Считается, что все адреса, начинающиеся со 127, это адреса узла, с которого выполняются обращения. Это позволяет узлу обратиться к самому себе – для проверки работы программ протокола и обращения к службам, запущенным на своем узле. Чаще всего используется адрес-петля 127.0.0.1.

Класс D. Массовая рассылка – IP-адрес идентифицирует только сеть в целом, а внутри данные рассылаются всем, кто может их принять. Применяется, например, для организации IP-телевидения.

Распределением IP-адресов занимается специальная международная организация – IANA.

Таблица 5

Класс	Первый байт	Первый байт	Адреса хостов	Маска
A	0xxxxxxx	1–127	10.0.0.1 – 126.255.255.254	255.0.0.0
B	10xxxxxx	128–191	128.0.0.1 – 191.255.255.254	255.255.0.0
C	110xxxxx	192–223	192.0.0.1 – 223.255.255.254	255.255.255.0
D	1110xxxx	224–239	224.0.0.1 – 239.255.255.254	255.255.255.0
E	11110xxx	240–247	240.0.0.1 – 255.255.255.254	255.255.255.0

Со временем выяснилось, что, во-первых, сети, как правило, нужно делить на более мелкие части, а во-вторых, что часть IP-сетей используется неэффективно (т. е. сеть выделена, а большая часть адресов в ней свободна). Поэтому было введено понятие маски – битовой карты, разделяющей адрес на две части. Маски позволили делить сеть на подсети, т. е. выделять более мелкие диапазоны адресов. Такой подход получил название бесклассовой адресации.

Маска применяется всегда, когда нужно определить, относится адрес к заданной сети или нет. Как мы знаем, именно эта операция необходима для определения способа доставки пакета узлу – напрямую, или через другую сеть.

Например, если в сети нашего класса компьютер имеет IP-адрес 192.168.5.10 с маской 255.255.255.0, то компьютер 192.168.5.15 в одной с ним сети, а вот 192.168.6.2 – нет. Определяется это наложением битовой маски: биты адреса, на позиции которых в маске стоит 1, относятся к номеру сети, а 0 – к номеру узла. По правилам все 1 в маске обязательно в начале, а 0 – в конце, перемешивать их нельзя.

Таблица 6

192								168								5								10							
1	1	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	1		1	0	0	0	0	1	0	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

По этой схеме после наложения маски номером сети будет 192.168.5.0, а номером узла в этой сети – 10.

Очевидно, что если вы неверно зададите маску, то и «свои и чужие» будут определяться неверно.

Несмотря на то что маски – основной способ разделения диапазонов адресов на сети, по соображениям обратной совместимости и поддержки стандартов правила деления пространства IP-адресов на сети по классам сохраняются. Например, адреса 224.0.0.0 будут именно широковещательными адресами.

Текущие параметры IP на вашем компьютере можно выяснить разными способами (практическое упражнение для урока). Например, в ОС Windows можно открыть сведения о сетевом подключении.

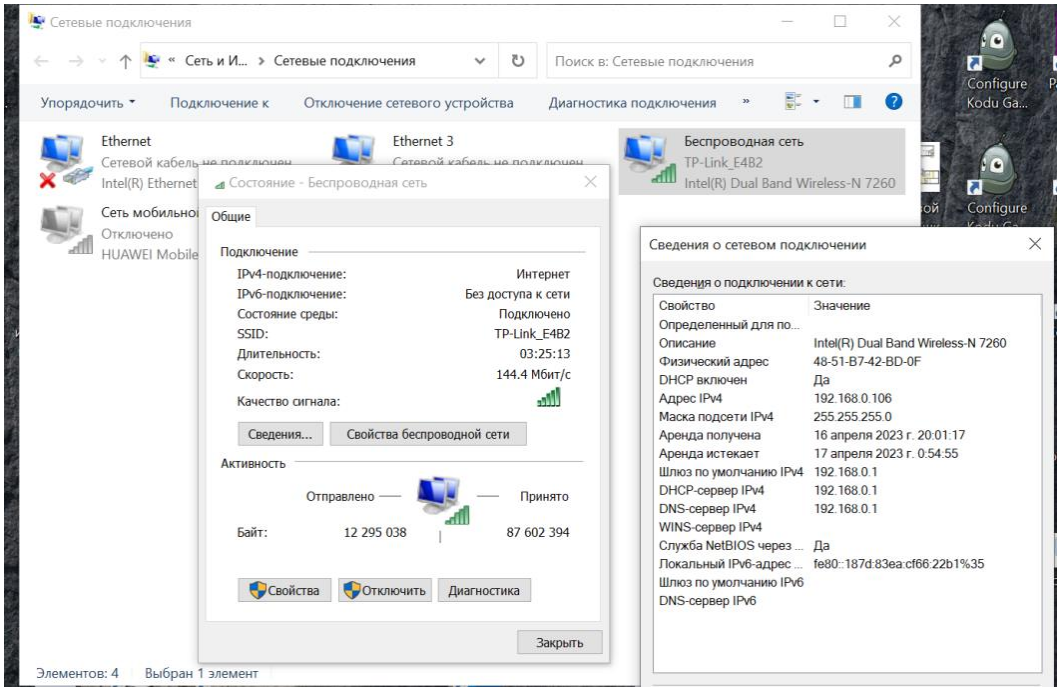


Рис. 2. Сведения о сетевом подключении в ОС Windows

Или в командной строке выполнить команду ipconfig.

```
   DNS-суффикс подключения . . . . . :

Адаптер беспроводной локальной сети Беспроводная сеть:

   DNS-суффикс подключения . . . . . :
   Локальный IPv6-адрес канала . . . : fe80::187d:83ea:cf66:22b1%35
   IPv4-адрес. . . . . . . . . . . . : 192.168.0.106
   Маска подсети . . . . . . . . . . : 255.255.255.0
   Основной шлюз. . . . . . . . . : 192.168.0.1
```

Рис. 3. Результат выполнения команды ipconfig

Как видите, «свои» непосредственно доступные адреса будут относиться к сети 192.168.0.0, а все остальные будут «чужими», т. е. доступными через маршрутизацию.

Маршрутизация

Чтобы связать сети между собой, стеком предусмотрено существование узлов, работающих одновременно как минимум с двумя IP-сетями, т. е. имеющими адреса из разных диапазонов (по правилам, и принадлежащими двум разным адаптерам).

Такой узел можно настроить так, чтобы он пересылал данные из одной сети в другую. Процедура пересылки называется ***маршрутизацией***, т. е. перенаправлением данных по некоторому маршруту.

Для определения, когда какие данные куда пересылать, у каждого узла есть таблица маршрутизации (рис. 4), т. е. таблица, в которой с помощью адресов сетей и масок указывается, какому узлу-маршрутизатору для какой сети пересылать пакеты. У маршрутизатора в такой таблице может быть перечислено несколько доступных сетей. Чтобы не перечислять все возможные маршруты, вводят понятие «шлюз по умолчанию» – и все пакеты, которые не посылаются узлам своей сети (доступным напрямую) и не перечислены в таблице, посылаются для доставки на адрес этого маршрутизатора. В случае небольшой локальной сети – это адрес шлюза провайдера.

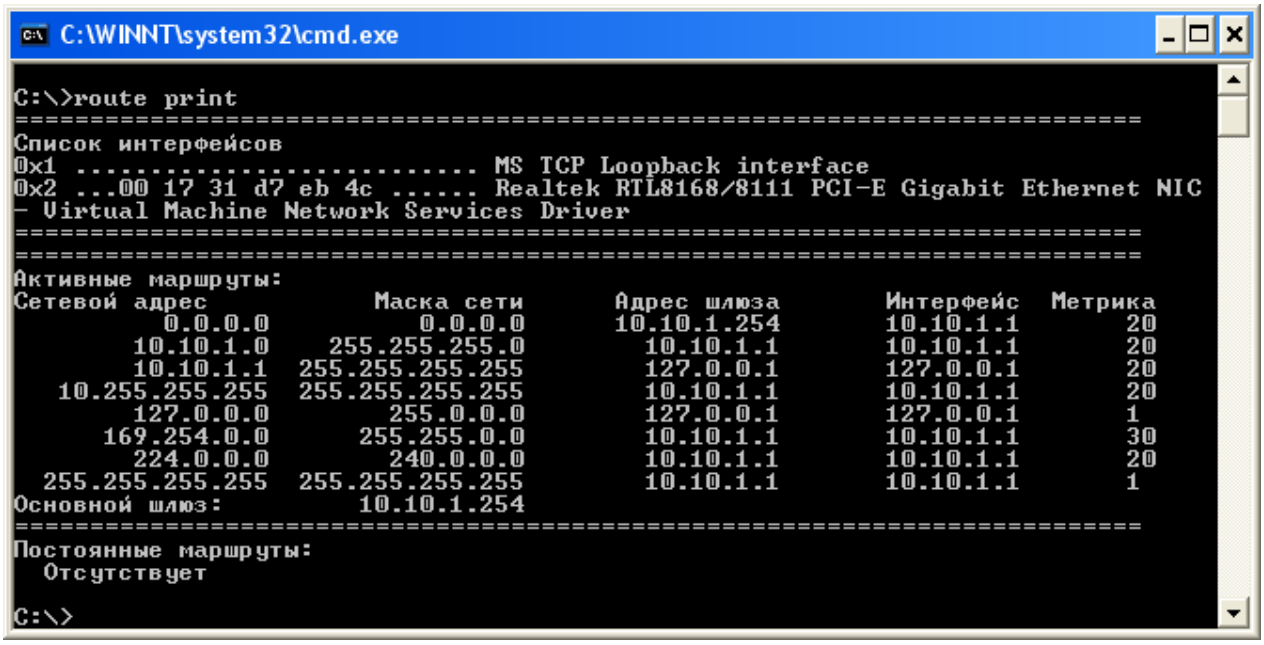


Рис. 4. Таблица маршрутизации внутреннего узла локальной сети

Еще одна проблема, проявившаяся с широким распространением доступа к объединенной среде, – это нехватка IP-адресов в самых востребованных диапазонах (класса B). Для решения этой проблемы были выделены «локальные» (link-local) диапазоны адресов, которые никогда не используются в общей среде напрямую, а предназначены для локальных сетей, в которых большинству узлов нет необходимости получать какие-то входящие обращения.

Тогда все обращения во вне из такой сети проходят через маршрутизатор, который пересылает их дальше уже от своего адреса, регистрирует запрос и потом отсылает на внутренний адрес полученный ответ[1]. Таким образом, у узлов «локальной» сети есть возможность получать данные из внешней сети, но у внешних узлов нет возможности отправить что-то внутренним узлам без запроса с их стороны.

Использование такого механизма позволяет существенно снизить потребность в реальных, свободно маршрутизируемых IP-адресах. Как правило, предоставление такого адреса в сети провайдера – отдельная платная услуга.

Во всех случаях маршрутизация пакетов ведется по принципу «от источника». Это означает, что пакеты (ответы на то или иное обращение) вполне могут пойти не по тому маршруту, по которому был получен запрос, хотя начальные и конечные точки скорее всего совпадут (рис. 5).

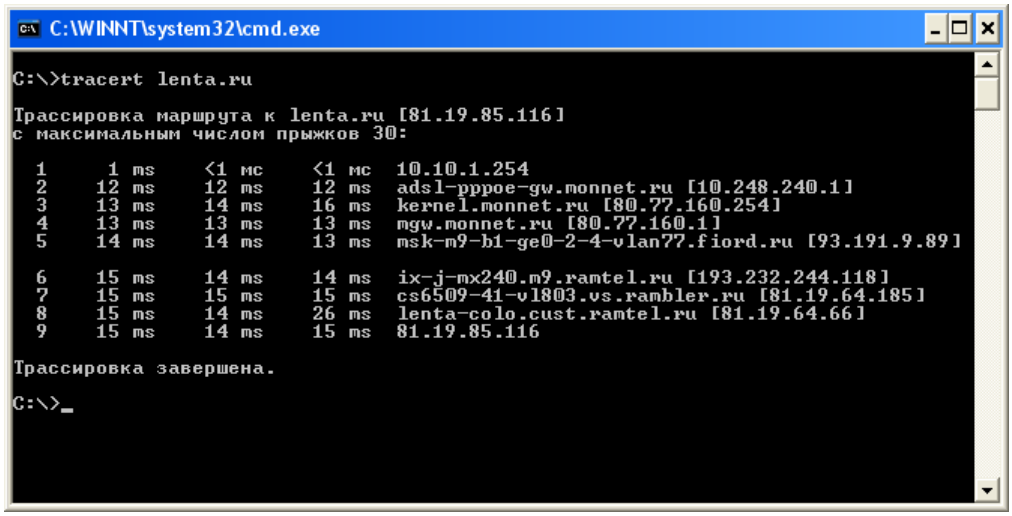


Рис. 5. Путь к популярному ресурсу

Для предотвращения «замусоривания» сети в результате ошибочных (кольцевых) маршрутов для пакетов в IP-протоколе предусмотрено «время

[1]Механизм называется NAT (Network Address Translation – преобразование сетевых адресов), его наиболее частая форма иногда обозначается как «маскарад».

жизни» – количество секунд, которые он может передаваться. При этом считается, что каждый маршрутизатор при пересылке снимает 1 секунду, поэтому фактически время жизни – это количество пересылок. На каждой пересылке время уменьшается, и, как только оно становится равным 0, пакет больше не передается, а его отправителю возвращается служебное сообщение о том, что пакет не удалось доставить.

В крупных сетях, в частности в сетях крупных провайдеров, используют специализированные алгоритмы маршрутизации, позволяющие выровнять нагрузку на каналы[1] и снизить общую стоимость передачи. Существующие протоколы маршрутизации формируют и изменяют общую таблицу динамически. Поскольку проходящие через магистральные маршрутизаторы потоки данных очень велики, то и такие устройства представляют собой крупные высокопроизводительные специализированные компьютеры.

Стоит отметить, что никакая проверка данных обычными маршрутизаторами не производится – слишком велики были бы вычислительные нагрузки и крайне сложно было бы разрабатывать новые протоколы обмена данными.

Несмотря на принятые меры, все равно ощущается недостаток адресного пространства. Кроме того, с ростом потока данных протокол IP версии 4 становится неэффективным – часть служебных данных не используется, процедуру маршрутизации приходится выполнять для каждого пакета в отдельности и т. д.

Поэтому разработан и постепенно внедряется новый основной протокол сетевого уровня – IPv6. Этот протокол предусматривает адрес размером в 128 бит, позволяет вкладывать IP-пакеты друг в друга, имеет значительно более эффективную систему маршрутизации, основанную на метках потоков.

Переход на новую версию выполняется постепенно, поэтому резкого изменения всех программ и замены оборудования не требуется.

Вопросы и задания для закрепления изученного материала:

1. Может ли существовать IP-адрес 228.12.22.31?

[1] Нельзя просто направить данные по кратчайшему маршруту — он очень быстро окажется перегруженным.

2. К какому классу относится IP-сеть 221.12.24.31?

3. Можно ли обратиться к IP-адресу 192.168.16.2, если компьютер находится в сети 192.168.22.0?

4. Может ли быть маршрутизатор с одним IP-адресом?

5. Потребует ли ввод протокола IPv6 перехода на новые каналы связи?

Ответы на задания

1. Да. Но никакому конкретному компьютеру он не может принадлежать – это адрес широковещательной рассылки.

2. Класс С. Но значение это имеет только с точки зрения соблюдения правил.

3. Да, если есть маршрут.

4. Только пока он не настроен и не работает как маршрутизатор. Если у устройства всего 1 адрес, оно не может пересылать пакеты между сетями.

5. Нет.

Транспортный уровень или использование протоколов TCP и UDP

Содержание темы урока

Как следует из определения, сетевой уровень управляет потоком данных между двумя узлами. Но данные мы направляем не просто от одного компьютера к другому, а от программы к программе. Для организации взаимодействия между конкретными программами (т. е. организации транспорта данных) и предусмотрен *транспортный уровень*.

Основными протоколами этого уровня в стеке TCP/IP являются протоколы передачи с контролем (Transmission Control Protocol, TCP) и датаграмм пользователя (User Datagramm Protocol, UDP – передача без подтверждения).

Для идентификации приложений оба эти протокола вводят общее понятие «порт». Речь идет не о физической точке подключения, а о логическом номере для идентификации приложения. Номер используемого порта будет зависеть от вида приложения, которые бывают двух видов:

1) приложение-сервер – запускается, занимает некоторый порт и ждет обращений к нему, получив обращение, обработает его и отправляет ответ;

2) приложение-клиент – занимает порт по мере необходимости, для обращения получает свободный порт, обращается к некоторому приложению-серверу и, получив ответ, закрывает порт.

Порт (номер) приложения-сервера должен быть известен клиентам. Не будем забывать, что клиент и сервер могут работать на разных узлах (и даже в разных сетях), а это значит, что порты сервера должны быть известны клиенту заранее.

Все номера портов делятся на три группы[1] (табл. 7).

Таблица 7

Well-known ports	0–1023	Порты, зарезервированные за серверами определенного назначения
Registered ports	1024–49151	Порты, предназначенные для серверов, но не имеющие обязательной привязки
Dynamic, private, ephemeral ports	49152–65535	Порты для приложений-клиентов, выделяемые временно по запросу

Первая группа – это так называемые «хорошо известные порты», т. е. номера, закрепленные за серверами, отвечающими на запросы по определенным протоколам (табл. 8).

Таблица 8

Порт(ы)	**Протокол**	**Назначение**
25	SMTP	Протокол передачи почты
53	DNS	Протокол разрешения имен
80	HTTP	Протокол передачи содержимого web-страниц
110	POP3	Протокол получения почты
21, 20	FTP	Протокол обмена файлами
443	HTTPS	Передача web-страниц защищенным (зашифрованным) способом

Вторая группа – регистрируемые порты, предназначенные для вновь разрабатываемых или перенастраиваемых серверов. Порты регистрируются (т. е. занимаются и начинают прослушиваться) приложением при запуске. Часто это различные серверы баз данных, специализированные системы обмена (например, видеоконференции) и т. п.

[1] Подробнее можно посмотреть по адресу http://en.wikipedia.org/wiki/Well_known_ports или обратиться к стандартуIANA (http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml).

Третья группа – порты динамические, т. е. выделяемые по запросу любому приложению временно. Этот диапазон используется для обращения клиентов: клиент занимает порт, обращается к серверу (указав, от какого порта и к какому), проводит сеанс обмена и закрывает порт.

TCP

Протокол TCP (протокол контролируемой передачи) – основной протокол обмена данными между приложениями в стеке TCP/IP. Он предназначен для передачи данных, которые должны быть переданы в строго определенном порядке и прием которых должен быть подтвержден. При необходимости он может повторить передачу данных, получение которых за указанное время не подтвердилось.

Процедура обмена данными в этом протоколе начинается с согласования номеров. Клиент обращается к серверу, выбрав случайным образом начальный номер[1] для передачи данных от клиента к серверу. Сервер подтверждает готовность, присылая свой начальный номер для передачи данных от сервера клиенту. Клиент подтверждает получение и только после этого начинается обмен данными.

При обмене и клиент, и сервер добавляют к начальным номерам количество переданных данных и отсылают с каждым пакетом подтверждение – сколько данных передано и получено. Эти номера позволяют выстроить все полученные пакеты в верную последовательность и определить, все ли переданные данные получены.

Если получен пакет данных, перед которым пока нет необходимых частей – он временно «складывается» в буфер. При получении недостающей части весь буфер считается принятым сразу – поэтому кратковременно скорость передачи может быть даже выше предельной пропускной способности канала.

Состояние данных внутри пакета проверяется контрольной суммой. Пакет считается доставленным, если сумма сходится с контрольной, в противном случае данные считаются испорченными.

[1] Конечно, удобнее бы было просто начинать с 0, но в этом случае злоумышленнику было бы очень легко предсказать номера пакетов для перехвата и подмены.

Такая процедура с точным отслеживанием полученных и отправленных данных позволяет гарантировать, что если данные будут получены, то будут получены в правильном виде, без потерь и переставленных кусков. Таким образом передают почту, иллюстрации, страницы, файлы и т. д.

UDP

При всех достоинствах протокола TCP, он требует довольно значительных затрат и при подготовке и отправке данных (например, буфер обмена), и при передаче – как минимум три пакета передаются полностью без реальных данных, только для согласования соединения.

Кроме того, TCP требует, чтобы получатель подтвердил получение, а отправитель за этим следил.

В тех случаях, когда вся процедура обмена предусматривает малое количество данных или когда получение невозможно подтвердить из-за очень большого количества участников (например, массовая рассылка – IP-телевидение), применяют протокол датаграмм пользователя – UDP.

Протокол UDP не предусматривает в своих датаграммах никаких указаний на номер в общей последовательности, не имеет процедуры согласования и фактически только передает фрагмент данных от одной программы к другой.

Целостность данных контролируется с помощью контрольной суммы аналогично протоколу TCP, но никакой повторной передачи не предусмотрено.

Протокол UDP значительно проще и требует гораздо меньше ресурсов, но порядок и доставку никак не гарантирует. Разницу между ним и протоколом TCP можно описать аналогией: TCP – это заказное письмо с уведомлением о доставке, а UDP – почтовая открытка.

Протокол UDP используется в системе DNS, в системах IP-телевидения, в игровых системах, при файловом обмене.

Стоит обратить внимание, что номера портов у TCP и UDP имеют формально два разных диапазона, но в реальности стараются поддерживать соответствие (например, порт № 53 – служба DNS).

Вопросы для закрепления изученного материала

1. При проверке полученного TCP-пакета обнаружено расхождение контрольной суммы с полученными данными. Как вы считаете, это означает, что поврежден только заголовок или, возможно, и сами данные?

2. Одна программа, как правило, занимает во время работы два TCP-порта. Может ли на одной машине работать более 40 000 сетевых приложений (при условии, что аппаратных ресурсов достаточно)?

3. Обеспечивает ли протокол TCP гарантированную скорость доставки данных?

4. Могут ли на одной машине работать приложение, занимающее порт TCP 80, и приложение, занимающее порт UDP 80?

5. Какой протокол транспортного уровня лучше подходит для программы обмена короткими сообщениями в локальной сети?

Ответы на вопросы

1. Возможно, повреждены и сами данные – контрольная сумма рассчитывается по всем словам пакета, включая заголовок и данные.

2. Только при наличии нескольких IP-адресов и функций операционной системы, позволяющей распределять такие ресурсы. На одном IP-адресе это невозможно.

3. Нет. Протокол предусматривает обработку задержек и повторную пересылку – нельзя при таких условиях гарантировать скорость.

4. Да, могут.

5. UDP – сообщения маленькие, вероятность потери очень низкая. Проверять доставку (если это необходимо) можно протоколом прикладного уровня.

Прикладной уровень или сервисы сети

Содержание темы урока

На прикладном уровне используется самое большое количество разнообразных приложений и протоколов. Именно к этому уровню относятся программы, с которыми непосредственно работает пользователь, и именно на этом уровне программы оперируют осмысленными частями данных – сообщениями.

Основная цель этих приложений – обслуживание непосредственных практических задач пользователей, поэтому рассматривают эти приложения как ***сетевые сервисы*** **(*службы*)** – комплексы программ для решения пользовательских задач, обменивающихся между собой данными по некоторым протоколам.

Существует два основных подхода к организации сетевого сервиса, которые называются архитектурой сетевых приложений:

1) *клиент-серверная архитектура*, когда программы делятся на клиентские (непосредственно работающие с пользователями) и серверные (обрабатывающие запросы от клиентов, но не имеющие своего пользовательского интерфейса). Как правило, серверное приложение хранит и накапливает большую часть данных, а клиенты получают только результат их обработки;

2) *распределенная архитектура*, когда участники делят выполнение задачи на части, и каждый выполняет свою часть работы, обмениваясь информацией с остальными. Такой частью могут быть просто данные (распределенные базы данных, файлообменные системы и т. д.), различные вычисления (например, поиск ключей для оценки стойкости шифров, поиск способов укладки белков и т. д.), передача данных аудио- и видео-обмена (сервис Skype) и т. д.

Стоит обратить внимание, что и клиент, и сервер – программы. Часто сервером называют компьютер, предназначенный для работы приложений-серверов, и это путает людей – трудно, например, представить, что на обычной рабочей станции можно запустить сервер.

На основе этого подхода 24 октября 1995 г. Федеральным советом по сетям[1] было принято определение ***единой сетевой среды Интернет***:

«Федеральный совет по сетям утверждает нижеуказанное высказывание, как такое, которое выражает определение термина «Интернет».

«Интернет» определяет глобальную информационную систему, которая:

(I) является логически взаимосвязанной единым адресным пространством, которое базируется на интернет-протоколе (IP) или его будущих расширениях / следующих версиях;

(II) способна поддерживать коммуникации путем использования семейства протоколов: протокола управления передачей/интернет-протокола (TCP/IP) или его (семейства) будущих расширений/следующих версий, и/или других IP-совместимых протоколов;

[1]FNC (Federal Networking Council – Федеральный совет по сетям) – организация, ответственная за удовлетворение сетевых потребностей федеральных агентств США.

(III) предоставляет, использует или делает доступными, публично или персонально, высокоуровневые сервисы, которые наложены на описанные».

Этим определением мы и будем пользоваться в дальнейшем.

Мы рассмотрим несколько сетевых служб, без использования которых работа с сетью сейчас фактически невозможна.

Domain Name System (DNS)

Как было сказано ранее, передача данных между узлами в сети ведется по протоколу IP, в котором для поиска и идентификации компьютеров используются числовые адреса (из 4 или 32 байт). Но людям, которые пользуются сетями, опираться в работе на такие адреса затруднительно, поскольку их очень много, числа никак не отражают доступные на этих узлах ресурсы, а это значит, что их трудно запоминать.

Изначально администраторы просто вели список узлов[1], в котором перечислялись имена и IP-адреса, соответствующие этим узлам.

Со временем список стал чрезмерно большим, обновлять его стало трудно (как, например, отразить изменения, сделанные одновременно в 3–4 местах?), и появилось несколько задач, которые таким списком не решались, например: как найти узел не по имени, а по требуемой услуге; как определить, кто может принять письма для почтового отделения, если к нему относится несколько узлов.

Для решения этих задач была создана специальная служба доменного именования – Domain Name System (DNS).

Основой работы этой службы является пространство имен, т. е. набор всех возможных имен. Имена объединяются в именованные логические группы – ***домены***. Имя домена может включать в себя от 2 до 57 больших и малых латинских букв[2], цифры и дефис. Имя домена не может начинаться или заканчиваться дефисом и содержать два дефиса подряд.

Считается, что все возможные имена входят в *корневой домен*, который обозначается «.». Для удобства этот домен делят. Первый уровень деления (домены первого уровня) выделяется по двум основным критериям:

[1] Файл hosts – общепринятое название, применяется до сих пор.

[2] Доменные имена с использованием символов национальных алфавитов преобразуются в такие имена по специальным правилам.

по принадлежности к стране[1] или по роду деятельности. В именах доменов первого уровня может быть от 2 до 6 символов.

Внутри каждого такого домена, в свою очередь, выделяют более мелкие группы – домены второго уровня. Чаще всего это сокращенные названия организаций или торговых марок. Внутри них тоже можно выделить группы – третьего уровня и т. д. На практике, деление больше чем в три шага делается чрезвычайно редко.

Каждый домен, как было указано в определении, состоит из имен. Узлы, которые необходимо находить, получают имена чаще всего исходя из названий запущенных на них сервисов, например – www.

С технической точки зрения, основная задача, которую решает служба DNS, – это *разрешение имен*, т. е. поиск IP-адреса по имени (и, возможно, службе). Основа решения этой задачи – распределенная база данных. База данных состоит из записей, в которых указывается имя, тип[2] и данные (например, IP-адрес). База данных поделена на *зоны* – участки пространства имен, собранные в единой части с общим управлением. Зона может содержать ссылки на другие зоны, например: зона ru фактически состоит из ссылок на зоны доменов второго уровня.

Обеспечивают работу зон и выполнение операции разрешения имен DNS-серверы. Чтобы обеспечить устойчивую работу каждой зоны, один из них считается главным (первичным) и при необходимости может существовать несколько вторичных серверов для распределения нагрузки и обеспечения работы при отказе или временной недоступности первичного[3]. Сервер может и не иметь собственной зоны, а только выполнять разрешение имен.

Каждый клиент, использующий службу DNS, должен иметь данные о «своем» DNS-сервере, т. е. его IP-адрес. Клиент направляет серверу запрос на разрешение имени. Сервер ищет имя в своей зоне (если обращение выполнено к ней) или в накопленном буфере. Если записи нет, то сервер обращается на уровень выше, т. е. к серверу провайдера или, выступая как клиент, ищет сервер нужной зоны. Он может обратиться к серверу зоны второго уровня или

[1] По классификатору ООН.

[2] Например: A – адрес, MX – почтовый сервер, SRV – доступный сервис, NS – сервер имен и др.

[3] Для домена в зоне ru существование минимум одного вторичного сервера в другой IP-сети – обязательное техническое требование.

первого и получить от них ссылку на нужный сервер имен или сообщение о том, что такого домена не существует. Если имя найдено, то его сообщают клиенту.

Корневую зону (домен «.») поддерживают 12 независимых операторов[1], распределенных по всему миру. Таким образом обеспечивается распределение нагрузки и надежность работы всей системы. Имена доменов первого уровня регистрируются международной организацией ICANN, остальных доменов – регистраторами[2] или владельцами зон.

Служба DNS позволяет:

– регистрировать имена в разных частях базы данных независимо от остальных;

– давать несколько имен одному узлу, например, для поддержки нескольких сайтов;

– регистрировать несколько IP-адресов для одного имени – для распределения нагрузки;

– находить адрес по функции, например почтовый сервер заданного домена.

Работать без службы DNS в объединенной среде Интернет можно, но чрезвычайно затруднительно. Целый ряд служб без ее участия совсем не могут работать.

Электронная почта

Сервис электронной почты – самый старый из существующих сетевых сервисов. Именно он стал самым первым настоящим сетевым сервисом, возникшим практически сразу после появления сетей.

Изначально сервис был построен на уже существовавшей программе – системе обмена сообщениями в терминальном комплексе. Для того чтобы сотрудники разных смен вычислительного центра могли обмениваться сообщениями, был создан общий каталог, в котором каждый сотрудник получил свой подкаталог (почтовый ящик). Для размещения и чтения сообщений была создана программа, которая позволяла просмотреть сообщения в своем разделе и положить сообщение в виде файла в чей-то раздел.

[1] http://root-servers.org/

[2] Для домена ru главным регистратором является ru-center, http://www.nic.ru.

С появлением сетей была разработана дополнительная программа – агент доставки почты (MTA, Mai lTransfer Agent), которая позволяла не просто положить сообщения в папку, а отправить их на удаленную машину с аналогичной системой почтовых ящиков.

Единицей обмена данными в такой системе стали (и остаются до сих пор) письма, а адрес их доставки состоит из двух частей: адреса почтового ящика и названия почтового отделения.

Со временем появилось множество различных протоколов обмена почтой. Основным из них является один из протоколов, вошедших в стек TCP/IP, – SMTP (Simple Mail Transfer Protocol – простой протокол передачи почты).

Этот протокол применяется для отправки почтовых сообщений, в качестве транспортного протокола используется протокол TCP, сервер для приема почты по этому протоколу имеет стандартный номер – 25.

Почтовое сообщение в рамках этого протокола представляет собой текст (поскольку протокол разрабатывался для ситуации, когда им мог воспользоваться человек без почтовой программы). Текст состоит из двух частей – заголовка, описывающего сообщения набором полей (от кого, кому, когда, на какую тему и т. п.), и тела письма.

Тело письма тоже представляло собой текст, что требовало перед отправкой бинарных файлов провести их специальную обработку – с помощью специальной кодировки превратить их в текст.

Со временем, для того чтобы можно было отправлять в письмах вложения, а также использовать в письмах не только текст на английском языке, был разработан специальный стандарт, позволяющий составить сообщение из нескольких частей: Multipurpose Internet Mail Extensions (MIME, многоцелевые расширения Интернет-почты). Этот стандарт позволяет составить письмо из нескольких разнородных частей, описав их названия, тип и способ представления. Стандарт оказался удобным и сейчас применяется и в других системах.

Отправка почтового сообщения по протоколу SMTP выполняется агентом доставки, который запрашивает службу DNS о сервере почтового отделения, после чего пробует связаться с MTA этого отделения и передать сообщение.

Протокол не требует проводить какую-либо проверку при отправке и приеме, поэтому формально в письме можно написать что угодно, в частности

в заголовке письма. Современные почтовые отделения просят пользователя представиться только при отправке через них писем, но не при получении письма.

Протокол SMTP позволяет отправить письмо, но не предоставляет возможностей просмотреть содержимое своего почтового ящика, не имея непосредственного доступа к каталогу с почтой. Поэтому в паре с протоколом SMTP применяется как минимум один из протоколов доступа к почтовому ящику. Чаще всего это протокол POP3 (Post Office Protocol 3 – протокол почтового отделения версии 3: протокол TCP, порт 110) или IMAP 4 (Internet Mail Access Protocol – протокол доступа к интернет-почте версии 4: протокол TCP, порт 143 или 993), которые позволяют просмотреть содержимое своего почтового ящика.

Существенной проблемой в работе системы электронной почты является нежелательная электронная почта разного (в основном, рекламного) характера, которая получила название SPAM. Возможность сравнительно дешево разослать сообщение по десяткам миллионов адресов привела к тому, что, несмотря на явное противодействие и недовольство пользователей и администраторов, даже малой доли «успешных» сообщений такого рода достаточно, чтобы окупить рассылку.

Для борьбы с такими сообщениями применяют различные фильтры[1]. Это существенно замедляет обработку сообщений, приводит к лишнему трафику и иногда – потерям. К сожалению, в настоящий момент по разным оценкам примерно половина всех пересылаемых сообщений – спам.

Рассылка сообщений часто ведется с подставных компьютеров, которые злоумышленники захватывают с помощью программ – «сетевых червей», что затрудняет их поимку и доказательство факта рассылки. Большинство провайдеров по этой причине очень подозрительно относятся к исходящему почтовому трафику из своих сетей.

Несмотря на многочисленные недостатки, электронная почта – основной сервис для обмена индивидуальными сообщениями. Адрес электронной почты часто играет роль личного адреса-идентификатора при работе в среде Интернет. Указывая личный адрес на различных ресурсах, необходимо понимать, что он может быть использован для сбора и поиска всей информации о вас лично.

[1] Некоторые алгоритмы упоминаются в разделе интеллектуальной обработки текста.

Служба World Wide Web

Служба Всемирной паутины была разработана в CERN (Европейский совет по ядерным исследованиям) для оперативной публикации статей, посвященных научным исследованиям.

Ключевым в работе этой службы является способ представления информации в виде фрагментов, связанных между собой направлениями перехода.

Такое представление существовало задолго до его применения в сетях и носит название *гипертекстовая парадигма* представления информации. Гипертекстом такую структуру называют потому, что она позволяет создать документ, который можно читать не только линейно от начала к концу, но и по своей собственной траектории.

Техническую основу службы составляют серверы, которые по запросу выдают фрагменты, – **web-серверы** и программы просмотра – **web-браузеры**.

Основным видом фрагментов в WWW-службе являются страницы – текстовые модули, содержащие указания об отображении. Указания о способе отображения страниц делаются на специальном языке разметки – Hyper Text Markup Language.

Логически единые совокупности страниц называют **сайтами**. Чаще всего сайт объединяет страницы общим именем и способом оформления.

Стоит особо заметить, что сервер выдает фрагмент, но не обязательно этот фрагмент должен храниться в виде файла. Часто запрос – это только ссылка на программу, которая запускается и готовит этот фрагмент, например, на основе данных из какой-либо базы данных.

Такая возможность позволяет использовать службу WWW как интерфейс для доступа к самым различным данным – главное, чтобы логически все сводилось к набору страниц, отображаемых на экране.

Для реализации того или иного функционала с помощью этих возможностей создают специализированные средства генерации web-страниц, получившие название *web-приложение*. Web-приложение состоит из двух основных частей – серверной (т. е. приложений, работающих на стороне сервера) и клиентской, работающей в браузере пользователя.

В обоих случаях обработкой данных чаще всего занимаются не программы в бинарном коде, ориентированные на исполнение процессором без посредников, а наборы указаний интерпретатору – *сценарии*.

Особенность этой службы – возможность ссылаться из фрагмента на любые другие объекты, в том числе расположенные на других узлах и других сайтах. Для того чтобы обеспечить такие ссылки, был разработан и введен специальный стандарт записи адресов ресурсов.

Универсальный адрес ресурса – URL (Universal Resource Locator)

Универсальные адреса ресурсов – одна из базовых составляющих службы WWW, которая часто используется и за ее пределами. Их назначение – указать способ поиска и доступа к любым ресурсам, существующим в сети.

По действующим правилам, части URL-адреса составляются из больших и малых букв латинского алфавита, цифр и специальных символов за исключением пробела, прямого слэша («/»), двоеточия и @ – эти знаки используются как специальные.

Общая форма этих адресов такова:

схема:[//] [имя[:пароль]@]адрес_узла[/имя_ресурса]

Схема – это способ обращения к ресурсу. Чаще всего схема – это название протокола, по которому осуществляется доступ, но как мы покажем на примере не всегда.

Имя и (возможно) пароль – дают возможность передать сразу при обращении к серверу необходимые данные о пользователе. По понятным причинам передавать таким способом пароль не рекомендуется.

Адрес узла – это адрес, по которому можно обратиться к узлу, на котором находится ресурс. Адрес может быть как адресом в службе DNS, так и прямым IP-адресом. Необходимо только помнить, что часто это не взаимозаменяемые адреса (чаще всего с одним IP-адресом сервера связано множество имен DNS).

Имя ресурса – это идентификатор ресурса на конкретном узле, обрабатываемый сервером. Если служба имеет внутреннюю древовидную структуру, то может указываться путь к ресурсу. Путь может не соответствовать файловой структуре. Если в имени ресурса встречаются какие-то

непредусмотренные стандартом символы, то они должны быть закодированы – каждый такой символ записывается знаком «%» и после него – шестнадцатеричный код символа.

И адрес узла, и имя ресурса могут отсутствовать в URL-адресе – в этом случае сервер подставляет значение по умолчанию, например, первую страницу сайта. В таблице 9 представлено несколько примеров адресов

Таблица 9

http:	//en.wikipedia.org	/wiki	Book:Adriatic_campaign_of_1807%E2%80%931814
схема	адрес узла	путь	имя ресурса

mailto:	info@	yandex.ru
схема	пользователь	адрес узла – в данном случае, почтовое отделение

ftp:	//tor:Passw0rd@	ftp.archive.org	/pub/doc/book/Inform10-11.pdf
схема	имя и пароль	адрес узла	имя ресурса

Наличие такого универсального способа записи адресов позволяет связывать разнородные информационные фрагменты, находящиеся в разных местах, и создавать своеобразную «всемирную паутину».

Представление страниц

Как мы уже указали, основная (создающая структуру) часть данных в службе WWW представлена в виде страниц. Страницы для браузера описываются как текст со специальными пометками о его отображении, т. е. в виде *размеченного текста*.

Язык разметки текста, как вы знаете из курса информатики основной школы, называется *Hyper Text Markup Language*. Возможность связывать страницы в гипертекст, как видно, отражена даже в названии языка.

Пометки об обработке страниц называют *тегами*. При записи теги отмечаются угловыми скобками, каждый тег после открывающей скобки имеет мнемоническое обозначение из нескольких букв.

Текущая версия языка разметки страниц – HTML 4.1. Это большой комплекс документов, описывающих не только собственно теги и их атрибуты, но и основные правила создания содержимого для web-страниц, способы их анализа, правила описания оформления и т. д.

В новой версии языка HTML особое внимание уделяется способам вставки видео и звука на страницы.

С развитием языка и увеличением областей его применения выявилось много технических сложностей с анализом получившихся документов (например, поисковой системе трудно верно определить, что в тексте данные, а что является просто оформлением) и необходимостью формировать разные представления для одних и тех же данных.

Для решения этих задач был разработан специальный комплекс языков для представления данных и преобразования этого представления в HTML-страницу. Этот комплекс получил название XML (eXtensible Markup Language – расширяемый язык разметки) и сейчас очень активно применяется для организации взаимодействия приложений и их частей. Во многих современных приложениях HTML-представление появляется только на самом последнем этапе – перед показом пользователю.

Рассмотренные нами технологии и протоколы позволяют построить единую информационную среду, решающую самые разные задачи пользователей, растущую и адаптирующуюся к самым разным условиям. Именно открытость и универсальность остаются основными причинами проникновения сети в самые разные области деятельности людей.

Вопросы для закрепления изученного материала

1. Нужно ли протоколу SMTP проверять корректность (отсутствие повреждений) полученных сообщений?

2. Может ли сервер обратиться к приложению-клиенту без запроса со стороны клиента?

3. Почему служба DNS использует транспортный протокол UDP?

4. Возможно ли существование такого адреса DNS: www.spacetuning.net?

5. Можно ли выполнять передачу данных в сети Интернет без использования DNS?

6. Протокол SMTP при обработке входящей почты не требует проверки существования адреса отправителя. Предположим, такое требование введено. К чему это приведет в случае сервера, получающего 50–100 писем в секунду?

7. Будет ли при отключении службы DNS нормально функционировать сервер компании, предоставляющей доступ к 50 сайтам на один IP-адресе? Почему?

8. В IP-датаграмме контрольная сумма заголовка не совпала с суммой, рассчитанной получателем по словам заголовка. Считается ли датаграмма поврежденной?

Ответы на вопросы:

1. Формально нет – это задача протокола TCP, который он использует для обмена. В реальной ситуации многое зависит от программ-серверов и клиентов, которые могут намеренно игнорировать некоторые виды ошибок.

2. Нет. Для этого со стороны клиента не выделен порт транспортного протокола, и клиент не прослушивает порты в ожидании обращения.

3. Потому что большинство запросов короткие – и либо на них приходит верный ответ, либо их все равно придется повторить.

4. Нет. Пробелы в именах запрещены.

5. Да, возможно.

6. К задержке обработки – для каждого письма придется проводить процедуру проверки, и значительному росту нагрузки. Кроме того, появится возможность вообще парализовать работу сервера, отправив, например, очень много писем с поддельным адресом.

7. Нет. В этом случае сайты будет очень трудно различить – если нет имен, то пользователю придется писать не только IP-адрес, но еще и порт, причем нестандартный.

8. Да, считается. В этом случае нельзя гарантировать верное определение длины данных и их дальнейшую обработку.

Рассмотрим далее тему **«Информационная безопасность»**, на которую выделено 7 часов учебного времени и предусмотрено две практические работы. Разделим тему на ряд подтем в соответствии с планированием. Учтем, что в тему «Информационная безопасность» входят и вопросы шифрования данных, которые ранее в курс информатики не входили, но часто изучались за счет дополнительных часов внеурочной деятельности, например при подготовке к олимпиадам. При изучении компьютерных сетей была использована уровневая модель (DOD), поэтому и защиту данных в сетях также следует рассматривать по описанным уровням.

Защита данных в сетях

Содержание темы урока

Вспомним еще раз о назначении межсетевой среды (например, Интернет) – это обеспечение взаимодействия. Очевидно, что чем больше наша зависимость от этой среды, чем больше задач мы решаем с ее помощью, тем более чувствительными для нас могут стать нарушения в работе сети, особенно нарушения намеренные.

Поэтому при работе в сети большое значение имеют меры по обеспечению информационной безопасности.

Информационная безопасность – процесс соблюдения (сохранения) трех аспектов (атрибутов) безопасности: ***доступности***, ***целостности*** и ***конфиденциальности*** информации.

Опишем эти аспекты подробнее:

1. **Доступность информации.** Информация в безопасном состоянии должна быть доступна для пользователя, т. е. должна быть сохранена возможность проведения всех операций по ее обработке. Для этого необходимо работающее оборудование, неповрежденные носители и, конечно, наличие необходимых программ, причем правильно настроенных.

2. **Целостность информации** – это соответствие логической структуры информации определенным правилам, т. е. логически корректное ее состояние. Процедуры обработки и изменения информации должны преобразовывать одно целостное состояние в другое.

3. **Конфиденциальность –** выполнение тех или иных операций с информацией должно происходить в соответствии с некоторыми правилами, составляющими существенную часть политики безопасности. Нарушение конфиденциальности – возможность выполнения операций (например, чтения или записи) теми, кто этого не должен делать.

Нужно отметить, что это именно аспекты, т. е. стороны одного и того же процесса. Все они тесно связаны между собой, нарушение одного из них вполне может стать нарушением и другого.

Также нужно помнить, что информационная безопасность – *процесс*, а не конечный результат каких-то мер. Информацию необходимо обрабатывать, получать и передавать – и во всех этих процессах учитывать компоненту обеспечения безопасности.

При построении системы безопасности используют несколько общих понятий:

1. **Угроза** – возможное общее направление нарушения аспектов безопасности. Например, можно рассматривать угрозу утечки, повреждения или потери данных, отказа оборудования и т. д. Если происходит какое-то событие указанного направления, то говорят о *реализации угрозы*.

2. **Уязвимость** – характеристика информационной системы или программно-аппаратного комплекса, которая может привести к реализации угрозы. Уязвимостью может быть особенность сетевой технологии, неверная настройка программ и т. д.

3. **Атака** – намеренная попытка реализации угрозы, например, выразившуюся в поиске уязвимости и т. п.

В основном при обеспечении безопасности уделяют внимание намеренным ее нарушениям. Тем не менее при проектировании системы защиты (даже такой простой, как защита домашней сети) нельзя забывать и о возможных аппаратных проблемах, например отказе жесткого диска.

Не стоит надеяться, что игнорирование проблем обеспечения безопасности (например, с популярным обоснованием «у меня все равно нет ничего важного») защитит вас от возможных атак. С учетом глобального характера современных сетей, большинство атак направлены не на кого-то конкретного, а выполняются по принципу «боя на площади», т. е. автоматические программы ищут уязвимости у всех, кого только могут обнаружить.

Несмотря на немалое количество теоретических сведений и технических средств, которые применяются для обеспечения информационной безопасности, можно выделить несколько общих принципов, необходимость соблюдения которых от технических средств зависит мало:

1. *Применение превентивных мер.* По техническим причинам реализация подавляющего большинства угроз при обработке информации с помощью компьютера происходит значительно быстрее, чем пользователь может распознать атаку и предпринять какие-то меры. По этой причине защита должна быть продумана и реализована до того, как проблема возникнет.

2. *Уменьшение поверхности атаки.* Чем меньше объектов, которые вообще могут быть подвержены тем или иным угрозам, тем меньше вероятность нарушения аспектов безопасности. Из этого принципа напрямую вытекает

необходимость минимизации количества программ и их взаимодействия с внешними источниками информации.

3. *Защита всех этапов обработки информации.* Степень уязвимости системы определяется по наиболее уязвимому узлу. Вне зависимости от общего количества принятых мер, они будут бесполезны, если останутся слабозащищенные этапы обработки.

4. *Эшелонирование защиты.* Все защитные комплексы создаются по принципу эшелонов, т. е. этапов, слоев обработки. Это позволяет отчасти компенсировать недостатки, снизить общую вероятность поражения системы или минимизировать ущерб от успешной реализации угрозы. Тем не менее каждый «эшелон» при построении считается единственным (т. е. все предшествующие считаются уже преодоленными) и делается максимально закрытым (см. принципы 1 и 2).

5. *Разграничение доступа.* Доступ к исполнению тех или иных операций должен соответствовать задачам, стоящим перед конкретным пользователем. Чем меньше таких операций доступно пользователю, тем меньше ущерб, который может быть нанесен (не обязательно самим пользователем, возможно одной из его программ)[1].

6. *Желание быть защищенным.* Самый уязвимый компонент защиты – плохо обученный пользователь. Никакие ухищрения не помогут, если пользователь не соблюдает мер предосторожности и не понимает, какие угрозы возникают во время его работы[2].

Защита на сетевом и транспортном уровне

Меры защиты, предпринимаемые на этих уровнях, объединены в общий раздел по двум основным причинам:

– во-первых, протоколы именно этих двух уровней – наиболее общая часть стека TCP/IP, которая так или иначе используется на всех узлах среды Интернет;

[1] Очень популярное обоснование отключения защитных мер – «они мешают мне работать». В этом случае стоит задуматься либо об организации защиты, либо, что более вероятно, об организации работы – иначе работа может неожиданно прерваться совсем. Нельзя не отметить, что в подобном случае «обосновавшие» отключение защитных мер лица предпочитают переложить ответственность на технический персонал.

[2] Большая часть ущерба наносится не в результате направленных вредоносных действий, а в результате ошибок. В подавляющем большинстве оставшихся случаев вред наносится сотрудником пострадавшей организации из ее внутренней сети.

– а во-вторых, потому, что основные программы, которые позволяют защитить информацию на этих уровнях, – комплексные и используют информацию этих двух уровней в совокупности.

Напомним, что сети-посредники не интересуются ни содержанием, ни тем более целью отправки фрагментов. Задача протоколов сетевого уровня – доставить отправленные данные по назначению, т. е. узлу-получателю.

Шлюз не обязан проверять «обратные» адреса и уж тем более определять верные ли они. Как мы видели в предыдущем разделе, это далеко не всегда так.

Данных об узле отправления и узле получения недостаточно для принятия решения о безопасности данных – как мы знаем, на конкретном узле может работать множество клиентов и серверов. Поэтому при анализе также используются данные транспортного уровня, позволяющие определить приложение и стадию обмена.

Логичным решением, повышающим уровень безопасности, является использование специальных программ-фильтров, определяющих откуда и к каким программам можно обращаться.

Программы, выполняющие фильтрацию данных, называются брандмауэрами (они же – межсетевые экраны, файрволы от англ. *Firewall* – противопожарная стена, огнеупор).

Современные брандмауэры – сложные и многофункциональные комплексы программ, задача которых – обеспечение безопасного взаимодействия сетей. Они содержат несколько компонентов, как правило, это:

- пакетный фильтр – программа, определяющая на основе правил данных сетевого и транспортного уровней пропускать ли пакет в сеть или из сети;
- посредники для протоколов прикладного уровня – proxy. В отличие от пакетного фильтра, сервер-посредник проверяет также данные прикладного уровня, например, URL-адреса. Конечно, такой посредник не универсален, поэтому, как правило, это web-посредник;
- средства организации защищенных каналов – VPN (Virtual Private Network – виртуальная частная сеть);
- модули обнаружения атак.

Межсетевые экраны – необходимый компонент защиты во всех современных сетях. Когда узлы сети независимы и нет возможности определить общую политику доступа (например, в сети провайдера) применяют

персональные межсетевые экраны, защищающие одну машину. В этом случае их часто интегрируют с антивирусными средствами, добавляют средства контроля поведения конкретных приложений и т. д.

Защита на прикладном уровне

Поскольку на прикладном уровне реализовано самое большое количество разнообразных программ и сервисов, то именно на этом уровне сосредоточено наибольшее количество уязвимостей и именно он чаще всего становится объектом атак (или ошибок пользователей).

Основной мерой предупреждения и уменьшения последствий реализации угроз на этом уровне является *разграничение доступа*, т. е. определение минимально необходимого количества возможных действий. Для того чтобы доступ разграничить, необходимо определить критерий разграничения.

По виду этого критерия чаще всего различают два типа систем:

1) системы с регулированием доступа на основе пароля.
2) системы, с регулированием доступа на основе личной учетной записи пользователя.

В первом случае доступ предоставляется тому, кто знает некоторые не всем доступные данные – пароль или ключ[1]. Например, такой способ часто используется при предоставлении доступа к беспроводной сети.

Во втором случае пользователь «представляется» системе и получает те или иные права в зависимости от того, что ему разрешено.

По многим причинам первый способ регулирования доступа используется только в небольших системах с ограниченным и постоянным количеством пользователей и небольшим количеством прав.

При работе с личными учетными записями, первое, что необходимо сделать, – пройти процедуру ***аутентификации***, т. е. установления соответствия между оператором и его учетной записью.

Чаще всего аутентификация выполняется на основе предъявления названия учетной записи (имени для входа в систему, действие LogIn) и личного пароля. Поскольку имя пользователя – информация общеизвестная, то подтверждение личности должно выполняться данными, доступными только пользователю – паролем.

[1] Разница между этими двумя понятиями есть – пароль только предъявляется, а ключ используется во время работы.

В большинстве систем пароль – единственный критерий «подлинности», а поэтому такой способ называют *однофакторной аутентификацией*.

Там, где обрабатываются важные данные, например финансовые, для усиления защиты используют *двухфакторную аутентификацию* – для входа требуется еще и физическое устройство с секретной информацией (в виде карты, ключа или мобильного телефона), а доступ к устройству регулируется персональным идентификационным кодом (PIN).

Основная угроза при работе с системами идентификации – утечка или утеря пароля (PIN-кода, ключа). У двухфакторных систем аутентификации вероятность реализации угрозы утечки намного ниже, поскольку необходимо «потерять» не только PIN, но и сам ключ.

Помимо нетехнических методов реализации этой угрозы, существует минимум одна типовая атака, очень простая по своей сути, – подбор пароля. Пример такой атаки – BruteForce, подбор методом грубой силы. Это прямой перебор всех возможных паролей.

Вероятность «удачи» атаки напрямую зависит от того, сколько вариантов придется перебрать, если в пароле используется ограниченный набор символов, да еще и связанных логическими соотношениями (дата рождения, например, или просто пароль по умолчанию для устройства с заводскими установками), подбор будет быстрым.

Для снижения вероятности утечки или подбора пароля рекомендуется соблюдать несколько несложных правил:

– иметь пароль, не связанный ни с какими личными фактами;

– иметь пароль достаточной длины (в современных условиях – не менее 7 символов);

– использовать в пароле символы из разных наборов: большие и малые буквы, цифры, специальные символы;

– время от времени менять пароль.

Система подтверждения доступа на основе учетных записей и паролей очень популярна, но может «в чистом виде» применяться только в том случае, когда каналы связи защищены от перехвата данных (иначе возможна утечка пароля) и есть возможность создать и вести единый список пользователей.

Это возможно в условиях четко определенной среды, например локальной сети с единым управлением, но в условиях крупных и сложных сетей этого недостаточно. Во-первых, при использовании среды Интернет мы очень часто передаем данные через недоверенные и непроверяемые сети. Во-вторых, возникает вопрос доверия к центру хранения учетных записей.

Как минимум необходимо принять особые меры для защиты базы данных имен и паролей пользователей, потому что ее кража сразу скомпрометирует систему. Поэтому в современных условиях пароли никогда не хранятся в открытом виде, а хранятся результаты их преобразования – хэш-коды. Такой хэш-код нельзя просто перевести назад в пароль, но можно при проверке привести к нему пароль, полученный от пользователя. Таким образом, кража базы данных паролей не приведет к немедленной компрометации системы.

Второй важный аспект борьбы с угрозами на прикладном уровне – это борьба с различными вредоносными программами. По определению, вредоносная программа – это специально разработанное средство, применение которого наносит пользователю ущерб. Это может быть как использование ресурсов пользователя (например, рассылка писем с его почтового ящика), так и явная кража данных (например, платежных), шантаж (например, потерей данных или доступа к ним) и масса других.

Для борьбы с этими угрозами применяется целый ряд средств.

1. **Антивирусные программы.** Название сформировалось в те времена, когда разнообразие вредоносных программ было значительно меньше, поэтому не полностью отражает специфику применения. В настоящий момент это средства, которые выявляют и блокируют (уничтожают) вредоносное ПО, обнаруживая его по характерным участкам (сигнатурам), или по применению характерных способов сокрытия содержимого, или поведению (например, попыткам изменения существенных системных файлов) и др. Антивирусные средства могут контролировать как файлы (при регулярной проверке или при доступе к ним), так и другие виды сообщений, например письма электронной почты.

2. **Системы контроля целостности файлов.** Эти системы создают контрольные суммы для существенных файлов (например, программ и настроек ОС) и регулярно контролируют их изменения – блокируя доступ к измененным файлам при необходимости.

3. **Системы контроля утечек.** Такие системы, как правило, применяют на уровне организации – для предупреждения и выявления попыток отправки конфиденциальной информации за пределы организации.

Проектирование и поддержание в актуальном состоянии системы защиты на предприятии – сложная инженерная задача. Но на уровне отдельного пользователя (или его домашней сети, личных устройств) меры по защите может и должен принять каждый.

Отдельной важной мерой обеспечения информационной безопасности (в части предупреждения потери информации) является формально не относящееся к ней резервное копирование.

Как видно по названию, ***резервное копирование*** – это создание копии информации на случай ее потери: в результате ошибки оператора, отказа оборудования, чьих-то злонамеренных действий (например, в результате работы вируса-шифровальщика).

Несмотря на логическую простоту и очевидность этих мер, о них часто забывают. Во избежание неприятных последствий, стоит предпринять стандартные меры:

1) определить существенную для вас информацию (включая программные средства, необходимые для доступа к ней), место и способ ее хранения, стоимость возможных потерь (включая потери времени, репутации и др.);

2) составить план ее копирования, с учетом п. 1. Например, если для вас приемлемы потери времени на повторную установку и настройку операционной системы – можете ее в план не включать. Если для вас это приведет к потере времени и средств (например, на вызов специалиста), стоит и ее включить в план. План также должен включать в себя регулярные сроки выполнения – для разных видов информации они разные (например, потеря месяца заполнения базы данных – неприемлема, а операционная система и основные настройки в течение месяца, как правило, существенно не меняются);

3) определить средства копирования и место хранения копий. Рекомендуется предусмотреть копирование на отдельный носитель, чтобы не потерять данные в результате отказа носителя. Для существенных данных носитель должен быть отдельным, недоступным для злоумышленника;

4) создать копии и запланировать их создание по возможности в автоматическом режиме;

5) время от времени проводить проверку резервных копий, делать это лучше восстановлением на чистый носитель всей сохраненной информации и проверки ее на пригодность. Это позволит выявить, например, неочевидные для вас зависимости, упущенные изменения и пр.

Обратите внимание – все меры по резервному копированию предпринимаются ЗАРАНЕЕ. До наступления события потери информации.

Самый простой и очевидный способ создания резервной копии – ручное копирование важных для вас файлов в другое расположение (другой носитель). Следует подумать о том, что любой важный для вас файл должен храниться минимум на двух разных носителях в каждый момент времени.

Второй способ – регулярное создание резервных копий по расписанию, с помощью специализированного программного обеспечения. Для обычного пользователя стоит обратить внимание на средства создания полного образа вашей рабочей среды (клона диска).

Как отдельную меру сейчас часто предусматривают создание резервной копии в облачном хранилище. Это позволяет восстановить данные даже в случае полной потери (например, в случае пожара). Тем не менее есть и некоторые особенности такого хранения:

1) защита данных от утечки и кражи недобросовестными сотрудниками оператора, в таком случае рекомендуется применять стойкие методы шифрования;

2) учет пропускной способности каналов связи, стоимости передачи и хранения.

Кроме проблемы «недоверенной среды», в среде Интернет невозможно обеспечить ситуацию, когда все участники обмена перечислены заранее в какой-то базе данных и центр может в любой момент быстро удостоверить «личность» пользователя. По этим причинам очень важную роль в обеспечении безопасности играют *системы шифрования данных*, которую рассмотрим на следующих уроках.

Для закрепления изученного материала предлагаем использовать проверочный тест из шести ситуаций, которые когда-либо имели место. К каждой ситуации подобраны задания с выбором правильного ответа (одного или нескольких) из предложенных. Ситуации можно разобрать в классе либо

дать тест для самостоятельного (домашнего) выполнения с обязательным последующим разбором ответов.

Ситуация 1

По сообщению пресс-служб МВД и ФСБ РФ, два гражданина РФ, имея специальные знания в области компьютерной техники и программного обеспечения, а также обладая опытом работы в сети Интернет, применили знания и опыт для незаконного обогащения. Для достижения криминальной цели они приобрели на стороне вредоносную программу и с ее помощью начали заражать компьютеры различных пользователей. Способ – перенаправление потенциальных жертв на созданный мошенниками поддельный сайт с последующей кражей индивидуальных кодов доступа к интернет-банкингу. После получения кодов злоумышленники взламывали счета вкладчиков, а лежащие на них денежные средства обналичивали. В ходе следствия установлено, что в период с марта 2013 по май 2015 г. их жертвами стали более 170 человек, а сумма ущерба – порядка 13 млн руб.

1.1. Определите, какие аспекты безопасности были нарушены.

а) доступность

б) конфиденциальность

в) целостность

г) уязвимость

1.2. Какие средства были применены злоумышленником?

а) вредоносные программы – вирусы

б) программы сбора данных

в) средства удаленного управления

г) агенты организации атак

1.3. Примените к ситуации нормы Федерального закона «Об информации, информационных технологиях и о защите информации» от 27 июля 2006 г. № 149-ФЗ и определите, кто должен был обеспечить безопасность данных.

а) провайдер

б) оператор информационной системы

в) обладатель информации

г) владелец банка

1.4. Для ответа на этот вопрос вам необходимо воспользоваться законодательными актами, действовавшими в тот период времени, когда ситуация происходила. За что и какие меры наказания были предусмотрены законодательством для злоумышленника?

а) за кражу имени пользователя и пароля в соответствии с уголовным кодексом

б) за несанкционированный доступ к информации в соответствии с административным кодексом

в) за создание и использование вредоносных программ, неправомерный доступ к компьютерной информации и мошенничество в соответствии с уголовным кодексом

г) за кражу и использование в своих целях услуг связи в соответствии с административным кодексом

Ситуация 2

Жители Москвы, пользующиеся единым проездным билетом, не смогли в пятницу утром (1 июня 2007 г.) пройти через системы контроля в наземном транспорте, а потом через турникеты в метро. Неполадки в электронной системе Мосгортранса аннулировали все билеты, прошедшие утром через систему автоматизированного контроля пассажиров. Как признались сами транспортники, во всем надо винить жару и грозу.

2.1. Определите, какие аспекты безопасности были нарушены.

а) доступность

б) конфиденциальность

в) целостность

г) уязвимость

2.2. Какие меры по защите информации необходимо было предпринять для предупреждения такой ситуации?

а) защитить все элементы информационной системы от нарушений температурного режима, влажности, электромагнитного фона и сбоев в системе электропитания

б) предусмотреть возможность незамедлительного восстановления информации, модифицированной или уничтоженной вследствие несанкционированного доступа к ней

в) обеспечить постоянный контроль уровня защищенности информации

г) защитить все элементы информационной системы от несанкционированного доступа к информации

Ситуация 3

Специалист по безопасности Дидье Стивенс (Didier Stevens) провел небольшой эксперимент, разместив в Google Adwords объявление: «Ваш компьютер свободен от вирусов? Заразите его здесь!»

За шесть месяцев кампании, которая обошлась в 23$, по этому объявлению щелкнуло 409 человек.

3.1. Какие аспекты безопасности были нарушены?

а) доступность

б) конфиденциальность

в) целостность

г) никакие

3.2. Какие средства были использованы для нарушения безопасности?

а) вредоносные программы – вирусы

б) никакие

в) средства удаленного управления

г) агенты организации атак

3.3. Какие меры по защите информации применимы в этом случае? Из приведенного списка выберите необходимые действия и запишите получившуюся последовательность букв.

а) не открывать почтовые сообщения из неизвестных источников

б) заблокировать выполнение активного содержимого на недоверенных сайтах

в) возможно обновить ПО

г) взять за правило не «ходить» по провокационным объявлениям или хотя бы вникать в смысл прочитанного в них до перехода

3.4. За что могут быть применены меры воздействия к автору объявления?

а) за распространение вредоносных программ

б) никакие меры воздействия не могут быть применены

в) за создание вредоносных программ

г) за ложные сведения в рекламе

Ситуация 4

Подоспело известие о том, что в немецкую версию статьи о черве Lovesan (он же MS Blaster) была вставлена ложная информация о новых версиях червя, а заодно добавлены ссылки на предлагаемые «исправления». После обнаружения случившегося администраторы немедленно удалили исправления в статье, но история на этом не закончилась. Принцип «что написано в вики, не вырубишь delete» проявил себя во всей красе. Страницы остались лежать в архиве, после чего атакующие стали рассылать спам от имени Википедии, вставляя в него ссылки на архив. Поскольку ссылки вели на вполне добропорядочный сайт, этим письмам зачастую удавалось прорваться сквозь защитные фильтры.

4.1. Какие средства на локальной машине позволят защититься от такой атаки?

а) блокирование почтовых сообщений

б) актуальные антивирусные программы

в) переустановка ОС

г) настройка брандмауэра

Ситуация 5

27 декабря 2005 г. была опубликована информация об уязвимости в библиотеке обработки WMF-файлов в ОС Windows. Уязвимость позволяет выполнить произвольный код, внедренный в изображение, на машине пользователя. Обновление ОС выпущено фирмой Microsoft 5 января 2006 г.

5.1. Какой наиболее вероятный способ для злоумышленника воспользоваться уязвимостью?

а) предложить возможность установить бесплатно обновление ОС

б) воспользоваться зараженной иллюстрацией с вредоносным кодом со специального сайта, или размещенной на форуме, или внедренной в текст письма

в) запустить на исполнение графический файл

г) попытаться широко распространить вредоносный код в форме изображения, входящего в состав крупной коммерческой библиотеки изображений

5.2. Как уменьшить риск поражения на непродолжительное время?

а) временно отключить обработку таких изображений

б) не пользоваться электронной почтой

в) временно не пользоваться ресурсами крупных библиотек

г) не читать сообщения на форумах, в блогах или гостевых книгах

5.3. Что необходимо предпринять для защиты?

а) переустановить почтовую программу

б) настроить сетевой брандмауэр

в) установить обновление операционной системы

г) сохранять все изображения перед просмотром на локальный носитель

Ситуация 6

Вирус-червь Nimda использует для заражения несколько путей: рассылку писем со своими копиями по электронной почте, доступные файловые ресурсы локальной сети, одну из уязвимостей ОС Windows, просмотр зараженных web-сайтов (подставные ссылки на зараженные исполняемые файлы), средства удаленного управления, оставленные другими вирусами.

6.1. Какие меры необходимо предпринять, чтобы предотвратить заражение? Из приведенного списка выберите необходимые действия и запишите получившуюся последовательность букв.

а) использовать антивирусный фильтр почтовых сообщений в актуальном состоянии

б) переустановить операционную систему

в) использовать антивирусный сторож

г) настроить брандмауэр на блокировку попыток обращения к локальной машине без запроса с ее стороны

д) провести полную проверку системы и установить обновление, исправляющее уязвимость

Ответы на тест

№ вопроса	1.1	1.2	1.3	1.4	2.1	2.2	3.1	3.2	3.3	3.4	4.1	5.1	5.2	5.3	6.1
Правильный ответ	Б	АБ	В	В	А	А	Г	Б	Г	Б	Б	Б	А	В	АВГД

Новым материалом для обучающихся является криптографическая тема. В ней необходимо рассмотреть методы шифрования и использование различных шифров. Предлагаем содержательный материал по методам шифрования, использованию простых шифров и по реализации алгоритма шифрования RSA, используемого в цифровой подписи. Популярные у старшеклассников вопросы: криптовалюты, хэш-функции и блокчейн с примерами использования раскрыты

в практикуме для инженерных классов: Самылкина Н. Н., Калинин И. А., Тарапата В. В., Салахова А. А. Информатика: 8–11-е классы: практикум. – М. : Просвещение, 2023. – 157 с.: ил. – URL: https://lbz.ru/books/1171/

Внимание! Тему урока **«Методы шифрования данных»** и следующую тему **«Использование различных шифров»** можно менять местами, на усмотрение учителя.

Методы шифрования данных

Содержание темы урока

Шифрование – это процесс преобразования данных с целью затруднения получения доступа к исходным (не преобразованным) данным.

Шифрование часто выполняется с помощью **ключа** – некоторых данных, которые используются при шифровании и позволяют выполнить обратное действие. По понятным причинам как минимум часть этой информации не разглашается.

Без ключа получение исходных данных (или повторное шифрование) становится гораздо более трудоемким и таким образом для заданного уровня рентабельности становится недостижимым.

Комплект из алгоритма шифрования, алгоритма расшифровки (если он существует) и необходимых для их работы данных (например, таблиц) называют **шифром.**

Стоит заметить, что получение исходных данных «в обход» процедуры все равно остается возможным, поэтому, когда речь идет о защите данных, вводят понятие **стойкости шифра**. Стойкость шифра оценивается как минимальное время, требуемое на получение расшифрованного сообщения. Для всех современных систем шифрования такой подбор возможен, поэтому не бывает абсолютно стойких систем и ключей. Когда говорят о надежной защите, на самом деле речь идет о достаточности мер защиты для имеющихся задач.

Классифицировать шифры можно по разным основаниям.

1. По общему принципу обработки: **поточные** и **блочные**. Поточные методы выполняют шифрование побайтно (или побитно), используя только уже полученные данные. Эти методы позволяют шифровать данные при передаче по каналам связи. Полученные данные сразу можно расшифровывать. Блочные

шифруют данные блоками **заданной длины,** выполняя преобразования с каждым блоком в отдельности. Для расшифровки нужно получить блок целиком.

2. По существованию дешифрующего преобразования: **обратимое**, для которого существует обратное преобразование (дешифрующее), и **необратимое**, для которого такого преобразования не существует[1].

3. По используемым ключам: **симметричные** и **асимметричные**. Симметричные методы шифрования используют один и тот же ключ и для шифрования, и для расшифровки сообщений. Асимметричные – два разных (хотя обычно связанных) ключа.

Исторически первыми использовались симметричные системы шифрования – те, в которых ключ должны были знать только доверенные лица. Потеря ключа означала и компрометацию канала. Чтобы затруднить вычисление ключа, их использовали целыми наборами, например, в виде шифроблокнота.

Канал связи, сообщения в котором шифруются и таким образом защищаются от перехвата и подмены, часто называют *защищенным* или *закрытым* каналом.

В современных системах для организации защищенного канала связь начинают, как правило, с выполнения специальной процедуры согласования метода шифрования и сеансового ключа, т. е. общего для двух абонентов ключа шифрования, который будет применяться ими в течение сеанса. Такая схема препятствует краже ключа и затрудняет его дешифровку, поскольку сокращается объем данных для анализа и время их использования. Предполагаемая выгода от дешифровки ключа оказывается меньше, чем затраты ресурсов на его подбор.

Возникает вопрос – если ключ согласовывается фактически при обращении, то как все-таки понять, кто на другом конце канала связи, особенно если это совершенно неизвестный абонент.

Системы шифрования с закрытым ключом позволяют организовать быстрые и надежно защищенные каналы связи, сравнительно дешевые

[1] Чаще всего это означает, что исходное преобразование в нескольких разных случаях приводит к одному результату. Необходимое условие состоит в том, что алгоритм обратного преобразования если и существует, то требует непомерно большого количества вычислений.

в реализации, но при всех достоинствах, есть несколько существенных проблем, которые мы теперь можем сформулировать с введенными терминами:

1. Для обмена сообщениями участникам нужно иметь общий ключ шифрования, полученный таким образом, при котором исключена возможность его перехвата. То есть необходим способ безопасного обмена ключами.

2. Два незнакомых абонента не могут проверить – тот ли их собеседник, за кого себя выдает. С формальной точки зрения, «своим» должен считаться тот абонент, который может подтвердить владение ключом.

3. Нельзя точно установить, кто из двух абонентов послал сообщение – любой из них мог его зашифровать и расшифровать.

4. Если есть некий центр хранения ключей – он может выдать себя за кого угодно.

Сертификаты и доверие

В среде Интернет, как мы уже указали, невозможно гарантировать защиту канала от перехвата (а значит и от подмены ключа), нельзя знать всех абонентов и создавать всеобщие проверяющие центры. Кроме того, любой торговый обмен требует возможности предъявить документ, от которого «подписавший» его не сможет отказаться, а «проверяющий» центр мог бы выдать себя за кого угодно – и таким образом ему нельзя доверять по определению.

Эти задачи решаются с помощью асимметричных шифров с открытым ключом. В таких шифрах собственно шифрование и дешифрование выполняются с помощью разных ключей (они, конечно, должны быть связаны специальным соотношением). Тогда один из этих ключей становится открытым (общеизвестным), а второй вообще никому не сообщается и называется секретным.

Перечислим особенности этого метода шифрования, которые позволяют решать указанные проблемы:

1. Владение открытым ключом не дает возможности выдавать себя за другого. Оно дает только возможность зашифровать сообщение для кого-то или расшифровать сообщение от кого-то.

2. Появляется возможность создавать электронную подпись – контрольный «отпечаток» сообщения. Создать его может только отправитель,

т. е. абонент, имеющий оба ключа, а проверить – любой участник обмена, у которого есть открытый ключ.

Использовать описанные методы можно только в том случае, когда можно ответить на вопрос: стоит ли верить переданному открытому ключу?

Существует два подхода к решению этой задачи, реализуемых в виде комплексов программ и данных, имеющих общее название – ***инфраструктура открытого ключа***. В качестве удостоверения в таких системах используется специальное сообщение стандартного формата[1] – ***сертификат***. Сертификат включает в себя данные о пользователе (фамилию, имя и отчество или адрес сайта, электронной почты и т. п.), срок действия, цели для которых сертификат выдан[2], открытый ключ и удостоверение подлинности (рис. 6).

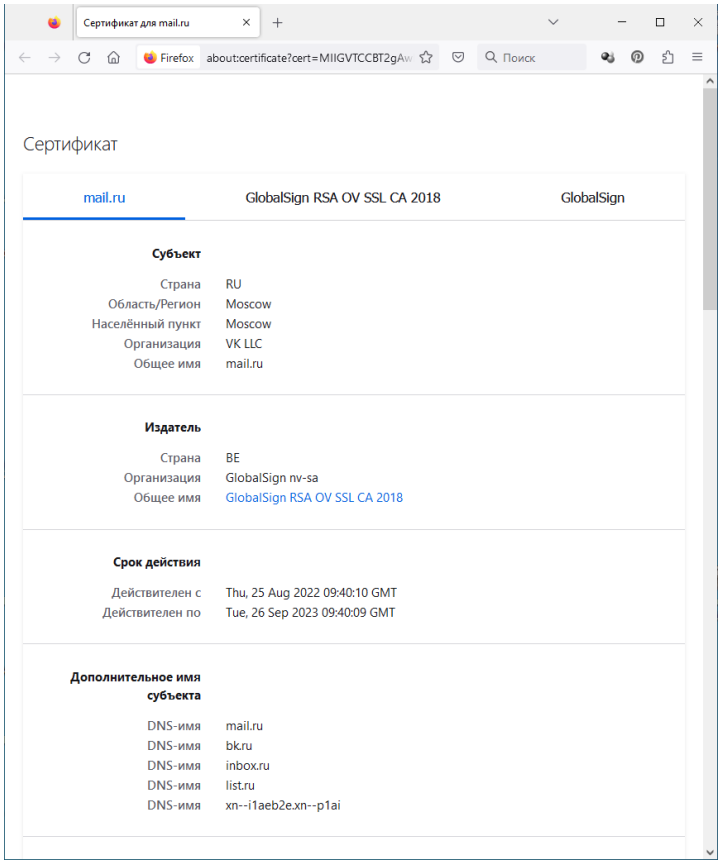


Рис. 6. Данные о сертификате популярной почтовой службы

Подлинность сертификата может быть удостоверена двумя способами:

1. Способ, ориентированный на сертификаты и удостоверяющие центры. Чтобы удостоверять подлинность сертификата, он включает в себя «подпись»

[1] См., например, ссылку в начале параграфа.

[2] Нельзя, например, подписать личное сообщение с помощью сертификата, удостоверяющего подлинность сервера.

специально уполномоченного узла – удостоверяющего центра. ***Удостоверяющий центр*** – организация, подписывающая сертификаты, на основании полученных после некоторой проверки данных[1]. Удостоверяющий центр не имеет закрытого ключа пользователя и не может изменить содержание сертификата. Пользователь не знает закрытого ключа центра и не может изменить подпись. Конечно, чтобы проверить сертификат нужно доверять центру (т. е. сертификат самого цента с его открытым ключом должен быть на компьютере заранее). Знать все центры заранее невозможно, а поэтому центры связываются по цепочке, т. е. один центр может выдать сертификат другому центру, тот – третьему и т. д. Если пользователь доверяет корневому центру, и не знает о компрометации подчиненных ему – он может быть уверен, что абонент именно тот, за кого себя выдает. Корневой сертификат пользователь устанавливает сам, или получает вместе с ОС. Существует несколько крупных центров сертификации, сертификаты и открытые ключи которых приходят в поставке операционных систем и в дальнейшем постоянно обновляются. Такие сертификаты могут пересылаться с web-страницами, программами (особенно ориентированными на распространение через Интернет), сообщениями электронной почты. На базе такой инфраструктуры действуют платежные системы, торговые площадки, корпоративные системы и т. п.

2. Способ, ориентированный на коллективное доверие. В этом случае создается структура, в которой вы лично определяете тех субъектов (людей, программ, устройств), которым доверяете. Если во время проверки выясняется, что кто-то из тех, с кем вы работали раньше, тоже доверяет этому сертификату, то сертификат считается доверенным. Такой подход использован в популярном и известном протоколе организации защищенного канала связи SSH.

Приведем два наиболее частых примера использования системы доверенных сертификатов:

1. **Организация защищенного доступа к сайтам в среде Интернет.** Для исключения перехвата и подмены данных в основном протоколе обмена гипертекстовыми страницами, как мы знаем, средств не предусмотрено. Но для защиты применяется протокол защиты данных на транспортном уровне

[1] Например, организации сертификат выдается после получения и проверки заверенных копий учредительных документов, человеку (долгосрочный сертификат) – на основании удостоверения личности и т. д.

(Transport Layer Security, а ранее – Secure Socket Layer), который превращает протокол HTTP в защищенный HTTPS.

Перед началом обмена данными в этом протоколе устанавливается защищенное соединение.

Процедура установления защищенного соединения предусматривает согласование методов шифрования, а также формирования сеансового ключа. Но в этом случае возникает проблема: каким образом предупредить перехват данных уже в самом начале сеанса, когда может вмешаться злоумышленник? Для этого минимум одна сторона (сервер) должен подтвердить свою подлинность, отправив данные с применением шифрования с открытым ключом и своего сертификата, т. е. прислать данные, которые кроме него никто не может сформировать (поскольку не имеет закрытого ключа). Сертификат в этом случае крайне желательно иметь подтвержденный доверенным центром сертификации.

Если вся процедура соблюдена, то каждый сеанс связи проводится с подтвержденным абонентом, причем ключ шифрования временный, нигде не хранится, и его подбор – крайне затратная процедура.

2. **Электронная подпись (ЭП) документов.** Аналогично решается проблема формирования юридически-значимых документов. Для подтверждения подлинности файла формируется его электронная подпись (ранее называлась электронно-цифровая подпись) – набор символов, который зависит от содержания документа и шифруется закрытым ключом.

После этого документ с подписью (иногда в виде отдельного файла, иногда части документа) и сертификатом можно отправить контрагенту или на хранение. Подписавший не может «незаметно» отказаться от подписи, поскольку никто кроме него не имеет закрытого ключа, получатель не может «незаметно» изменить содержание документа, поскольку в этом случае сформировав дайджест и сравнив его с уже имеющимся в ЭП, мы сразу увидим несовпадение. А «переделать» ЭП можно только имея закрытый ключ, соответствующий сертификату.

В обоих решениях, как мы видим, ключевой особенностью является проблема доверия общим удостоверяющим центрам.

В Российской Федерации создана инфраструктура удостоверяющих центров, соответствующая российскому законодательству в области криптографических средств, что позволяет снизить зависимость от зарубежных

удостоверяющих центров, но требует установки и применения дополнительного программного обеспечения.

Вопросы для закрепления изученного материала

1. Как называется существующая возможность нарушения безопасности за счет ошибки в конфигурации операционной системы?

2. Что можно выполнить, чтобы сделать перехват информации бесполезным?

3. Вы получаете по электронной почте письмо с предложением сменить ваш нестойкий пароль на другой, указанный в письме. Будете менять?

4. В результате ошибки администратора стерт один из вспомогательных файлов базы данных. Какой аспект безопасности нарушен?

5. Укажите несколько ситуаций, которые не дают отказаться от списков отзыва сертификатов.

6. Почему никто не предлагает ввести единый мировой корневой удостоверяющий центр?

7. Что мешает считать сервис Wikipedia и ему подобные достоверным источником информации?

8. Возможная ситуация: не открывается сайт. Как выяснить причину? Укажите технические действия, возможные результаты действий и ваши выводы.

Ответы на вопросы:

1. Уязвимость.

2. Зашифровать ее достаточно стойким шифром. Наложить временные ограничения на ее использования, которые не дают времени на использование перехваченной информации (если это возможно).

3. Нет. Если пароль прислан кем-то другим, то он известен как минимум отправителю. Кроме того, а как отправитель узнал ваш пароль, чтобы определить его «нестойкость»?

4. Целостность, а возможно – если работа остановилась – доступность.

5. В результате утечки доступ к закрытому ключу получил злоумышленник. При отсутствии понятия «Список отзыва» он сможет использовать сертификат до истечения его срока.

Скомпрометирован сертификат подчиненного удостоверяющего центра, при этом список выданных им сертификатов уничтожен. Необходимо отозвать все сертификаты – но известить об этом тех, кто их получал невозможно.

6. Потому что в этом случае зависимость от этого центра станет критической.

7. Не существует технических процедур, которые могли бы адекватно оценить полноту и достоверность информации. А значит, поскольку это открытый сервис, злоумышленник (или просто не владеющий всей информацией пользователь) может внести туда ошибочные сведения – и они будут там находиться неопределенное время.

8. Надо определить характер недоступности:

а) недоступны никакие сайты, возможно, проблема возникла с доступом в Интернет у вас или вашего провайдера. Надо проверить доступность вашего шлюза, шлюза провайдера, DNS-серверов (например, 8.8.8.8);

б) не известно имя – ошибка в DNS-имени или проблемы с доменом. Надо проверить разрешение имен с помощью утилиты nslookup;

в) сайт не отвечает – возможно, остановлен или возникли проблемы с доступом к Интернету уже на его стороне;

г) при обработке данных на сайте возникают ошибки. Вы можете только сообщить об этом в поддержку сайта, если она вам известна.

Использование различных шифров (Цезаря, Гронсфельда или Виженера, аффинного). Шифры перестановки

Содержание темы урока

Самым простым считается шифр перестановки. Алгоритм данного вида шифрования подразумевает под собой изменение порядка следования элементов исходного открытого текста, при этом сами элементы не изменяются.

Например, зашифруем слово ВЕСНА с помощью простого шифра перестановки. Пронумеруем каждую букву этого слова

В	Е	С	Н	А
1	2	3	4	5

Условимся, что буква под номером 1 перейдет на 4-е место, и далее так: 2 →3, 3→1, 4→5, 5→2. Получаем:

В	Е	С	Н	А
1	2	3	4	5

Осуществим переход к новым местам букв (через «’» обозначается номер буквы в новом слове):

4’	3’	1’	5’	2’

Переставим буквы на новые места в порядке возрастания цифр, под которыми они значатся:

1’	2’	3’	4’	5’
С	А	Е	В	Н

САЕВН – зашифрованное с помощью шифра перестановки слово ВЕСНА.

Запишем условие нашей перестановки в более удобном виде, чем простое перечисление. Пусть наше условие будет таблицей с двумя строками, где в верхней строке мы запишем порядковые номера букв слова ВЕСНА, а во второй – номер места, на которое буква под определенным номером должна перейти в шифротексте в соответствии с условием. (Буква под номером 1 перейдет на 4-е место, под номером 2 – на 3-е и далее: 3→1, 4→5, 5→2.) Получим:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 3 & 1 & 5 & 2 \end{pmatrix},$$

В общем виде записать шифр перестановки для открытого сообщения длины n будет выглядеть таким образом:

$$\begin{pmatrix} 1 & 2 & \cdots & n \\ i_1 & i_2 & \cdots & i_n \end{pmatrix},$$

где i_1 – номер места шифротекста, на которое попадает первая буква при заданном преобразовании, i_2 – номер места шифротекста, на которая попадает вторая буква и т. д. В верхнем ряду стоят порядковые номера букв исходного сообщения, в нижнем – числа от 1 до n в произвольном порядке.

Дешифрование при знании таблицы шифра перестановки осуществляется очень просто. Попробуем расшифровать сообщение ОАКШЛ с помощью перестановки, описанной выше.

Мы знаем, что буква, оказавшаяся на первом месте в шифртексте, стояла в открытом сообщении на третьем месте, буква, стоящая сейчас на втором месте, ранее стояла на пятом и т. д. Запишем обратное преобразование:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 5 & 2 & 1 & 4 \end{pmatrix},$$

где в верхнем ряду укажем номера мест букв в шифротексте, а в нижнем – в открытом сообщении.

Получим слово ВЕСНА.

Можно заметить, что шифры простой перестановки станут неудобны при большом значении n.

Задания на отработку шифра перестановки

1. Зашифруйте слово с помощью данных перестановок.

ЗАДУМКА

$$\begin{pmatrix} 1 & 23 & 45 & 6 & 7 \\ 6 & 43 & 71 & 5 & 2 \end{pmatrix}$$

2. Дешифруйте сообщение, зная шифрующую перестановку.

НЫБНАА

$$\begin{pmatrix} 1 & 2 & 34 & 5 & 6 \\ 3 & 5 & 16 & 4 & 2 \end{pmatrix}$$

Шифр Цезаря

Для применения шифра Цезаря к открытому тексту необходимо каждую букву послания заменить на букву того же алфавита путем сдвига на *k* букв.

При достижении конца алфавита выполняется циклический переход к его началу.

Предположим, что используется *N*-буквенный алфавит с числовыми эквивалентами букв: **{0, 1, ... , *N* – 1}**. Тогда каждая буква шифротекста будет находится по правилу:

$$C = f(p) = p + k,$$

где p – буква открытого текста; C – буква шифротекста; k – целое число, являющееся ключом шифрования.

Например, зашифруем букву «ц» шифром Цезаря с ключом шифрования 4.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
а	б	в	г	д	е	е	ж	з	и	й	к	л	м	н	о	п
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	
р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	

$$C = f(2) = 2 + 4 = 6 \rightarrow \text{ъ}$$

Обратим внимание, что при шифровании букв с большим порядковым номером легко выйти за пределы алфавита. Тогда вновь переходим к его началу. Допустим, зашифруем букву «ю» с ключом из предыдущего примера.

$$C = f(31) = 31 + 4 = 35 \rightarrow ?$$

Для нахождения номера буквы в алфавите достаточно найти остаток от деления получившегося числа на количество букв в алфавите.

$$35 \bmod 33 = 2 \rightarrow \text{в}$$

Немного усовершенствуем нашу формулу:

$$\boldsymbol{C = f(p) \equiv p + k \pmod{N}}.$$

Данная запись означает, что мы ищем остаток от деления на N значения выражения $p + k$.

Например, зашифруем шифром Цезаря с ключом $k = 15$ слово ДОМ. В русском алфавите 33 буквы (N). Шифрующее преобразование примет вид:

$$C = f(p) \equiv p + 15 \pmod{33}.$$

В формуле знак $\equiv$ означает операцию эквивалентности (логическая равнозначность).

$$f(\text{д}) \rightarrow f(4) \equiv 4 + 15 \pmod{33} \equiv 19 \rightarrow \text{Т}$$

$$f(\text{о}) \rightarrow f(15) \equiv 15 + 15 \pmod{33} \equiv 30 \rightarrow \text{Э}$$

$$f(\text{м}) \rightarrow f(13) \equiv 13 + 15 \pmod{33} \equiv 28 \rightarrow \text{Ы}$$

Получим ответ: ТЭЫ.

Общеизвестно, что Юлий Цезарь в своих посланиях использовал ключ $k = 3$, т. е. каждая буква его послания являлась буквой, смещенной от исходного места на три позиции.

Для дешифрования элемента шифртекста $C \in \{0, 1, \ldots, N-1\}$ необходимо вычислить

$$\boldsymbol{p = f^{-1}(C) \equiv C - k \pmod{N}}.$$

Данный вид шифрования очень прост, но, к сожалению, не является надежным. Зашифрованное послание можно с легкостью дешифровать при помощи частотного анализа. Допустим, чаще других в шифртексте встречается буква «т». Это значит, что сдвиг преобразует «о» = 15 (наиболее часто встречаемую букву в русском алфавите) в «т» = 19.

Найдем k:

$$19 \equiv 15 + k \pmod{33},$$

$$k \equiv 4 \pmod{33}.$$

Чтобы дешифровать сообщение ОФМУЦТЖФДШМГ, остается вычесть 4 (по модулю 33) из числовых эквивалентов букв послания.

ОФМУЦТЖФДШМГ =

= 15 21 13 20 23 19 7 21 4 25 13 3 11 17 9 16 19 15 3 17 0 21 9 32 =

= КРИПТОГРАФИЯ

Замечание: при решении сравнений могут возникать случаи, когда число равняется отрицательному числу по модулю.

$$17 + x \equiv 9 \pmod{5}$$

$$x \equiv -8 \pmod{5}$$

В таком случае представляем в следующем виде:

1) $-8 = 5 \cdot (-1) - 3$ (т. е. при делении на 5 итог отрицательный остаток –3);

2) $-8 = 5 \cdot (-2) + 2$ (при таком разложении остаток положительный).

При решении задач нам потребуется положительное решение, поэтому:

$$x \equiv 2 \pmod{5}$$

Про числа –8, –3, 2, 7 и т. д. говорят, что они сравнимы по модулю, так как при делении на эти числа дают один и тот же положительный остаток 2. Но даже если имеется небольшое послание, не дающее нам определить часто встречающуюся букву, то для этого имеется всего 33 возможности – можно просто попробовать их все. В конце концов лишь одному значению будет соответствовать осмысленное сообщение, такое и будет ключом шифрования.

Шифр Гронсфельда (Виженера)

У рассмотренного шифра Цезаря существует модификация в виде дополнения числовым ключом, делающая его более надежным. Для этого за каждой буквой исходного текста закрепляется цифра числового ключа. Если длина ключа меньше длины сообщения, то ключ циклически повторяется до конца. Ключ шифрования для каждой буквы исходного текста получается свой, определенный числовым ключом. Таким образом, получается шифр Гронсфельда. Он похож на шифр Виженера, но проще за счет использования 10 цифр, вместо 26 букв латинского алфавита или 33 букв кириллицы.

Например: зашифруем слово ВИЖЕНЕР с помощью числового ключа $k = 1572$.

1	5	7	2	1	5	7
В	И	Ж	Е	Н	Е	Р

$$f(\text{в}) \rightarrow f(2) \equiv 2 + 1 \pmod{33} \equiv 3 \rightarrow \text{г}$$

$$f(\text{и}) \rightarrow f(9) \equiv 9 + 5 \pmod{33} \equiv 14 \rightarrow \text{н}$$

$$f(\text{ж}) \rightarrow f(7) \equiv 7 + 1 \pmod{33} \equiv 8 \rightarrow \text{з}$$

$$f(\text{е}) \rightarrow f(5) \equiv 5 + 2 \pmod{33} \equiv 7 \rightarrow \text{ж}$$

$$f(\text{н}) \rightarrow f(14) \equiv 14 + 7 \pmod{33} \equiv 21 \rightarrow \text{ф}$$

$$f(\text{е}) \rightarrow f(5) \equiv 5 + 5 \pmod{33} \equiv 10 \rightarrow \text{й}$$

$$f(\text{р}) \rightarrow f(17) \equiv 17 + 7 \pmod{33} \equiv 24 \rightarrow \text{ч}$$

Получим закодированное слово – ГНЗЖФЙЧ.

Обратите внимание, что при использовании длинного числового ключа одинаковые буквы при шифровании переходят в разные, а разным буквам может соответствовать одна буква шифротекста. Все это затрудняет частотный анализ.

Несомненным достоинством шифра Гронсфельда является то, что количество возможных числовых ключей практически неисчерпаемо. Это делает частотный анализ затруднительным, но все же возможным, если учесть, что в числовом ключе каждая цифра имеет только десять значений, а значит, имеется лишь десять вариантов прочтения каждой буквы шифротекста. Однако, шифр Гронсфельда допускает дальнейшие модификации, улучшающие его стойкость, в частности двойное шифрование разными числовыми ключами.

Шифр Гронсфельда был очень популярен в европейских странах, а в других странах использовали шифр Виженера. Отличаются они тем, что вместо цифр в шифре Виженера используют буквы.

О шифре Виженера см.: https://calculatorium.ru/cryptography/vigenere-cipher

Аффинный шифр

Рассмотрим более надежный шифр – аффинный. Для шифрования буквы исходного сообщения пользуются аффинными отображениями:

$$\boldsymbol{C \equiv a \cdot p + b \pmod{N}},$$

где a и b – фиксированные целые числа, образующие ключ шифрования.

Замечание: на выбор a накладывается ограничение: переменные a и N должны быть взаимно простыми, в противном случае невозможно будет однозначно восстановить открытый текст по шифрованному.

Например, зашифруем сообщение СИСТЕМА, используя аффинное отображение с ключом шифрования $a = 7$, $b = 4$.

$$f(\text{с}) \to f(18) \equiv \mathbf{7 \cdot 18 + 4 \ (mod\ 33)} \equiv 130 \equiv 31 \to \text{ю}$$

$$f(\text{и}) \to f(9) \equiv \mathbf{7 \cdot 9 + 4 \ (mod\ 33)} \equiv 1 \to \text{б}$$

$$f(\text{с}) \to f(18) \equiv \mathbf{7 \cdot 18 + 4 \ (mod\ 33)} \equiv 31 \to \text{ю}$$

$$f(\text{т}) \to f(19) \equiv \mathbf{7 \cdot 19 + 4 \ (mod\ 33)} \equiv 5 \to \text{е}$$

$$f(\text{е}) \to f(5) \equiv \mathbf{7 \cdot 5 + 4 \ (mod\ 33)} \equiv 6 \to \text{е}$$

$$f(\text{м}) \to f(13) \equiv \mathbf{7 \cdot 13 + 4 \ (mod\ 33)} \equiv 25 \to \text{ш}$$

$$f(\text{а}) \to f(0) \equiv \mathbf{7 \cdot 0 + 4 \ (mod\ 33)} \equiv 4 \to \text{д}$$

Получим ЮБЮЕЕШД.

Для дешифрования сообщения, зашифрованного применением аффинного отображения нужно выразить p через C:

$$\boldsymbol{p \equiv a^{-1} \cdot C + b^{-1} \ (mod\ N)},$$

где a^{-1} – обратное к а по модулю число, $b^{-1} = -a^{-1} \cdot b$

Например, определим, какая буква алфавита шифруется буквой «л» при аффинном преобразовании.

1. Сначала найдем a^{-1}:

$$7 \cdot a^{-1} \equiv 1 \ (\text{mod } 33)$$

$$a^{-1} \equiv 19 \ (\text{mod } 33)$$

Замечание: при нахождении неизвестного из сравнения $\boldsymbol{k \cdot x \equiv m \ (mod\ N)}$ советуем найти такое целое x, при котором $k \cdot x$ равнялось одному из чисел $m + N$, $m + 2N$, $m + 3N$ и т. д. В разбираемом примере:

$7 \cdot x = 34$ – нет такого целого;

$7 \cdot x = 67$ – нет такого целого;

$7 \cdot x = 100$ – нет такого целого;

$7 \cdot x = 133$ – есть, $x = 19$.

2. Дешифрующее преобразование примет вид:

$$p = 19 \cdot C - 19 \cdot 4 \ (\text{mod } 33);$$

$$p = 19 \cdot C - 76 \ (\text{mod } 33).$$

3. Подставим вместо C порядковый номер буквы «л»:

$$p \equiv 19 \cdot 12 - 76 \pmod{N} \equiv 152 \pmod{N} \equiv 20 \pmod{N} \rightarrow y.$$

Ответ: буква «у» при данном шифровании переходит в букву «л».

Для нахождения ключа при криптоанализе необходимо знать две буквы шифротекста и соответствующие им буквы открытого текста, для того чтобы можно было составить и решить систему сравнений.

Например, в шифротексте чаще всего встречаются буквы «т» и «к». Разумно предположить, что ими зашифрованы две наиболее часто встречающиеся буквы алфавита «о» и «е». Заменяя буквы их числовыми эквивалентами и подставляя последние в формулу дешифрования, получаем:

$$19 \cdot a^{-1} + b^{-1} = 15 \pmod{33};$$

$$11 \cdot a^{-1} + b^{-1} = 5 \pmod{33}.$$

Замечание: система сравнений в данном случае решается как система уравнений. Методом вычитания получим:

$$8 \cdot a^{-1} = 10 \pmod{33};$$

$$a^{-1} = 26.$$

Находим b^{-1}, подставляя a^{-1} в любое сравнение системы и решая его: $b^{-1} = 16$, т. е. все сообщение может быть дешифровано применением формулы:

$$p \equiv 26 \cdot C + 16 \pmod{33}.$$

Задачи на отработку навыков шифрования по темам: «Шифр Цезаря», «Шифр Гронсфельда (Виженера)» и «Аффинный шифр»

1. Зашифруйте при помощи шифра Цезаря, используя ключ $k = 8$, следующее сообщение: НАЧИНАЕМ (используется полный 33-буквенный русский алфавит; для удобства прочтения пробелы не пропускать – шифровать их не нужно).

2. Поупражняемся в дешифровании сообщений (пунктуация сохранена только для удобства прочтения). Найдем первое и второе значения ключа. Для этого нужно декодировать два сообщения: 1-е – ЕЯЫМЦДСМИ; 2-е – ЗИПИСМИ; значение ключа = 4. В них зашифрованы две арифметические операции. Подставьте их на место «...» в арифметические выражения, вычислите их значения. Результаты – первое и второе значение ключа соответственно:

1) 4 … 3 =

2) 28 … 7 =

3. Декодируйте с помощью шифра Гронсфельда сообщение: ШЖЩЬТЛ; $k = 127$. Результат – третье значение ключа.

4. Взломайте сообщение ШРУМЗ, зная, что при шифровании использовалось трехразрядное число и что буквы «с» и «ч» были зашифрованы буквами «у» и «ш» соответственно. Одно из значений ключа – 8. Данное сообщение вам понадобится при решении задачи 6.

5. Зашифруйте сообщение ЙИГЙОА при помощи аффинного шифрования: $a = 2$, $b = 32$. Данное сообщение вам понадобится при решении задачи 6.

6. Найдите последнее значение ключа.

Последнее значение ключа – это ЙЛРЕРВЯ (результат пункта 5) и СЙВ (результат пункта 4). Слова ЙЛРЕРВЯ и СЙВ зашифрованы при помощи аффинного шифрования. Декодируйте сообщения, чтобы понять, что написано в задании.

1) ЙЛРЕРВЯ

$a = 2$

$b = 7$

2) СЙВ

$a = 4$

$b = 2$

Поздравляем! Вы собрали ключ. Теперь вы можете дешифровать послание Юлия Цезаря: ПУЯЪГХИФФЧЫРУИПД.

№ задания	Ответ
Послание Цезаря	ОПЫТ ВСЕМУ УЧИТЕЛЬ (значение ключа – 1448)
1	ХЗЯРХЗМФ
2	1) 1; 2) 4
3	ЧЕТЫРЕ
4	ЧИСЛА
5	ТРЕТЬЯ
6	8 (последнее значение ключа – это третья степень числа 2)

Цифровая подпись или алгоритм RSA

Содержание темы урока

Внимание! Для изучения темы необходимо повторить основы программирования на Python (реализацию функций, ветвления, алгоритм Евклида).

Криптографическая система с открытым ключом RSA реализует ассиметричное шифрование и названа в честь ученых из Массачусетского технологического института: Рональда **Р**ивеста, Ади **Ш**амира и Леонарда **А**длемана.

Алгоритм состоит из нескольких этапов:

Этап I: генерация открытого и закрытого ключа шифрования.

Выберем два различных простых числа p и q.

$$p = 17, q = 13$$

Найдем произведение этих чисел n. Это будет один из открытых ключей абонента.

$$n = 17 \cdot 13 = 221$$

Найдем количество натуральных чисел, меньше числа n и взаимно простых с ним (функция Эйлера). Так как n раскладывается только на два простых делителя, а для простых чисел все числа, меньше их самих, с ним взаимнопросты, то: $\varphi(n) = (p-1)(q-1)$.

$$\varphi(n) = (17-1)(13-1) = 16 \cdot 12 = 192$$

В качестве открытого ключа a (открытой экспоненты) выберем число, взаимно простое с $\varphi(n)$ и меньше $\varphi(n)$.

$$a = 5$$

$$\text{НОД}\ (5, 192) = 1$$

Вычислим закрытый ключ α: $\alpha \cdot a \equiv 1\ (mod\ \varphi(n))$;

$$5 \cdot \alpha \equiv 1\ (mod\ 192).$$

Замечание: при нахождении неизвестного из сравнения $k \cdot x \equiv m\ (\text{mod}\ N)$ советуем найти такое целое x, при котором $k \cdot x$ равнялось одному из чисел $m + N, m + 2N, m + 3N$ и т. д. В разбираемом примере $5 \cdot \alpha = 193$? (Нет такого целого α.) А $5 \cdot \alpha = 385$? (Есть целое и оно равно 77.)

$$\alpha = 77.$$

Таким образом у всех абонентов есть свой открытый ключ $(a,\ n)$, передаваемый другим абонентам, и закрытый $(\alpha,\ n)$, который передавать не следует.

Абонент A	p_1 и q_1	n_A	a	α
Абонент B	p_2 и q_2	n_B	b	β
Абонент C	p_3 и q_3	n_C	c	γ
…	…	…	…	…

Нашим открытым ключом является пара (5, 221), а закрытым – пара (77, 221).

Этап II: передача зашифрованного сообщения.

Чтобы передать сообщение, необходимо знать открытый ключ адресата $(a,\ n_A)$. Сообщением является число m.

Пусть открытым ключом нашего собеседника является (5, 221), а передаваемое сообщение $m = 6$.

Зашифрованное сообщение m_1 получается после вычисления следующей операции:

$$m_1 \equiv m^a (mod\ n_A);$$
$$m_1 \equiv 6^5 (mod\ 221);$$
$$m_1 \equiv 41\ (mod\ 221).$$

Этап III: дешифрование полученного сообщения.

1. Дешифрование осуществляется с использованием закрытого ключа самим получателем сообщения следующим образом:

$$m_2 = m_1^{\alpha} \equiv m\ (mod\ n);$$

$m_2 = 41^{77}\ (mod\ 221)$ (прекрасная задача, не правда ли?).

Замечание: при нахождении степеней по модулю промежуточные результаты также преобразуйте по модулю – это облегчит задачу. И не забывайте про свойства степеней!

$$41^2 = 1681 = 134\ (mod\ 221)$$
$$41^3 = 134 \cdot 41 = 5494 = 190\ (mod\ 221)$$
$$41^5 = 41^2 \cdot 41^3 = 134 \cdot 190 = 25460 = 45\ (mod\ 221)$$
$$41^{10} = 41^5 \cdot 41^5 = 45 \cdot 45 = 2025 = 36\ (mod\ 221)$$
$$41^{20} = 41^{10} \cdot 41^{10} = 36 \cdot 36 = 1296 = 191\ (mod\ 221)$$
$$41^{40} = 41^{20} \cdot 41^{20} = 191 \cdot 191 = 36481 = 16\ (mod\ 221)$$

$$41^{70} = 41^{40} \cdot 41^{20} \cdot 40^{10} = 16 \cdot 191 \cdot 36 = 110\,016 = 179\ (mod\ 221)$$

$$41^{77} = 41^{70} \cdot 41^{5} \cdot 41^{2} = 179 \cdot 45 \cdot 134 = 1\,079\,370 = 6\ (mod\ 221)$$

В реальной жизни используют крайне большие простые числа (не 17 и 13, а много большие). Авторы алгоритма в качестве открытых ключей системы использовали 129-значное число n и $a = 9007$.

В 2010 г. ученые успешно расшифровали данные, зашифрованные с использованием ключа стандарта RSA длиной 768 бит и советовали не использовать ключи менее 1024 бита.

Поскольку шифрование RSA лучше всего подходит именно для шифрования чисел (причем больших, тогда сообщение придется разбивать на большие блоки) и, кроме того, вычислительно довольно долгое, то его применяют по такой схеме:

1. Выбирают алгоритм для «потокового» шифрования.
2. Создают временный «сеансовый» ключ для этого шифра
3. Сеансовый ключ шифруют с помощью RSA и передают.
4. На стороне получателя принимают сообщение, расшифровывают сеансовый ключ и потом расшифровывают потоковые данные. По окончании сеанса временный ключ аннулируется.

В нашем случае шифров немного, так что будем использовать вместе с RSAаффинное шифрование – с одной стороны более стойкое, чем шифры Цезаря и Виженера, а с другой – более вычислительно легкое, чем RSA.

Задание 1 (практическая работа)

1.1. Подготовьте программу для вычисления открытого и закрытого ключа, шифрования и дешифрования методом RSA на любом языке программирования.

1.2. Выберите и вычислите открытые и закрытые ключи для двух абонентов. Используйте числа *P* = 19, *Q* = 17 и минимальной экспонентой.

1.3. Для теста зашифруйте числа *A* = 19 и *B* = 7 для второго абонента с помощью чисел *P* = 19, *Q* = 17 и минимальной экспонентой.

Расшифруйте эти числа сами для проверки работы алгоритма.

1.4. Вы создали пару ключей для чисел *P* = 19, *Q* = 17 и выбрали экспоненту 5. Было получено сообщение:

282 166 ЯДТБИМЯЪБГФРГХРЪФРБНХСЕМГРЪМ

Расшифруйте его.

Решения и ответы:

1.1. Программа на языке Python для шифрования RSA.

```
def ExtendedEuclide (a,b):
    res = []
    if b == 0:
        res.append(a)
        res.append(1)
        res.append(0)
    else:
        x2=1
        x1=0
        y2=0
        y1=1
        while b > 0:
            q = a // b
            r = a - q*b
            x = x2 - q*x1
            y = y2-q*y1
            a = b
            b = r
            x2 = x1
            x1=x
            y2=y1
            y1 = y
        res.append(a)
        res.append(x2)
        res.append(y2)
    return res

def inverseMod (a,m):
    tmp = ExtendedEuclide(a,m)
    return (tmp[1]+m)%m

def powerMod(a,n,m):
    pw = 1
    k = a
    while n > 0:
        if n % 2 == 1:
            pw = (pw * k)%m
        k = (k*k) % m
        n = n // 2
    return pw
```

```
def selectExp (phi):
    ex = 2
    res = ExtendedEuclide(ex,phi)
    while res[0] != 1:
        ex = ex +1
        res = ExtendedEuclide(ex,phi)
    return ex

p=int(input('Число P ==>'))
q=int(input('Число Q ==>'))
n=p*q
phi = (p-1)*(q-1)
e = selectExp (phi)
d = inverseMod(e,phi)
print('Открытый ключ: (',e,',',n,')')
print('Закрытый ключ: (',d,',',n,')')

print('Расшифровано 282:',powerMod(282,d,n))
print('Расшифровано 166:',powerMod(166,d,n))
```

Программа на языке Python для расшифровки аффинного шифра:

```
def ExtendedEuclide (a,b):
    res = []
    if b == 0 :
        res.append(a)
        res.append(1)
        res.append(0)
    else:
        x2=1
        x1=0
        y2=0
        y1=1
        while b > 0:
            q = a // b
            r = a - q*b
            x = x2 - q*x1
            y = y2-q*y1
            a = b
            b = r
            x2 = x1
            x1=x
            y2=y1
```

```
                y1 = y
            res.append(a)
            res.append(x2)
            res.append(y2)
        return res

    alphabet = 'абвгдеежзийклмнопрстуфхцчшщъыьэюя'
    abclen = len(alphabet)
    abc = { alphabet[i]:i for i in range(abclen) }
    ka=int(input('Число A ключа аффиннного шифрования ==>'))
    kb=int(input('Число B ключа аффиннного шифрования ==>'))
    crypt =''.join(str(input('Строка для расшифровки
==>')).lower().split())

    euclide = ExtendedEuclide (ka,abclen)
    print ('Ключ расшифровки:(',euclide[1],',',kb,')')
    decrypt = ''.join(str(alphabet[(euclide[1]*(abc[crypt[i]]-
kb+abclen))%abclen]) for i in range(len(crypt)))
    print('Расшифровка:', decrypt)
```

1.2. Для чисел $P = 19$, $Q = 17$, $e = 5$:

Открытый ключ: (5,323). Закрытый ключ: (173,323).

1.3. 304 и 11.

1.4. Ключи: 28 и 13. Сообщение:

ПОЛИНАПРИВЕТВСТРЕТИМСЯЗАВТРА

Задание 2 (домашняя практическая работа)

2.1. Используя готовую программу из задания 1, выберите и вычислите открытые и закрытые ключи для двух абонентов. Используйте числа $P = 23$, $Q = 13$ и минимальной экспонентой.

2.2. Для теста зашифруйте числа $A = 13$ и $B = 11$ для второго абонента с помощью чисел $P = 23$, $Q = 13$ и минимальной экспонентой. Расшифруйте эти числа для проверки работы алгоритма сами.

2.3. Вы создали пару ключей для чисел $P = 23$, $Q = 13$ и выбрали экспоненту 5. Для вас получено сообщение:

234 189 ЗВШРКФБЬДЙЪДНЪБЙЪЬОНЮПКДЪБК

Что вам написали?

Ответы к заданию 2

2.1. Открытый ключ: (5,299). Закрытый ключ: (53,299).

2.2. 234, 189.

2.3. ОЛЬГАПРИВЕТВСТРЕТИМСЯЗАВТРА

В Приложении 1 к данному пособию представлены кейсы по основам криптографии. Приложение 2 содержит кейсы по социальной инженерии для изучения правовых основ информационной безопасности.

Содержание темы «Технология блокчейн», а также материал для проверки усвоения темы можно взять из учебного пособия: Самылкина Н.Н., Калинин И.А., Тарапата В.В. Салахова А.А. Информатика: 8–11-е классы: практикум. – М. : Просвещение, 2023. – 157с.: ил. – URL: https://lbz.ru/books/1171/

Методические рекомендации по изучению тематического раздела «Теоретические основы информатики»

Тематический раздел «Теоретические основы информатики» на углубленном уровне представлен в 10 классе следующими темами:

1. Представление информации в компьютере (19 ч).
2. Основы алгебры логики (14 ч).
3. Компьютерная арифметика (7 ч).

В 11 классе изучаются две темы:

4. Информация и информационные процессы (10 ч).
5. Моделирование (8 ч).

На изучение раздела в 10 классе выделяется 40 часов, в 11 классе – 18 часов.

В 11 классе предусмотрено резервное время 24 часа, которое рекомендуется использовать для расширения практической части данного тематического раздела.

В Рабочей программе предусмотрено 10 практических работ за 2 года обучения.

Предметные результаты на двух уровнях изучения данного раздела представлены в таблице 1.

При изучении темы **«Представление информации в компьютере»** (19 ч) получает развитие вопрос разделения двух видов сигналов – аналоговых и дискретных, который уже рассматривался упрощенно в основной школе. Сигнал является физическим способом передачи информации. *Сигнал* – это физическое воздействие, которое передается от одной физической системе к другой. *Информация* – это те изменения, которые произойдут под воздействием сигнала у его получателя, т. е. правильная интерпретация получателем сигнала позволит понять его смысл. Смысловое содержание сигнала, понятое получателем и есть информация. Таким образом, можно сказать, что информация не содержится в сигнале, но под его воздействием возникает у получателя.

Чаще всего при рассмотрении сигналов, с помощью которых мы получаем некоторую информацию, выделяют собственно сигнал, т. е. полезное изменение величины и ***помехи*** – изменения той же величины, которые мешают информацию извлечь.

Стандартная схема *источник – канал – приемник*, рассмотренная в основной школе, немного усложняется, добавляются кодирующее и декодирующее устройства и учитываются помехи, возникающие в канале связи (рис. 7).

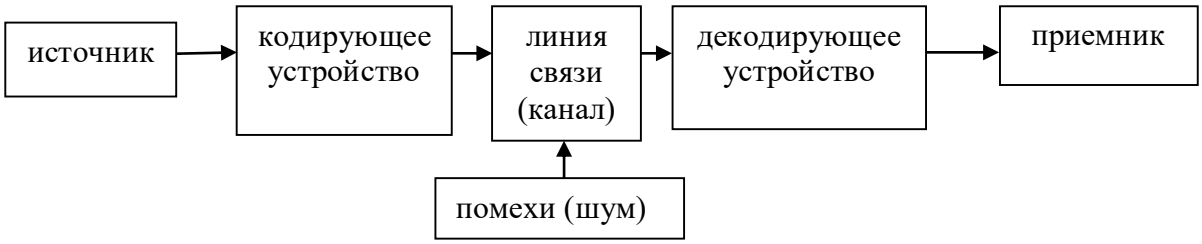


Рис. 7. Схема процесса передачи информации

Как известно из курса физики, в реальной среде любое изменение распространяется волнообразно, а скорость распространения волны будет зависеть от самой среды. В информатике иначе среду называют ***каналом связи***. Фиксируемое изменение (сигнал) в виде некоторого параметра, также будет изменяться волнообразно. Для описания сигнала рассматривают изменение величины, анализируя его амплитуду и частоту. Сигналы в реальном мире имеют максимальную амплитуду в момент «старта», далее из-за потерь энергии в среде она будет уменьшаться. Таким образом, если мы рассматриваем сигнал как средство передачи (или получения информации), то для нас важнее как именно среда изменяется, т. е. с какой частотой.

С точки зрения анализа сигнала важнее всего понять, в какие именно частоты было вложено больше всего энергии (т. е. что именно создавал передатчик). Распределение энергии сигнала по частотам изменения называется ***спектром сигнала***.

При создании различных средств связи со времени их изобретения существовала проблема качественной передачи информации, т. е. избавления от шумов, возникающих в канале передачи.

В технических системах, которые опираются на прием и передачу сигнала, сам сигнал можно использовать двумя способами:

1) когда изменение одного сигнала соответствует некоторому изменению другого непрерывно, т. е. при этом способе, возможно с помощью сложной схемы, сигналы (как волнообразные изменения) превращаются в другие сигналы без потери промежуточных значений (звуковые колебания среды в электрические импульсы, например, когда кто-то поет с микрофоном и это

записывается на какой-либо носитель). Поскольку фактически *сигналам ставятся в соответствие их аналоги* в другой среде, то способ и сами сигналы в таких системах называются ***аналоговыми***;

2) если вместо операций с сигналом оперируют отдельными результатами замера величины, то вместо гладкой кривой сигнала появится некоторый прерывистый ряд значений, полученных в результате измерений. Такой способ обработки сигнала (и сам сигнал, ориентированный на такую обработку) называется ***дискретным (цифровым)***.

На первый взгляд кажется, что второй способ значительно хуже – он явно приводит к потере точности и требует определения процедуры замеров и обратных преобразований. Но на самом деле этот способ обладает рядом ключевых преимуществ:

1) дискретный сигнал значительно более удобен при обработке и при хранении, т. е. для него можно быстро строить сложные схемы обработки на базе математических методов;

2) для обработки дискретных сигналов можно создавать универсальные (т. е. мало зависящие от природы исходного сигнала) устройства;

3) дискретный сигнал гораздо устойчивее к помехам.

Для практического использования важно, каким образом (или при каких условиях) можно восстановить исходный аналоговый сигнал из дискретного.

Было сначала определено экспериментально (Найквист), а позже и доказано теоретически (Котельников), что для этого *частота дискретизации (частота замеров) должна быть в два раза больше, чем спектр дискретизируемого сигнала*.

То есть:

$$fp \leq 2B,$$

где fp – количество пульсаций (замеров); B – частота сигнала в герцах.

Это утверждение называется теоремой Котельникова–Найквиста. Например, для воспроизведения звука (при ориентации на человеческий слух) частота дискретизации составляет примерно 44 КГц, поскольку максимальная фиксируемая ухом частота составляет примерно 22 Кгц.

В своей теореме В.А. Котельников доказал, при каких условиях можно восстановить исходный сигнал из дискретного, что позволило создать современные технические и программные средства работы со звуком. Приоритет

нашего соотечественника в исследованиях по данной тематике признан мировым научным сообществом. В целом обсуждение вклада российских ученых в развитие информатики является элементом патриотического воспитания и расширяет кругозор старшеклассников.

Теорема была предложена и доказана В. А. Котельниковым в 1933 г. в работе «О пропускной способности эфира и проволоки в электросвязи». Можно использовать математически более точную формулировку теоремы: «Любую функцию $f(t)$, состоящую из частот от 0 до f_c, можно непрерывно передавать с любой точностью при помощи чисел, следующих друг за другом через $1/(2f_c)$ секунд».

Безусловно, такой сложный материал как теорема Котельникова–Найквиста не следует предлагать обучающимся с полной математической выкладкой всех преобразований, как в вузе. Это и в вузе изучают не все, а только те, кто будет профессионально заниматься связью и телекоммуникациями. Всем обучающимся необходимо понимать, что информация передается и принимается техническими устройствами в виде сигналов, и что аналоговые сигналы можно преобразовать в цифровые, а цифровые – в аналоговые.

Обратите внимание на то, что объяснение сущности сигналов отличается от данного в основной школе, становиться точнее. Объясняются и преимущества цифровых сигналов.

Для преобразования аналогового сигнала в цифровой необходимо последовательно выполнить над ним три операции: дискретизацию, квантование и кодирование.

Дискретизация – это получение мгновенных значений сигнала (отсчетов, замеров, пульсаций) через определенные промежутки времени (т. е. с определенной частотой – частотой дискретизации) (рис. 8).

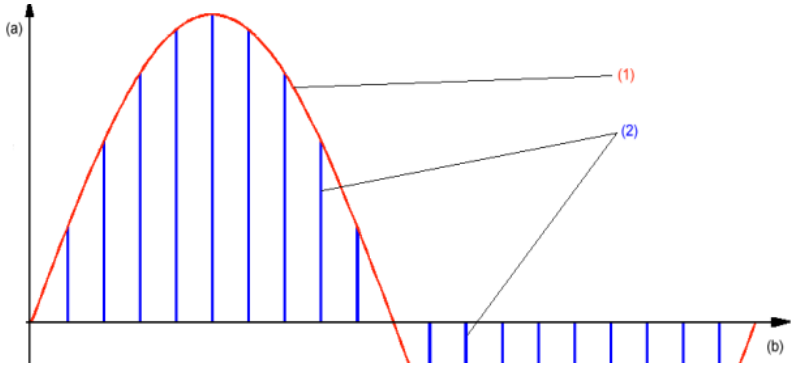


Рис. 8. Дискретизация: 1 – сигнал; 2 – отсчеты

Квантование – это «округление» полученных мгновенных значений до ближайших заранее заданных уровней (с определенной ошибкой квантования). Например, если имеется 8 уровней с шагом 2: 0, 2, 4, 6, 8, и т. д., а некоторые мгновенные значения равны 3,6; 7,1; 2; 0,5; 1,8, то они будут округлены до 4, 8, 2, 0, 2 соответственно (рис. 9).

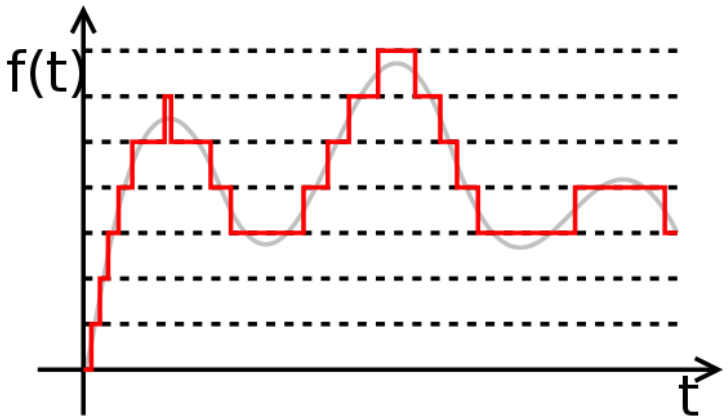


Рис. 9. Квантование

Кодирование – это представление значений полученных уровней в виде кода (например, двоичного). Квантование и кодирование практически всегда являются неотъемлемыми частями друг друга.

Рассмотрим, как преобразуется человеческая речь в цифровой сигнал при использовании цифровой телефонии.

Наша речь занимает полосу частот приблизительно 80–10 000 Гц. Это достаточно широкий диапазон частот. Для нормальной разборчивой речи достаточно полосы частот в 300–3400 Гц, т. е. верхняя граница составляет 3,4 кГц. Все, что выше 3,4 кГц «срезается» фильтром, чтобы избежать помех в будущем. Согласно теореме Котельникова, частота дискретизации для представления аналогового сигнала, ограниченного по спектру (помним о фильтре), в виде отсчетов должна превышать удвоенную верхнюю частоту сигнала. Для простоты расчетов, а также некоторого запаса, верхняя граница округляется до 4 кГц. Таким образом, частота дискретизации в нашем случае должна быть равной 8 кГц.

Квантование в цифровой телефонии 256-уровневое и неравномерное. Неравномерность квантования связана с тем, что шаг квантования (расстояние между соседними уровнями в единицах измерения характеристики аналогового сигнала, которая квантуется; в данном случае – напряжение сигнала в вольтах) для малых амплитуд (т. е. тихого звука) выбирается минимальным, для средних –

бóльшим и для больших – самым большим. Это сделано для того, чтобы повысить точность передачи сигналов с низкой амплитудой. 256 уровней квантования можно «уместить» в одно 8-разрядное двоичное число, таким образом, один отсчет представляется в виде 8-разрядной кодовой комбинации. Все 256 уровней делятся на две группы: положительные и отрицательные. Для положительных сигналов первый бит в кодовой комбинации равен 1, для отрицательных – 0. Каждая группа делится на 8 сегментов. Обратите внимание, что в пределах одного сегмента шаг квантования неизменный, в то время как от сегмента к сегменту он меняется, увеличиваясь с возрастанием номера сегмента. Под номер сегмента отводятся следующие 3 бита. Последние 4 бита занимает номер уровня в сегменте, всего этих уровней 16. Итого имеем: 2 группы × 8 сегментов × 16 уровней = 256 уровней.

Например, число 10010101 представляет собой положительный сигнал (1) в 1-м сегменте (001) с уровнем 5 (0101).

Теперь можно посчитать скорость полученного цифрового сигнала:

8000 отсчетов/с × 8 бит/отсчет = 64 000 бит/с = 64 Кбит/с.

Данные сигналы являются простейшими сигналами в цифровой телефонии. Для их передачи до сих пор используются основные цифровые каналы со скоростью 64 Кбит/с.

Рекомендуется рассматривать процесс передачи информации подробно, на примере звуковой информации и добиться его понимания. Впоследствии обучающимся будут понятны другие современные форматы передачи и хранения звуковой информации. Далее, необходимо подобрать задачный материал

на кодирование звука, поскольку эта тема входит в государственную итоговую аттестацию. Предварительно повторить формулу расчета.

Чтобы найти информационный объем звукового файла I, нужно перемножить между собой длительность звукового файла t, частоту дискретизации f, количество каналов k и глубину кодирования i:

$$I = t \cdot f \cdot k \cdot i,$$

где I – информационный объем звукового файла;

t – время (с);

f – частота дискретизации (Гц);

k – количество каналов (моно – 1, стерео – 2);

i – глубина кодирования (бит).

Задача из демонстрационного варианта ЕГЭ (2023)

Музыкальный фрагмент был записан в формате моно, оцифрован и сохранен в виде файла без использования сжатия данных. Размер полученного файла –28 Мбайт. Затем тот же музыкальный фрагмент был записан повторно в формате стерео (двухканальная запись) и оцифрован с разрешением в 3,5 раза выше и частотой дискретизации в 2 раза меньше, чем в первый раз. Сжатие данных не производилось. Укажите размер полученного при повторной записи файла в Мбайт. В ответе запишите только целое число, единицу измерения писать не нужно.

Решение:

Дано:	*Формулы:*	*Решение:*
$I_0 = 28$ Мб	$f_1 = f_0/2$	$I_1 = 2 \cdot \boldsymbol{i}_0 \cdot 3{,}5 \cdot f_0/2 \cdot t = 3{,}5 \cdot I_0 = 3{,}5 \cdot 28$
$k_0 = 1$	$\boldsymbol{i}_1 = \boldsymbol{i}_0 \cdot 3{,}5$	
$k_1 = 2$	$I_1 = k_1 \cdot i_1 \cdot f_1 \cdot t$	
$I_1 = ?$		

Ответ: $I_1 = 98$.

Переход к ***единицам измерения количества информации*** лучше всего осуществить с рассмотрения сути минимального сигнала и его фиксации. Особенно это важно, если в основной школе этот материал не рассматривался. Практика показала, что при наличии двух часов на изучение информатики в основной школе изучать два подхода к измерению информации (вероятностный и алфавитный) лучше в 7 классе основного общего образования. Но в переходный период место этого материала определяет учитель.

Передача информации в системе «источник – приемник», которую мы описали выше, с точки зрения получателя информации (приемника) – это процесс, в котором меняется представление получателя о сообщении. До начала ему неизвестно ничего и получение любого из них – равновероятно. Получение сообщения сделает вероятность одного из них равной единице (т. е. того, которое мы получили), а остальных – нулю. Минимальное количество сообщений, которое мы можем рассматривать в начале процесса, ровно двум. Поскольку оно будет одно, то его не нужно передавать – так все ясно. Таким образом, ***минимальное количество информации – это сообщение, снижающее неопределенность знаний о некотором событии в два раза.*** Простейшим этот случай является и при передаче информации – такое сообщение будет

наименьшим, которое можно передать в сигнале. Это просто сам факт возникновения сигнала (или его отсутствие – как второе событие).

Именно такое сообщение, которое может иметь всего два значения, и есть минимальная единица информации – ***один бит***. Больше в минимальный сигнал просто не умещается.

Если у нас есть некоторое длинное сообщение, то изначально, до момента начала передачи, получатель не знает о нем ничего и может предполагать все что угодно. На самом деле, поскольку сообщение имеет ограниченную длину, то и общее количество вариантов ограничено. Получив первое минимальное сообщение («да» или «нет»), он сокращает количество вариантов в 2 раза.

Таким образом, если мы передаем сообщение минимальными порциями, то ***количество информации*** мы будем определять как количество битов. Количество сообщений (как вариантов событий для выбора) может быть любым.

Основание 2 оказалось очень удобным. Количество минимальных событий для выбора – 2, условно можно закодировать «да» – 1, «нет» – 0. Компьютер работает с двоичным кодом 0,1. Считают 0 – 1 бит, 1 – 1 бит. Полбита в компьютерной памяти неосуществимо. Для вывода формулы, заполним таблицу взаимосвязи двух величин.

Таблица 10

Бит (i)	0	1	2	3	4	5	6	7	8	9	10
Количество сообщений или событий (N)	1	2	4	8	16	32	64	128	256	512	1024

Получим формулу зависимости двух величин из таблицы: $\boldsymbol{N = 2^i}$. Это основная формула в информатике. Для неравновероятных событий можно вычислить вероятность $\boldsymbol{p = 1/N}$.

Речь идет о сообщении как наборе битов, не зависящим от способа передачи, – именно этот набор и будет содержанием сигналов (т. е. колебаний среды).

Очевидно, что любого пользователя системы передачи информации интересует не сам набор битов, а то, что он представляет, какую информацию позволяет извлечь (т. н. семантический аспект). Откуда и появляются задачи:

– представить в виде годных к обработке наборов битов сообщения, имеющие иную природу (иллюстрации, числа, звук, текст и т. д.);

– разработать методы хранения, обработки и передачи таких сообщений;

– разработать способы демонстрации таких сообщений в годном для восприятия виде.

Преимуществом описанного подхода к измерению информации можно считать то, что он позволяет использовать математические приемы подсчета информационного сообщения. С точки зрения на информацию как «на снятую неопределенность», количество информации зависит от вероятности получения того или иного сообщения. Причем, чем больше вероятность события, тем меньшее количество информации содержится в сообщении. Иначе этот подход называют **вероятностным** или **содержательным**.

Существует другой способ измерения информации, называемый **алфавитным,** который рассматривался в основной школе.

В основе алфавитного подхода лежит представление о сообщении как тексте, то есть о наборе букв (символов) из заранее заданного конечного множества букв, которое и называется ***алфавитом***. Количество букв в алфавите называется ***мощностью алфавита.***

Подход к измерению количества информации на основе этого подхода состоит в подсчете количества символов в сообщении. При этом информационная емкость каждого знака зависит от количества знаков в используемом алфавите. Таким образом, можно написать формулу и вычислить информационную емкость каждой буквы в битах. Действительно, общее количество букв – N, таким образом, число i такое, что:

$$\mathbf{2}^{i} = \boldsymbol{N},$$

где i будет количеством битов, необходимых для представления одной буквы.

Поскольку число битов – целое, то на самом деле выбирают количество битов минимально достаточное – равное или большее на 1.

Остается подсчитать количество букв в измеряемом тексте сообщения k. Если число букв в тексте – k, а мощность алфавита – N, i – количество битов для представления буквы, то информационный объем всего сообщения:

$$I = k \cdot i.$$

То есть можно представить текст в виде набора битов – номеров букв алфавита.

Набор битов тоже можно считать текстом – с алфавитом мощностью 2. Из написанного выше следует, что это – минимальная возможная мощность

алфавита. Так можно показать эквивалентность этих подходов – то, что описано как текст, может быть описано как набор битов. Алфавитный подход к измерению информации подходит для понимания и применения различных способов кодирования и сжатия (упаковки) информации техническими устройствами и передаче данных по каналам связи.

Таким образом, обучающимся показана математическая суть бита, как минимальной единицы информации, связь с алфавитным подходом к измерению информации. При решении задач необходимо демонстрировать как меняются по смыслу компоненты основной формулы $2^i = N$, когда мы работаем с различной информацией (звуковой, текстовой, графической).

Кодирование текстовой и графической информации

С помощью i бит можно закодировать не более N различных символов или цветов:

$$N = 2^i,$$

где N – количество символов или цветов;

i – глубина цвета (информационный объем одного пикселя) или емкость одного символа.

Чтобы найти информационный объем текста или изображения I, нужно умножить количество символов или пикселей k на число бит i для хранения одного символа или пикселя:

$$I = k \cdot i,$$

где I – информационный объем текста или изображения;

k – количество символов (букв в строке × строк) или пикселей (высота изображения × ширина изображения в пикселях).

Чтобы найти информационный объем переданной информации I, нужно умножить время передачи t на пропускную способность канала связи v:

$$I = t \cdot v,$$

где I – информационный объем переданной информации;

t – время (с);

v – пропускная способность канала/скорость передачи данных (бит/с).

Для решения задач по теме необходимо ориентироваться на материалы государственной итоговой аттестации (https://fipi.ru/).

Задача из демонстрационного варианта ЕГЭ (2023)

Для хранения произвольного растрового изображения размером 128 × 320 пикселей отведено 20 Кбайт памяти без учета размера заголовка файла. Для кодирования цвета каждого пикселя используется одинаковое количество бит, коды пикселей записываются в файл один за другим без промежутков. Какое максимальное количество цветов можно использовать в изображении?

Решение лучше оформлять следующим образом:

Дано:
$k = 128 \cdot 320$
$I = 20$ Кб
$i = ?$
Найти: $N = ?$

Формулы:
$N = 2^i$
$I = k \cdot i$
$i = I/k$

Решение:

$I = 20$ Кбайт $= 20 \cdot 1024$ байт $= 20 \cdot 1024 \cdot 8$ бит;

$$i = \frac{20 \cdot 1024 \cdot 8}{128 \cdot 320} = \frac{5 \cdot 2^2 \cdot 2^{10} \cdot 2^3}{2^7 \cdot 2^6 \cdot 5} = \frac{5 \cdot 2^{15}}{5 \cdot 2^{13}} = 2^2 =$$

$= 4$ бита;

$N = 2^4 = 16$

Ответ: $N = 16$.

Хорошо известная тема «Системы счисления» также рассматривается в контексте темы «Представление информации в компьютере». На уровне основного общего образования обучающиеся научились переводить из разных систем счисления в десятичную систему и обратно, освоили двоичную арифметику и рассмотрели связи систем счисления с основанием 2 для рационального и быстрого перевода из двоичной в восьмеричную и шестнадцатеричную системы счисления и обратно. Для углубленного уровня изучения информатики на уровне среднего общего образования понадобится знание этой темы, так же как и основ логики для изучения в 11 классе алгоритмов кодирования Хаффмана и Хемминга, а также для программирования и анализа данных. Системы счисления изучаются в основной школе, где навыки перевода в разные системы счисления отрабатываются до автоматизма, но практически не удается осмыслить и добиться полного понимания общей формулы развернутой записи чисел. Поэтому фактически завершается изучение темы системы счисления в старшей школе рассмотрением общей формулы развернутой записи числа, решением более сложных аналитических задач, где знание этой формулы помогает быстрее решить задачу. Задачи на системы счисления присутствуют в материалах итоговой аттестации по информатике.

Развернутой формой записи числа называется запись вид:

$$A_n = \pm\left(a_{m-1}n^{m-1} + a_{m-2}n^{m-2} + \ldots + a_0n^0 + a_{-1}n^{-1} + a_{-2}n^{-2} + \ldots + a_{-k}n^{-k}\right).$$

Здесь A_n – само число, n – основание системы счисления, a_i – цифры данной системы счисления**,** m – количество разрядов целой части числа, k – количество разрядов дробной части числа.

Так, например,

$$765{,}345_{10} = 7 \cdot 10^2 + 6 \cdot 10^1 + 5 \cdot 10^0 + 3 \cdot 10^{-1} + 4 \cdot 10^{-2} + 5 \cdot 10^{-3},$$

$$1011{,}1_2 = 1 \cdot 2^3 + 0 + 1 \cdot 2^1 + 1 \cdot 2^0 + 0 + 1 \cdot 2^{-2} = 11{,}25_{10}$$

$$10FC_{16} = 1 \cdot 16^3 + 0 + F(15) \cdot 16^1 + C(12) \cdot 16^0 = 300_{10}$$

При записи числа в развернутой форме надо обращать внимание обучающихся на правильное использование оснований систем счисления. Далее можно произвести подсчет (он осуществляется по правилам десятичного счета) и получить десятичный эквивалент числа. Правило перевода из произвольной системы счисления числа в десятичную систему очень просто запомнить: *разворачиваем и считаем*. Для отработки понимания темы решаем задачи с разнообразным условием, не только в формулировке материалов итоговой аттестации.

Пример 1. Заданы четыре числа в различных системах счисления: $A = 232_4$, $B = 2F_{16}$, $C = 53_8$, $D=101100_2$. Необходимо из максимального числа вычесть минимальное. Получите результат в десятичной системе счисления.

Решение: переведем все числа в десятичную систему счисления:

$A = 2 \cdot 4^2 + 3 \cdot 4^1 + 2 = 46_{10}$, $\quad B = 1 \cdot 16^1 + 15 = 47_{10}$,

$C = 5 \cdot 8^1 + 3 = 43_{10}$, $\quad D = 1 \cdot 2^5 + 1 \cdot 2^3 + 1 \cdot 2^2 = 44_{10}$.

Видим, что максимальным является число B, а минимальным – C. Разность между максимальным и минимальным равна $B - C = 47 - 43 = 4$.

Ответ: 4.

Пример 2. Укажите через запятую в порядке возрастания все десятичные числа, не превосходящие 17, запись которых в системе счисления с основанием 3 оканчивается на две одинаковые цифры.

Решение: Напоминаем, что в троичной системе счисления присутствуют цифры 0, 1, 2. Поскольку алфавит троичной системы счисления небольшой, можно решить простым перебором. Последовательно выпишем числа, оканчивающиеся на две одинаковые цифры, и переведем в десятичную систему счисления:

$11_3 = 1 \cdot 3^1 + 1 = 4_{10}$,

$22_3 = 2 \cdot 3^1 + 2 = 8_{10}$,

$100_3 = 1 \cdot 3^2 + 0 \cdot 3^1 + 0 = 9 + 0 + 0 = 9_{10}$,

$111_3 = 1 \cdot 3^2 + 1 \cdot 3^1 + 1 = 9 + 3 + 1 = 13_{10}$,

$122_3 = 1 \cdot 3^2 + 2 \cdot 3^1 + 2 = 9 + 6 + 2 = 17_{10}$.

Понятно, что следующее число – 200_3, оно будет уже больше 17.

Ответ: 4, 8, 9, 17.

Пример 3. В системе счисления с неизвестным основанием число 129 записывается в виде 1004. Найдите это основание.

Решение: Обозначим неизвестное основание через *x* и запишем число в развернутом виде: $129_{10} = 1 \cdot x^3 + 0 \cdot x^2 + 0 \cdot x^1 + 4 = x^3 + 4$. Решая уравнение $x^3 + 4 = 129$, получаем $x = 5$.

Ответ: 5.

Пример 4. Найдите основание позиционной системы счисления *x*, в которой будет справедливо следующее равенство: $13_x + 31_x = 110_x$.

Решение: Запишем все числа, входящие в выражение в развернутом виде, и вычислим результат:

$1 \cdot x^1 + 3 + 3 \cdot x^1 + 1 = 1 \cdot x^2 + 1 \cdot x^1 + 0$. Решая уравнение $4x + 4 = x^2 + x$, получаем $x = 4$ ($x = -1$ является посторонним корнем).

Ответ: 4.

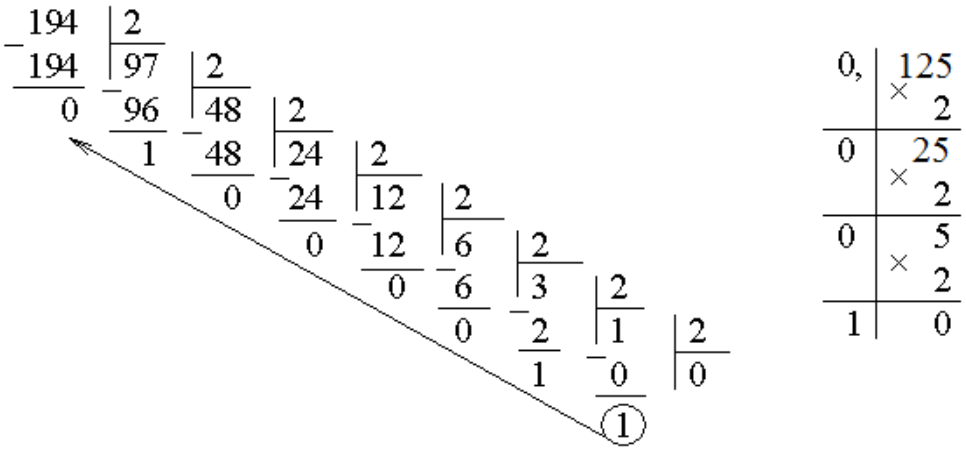
Пример 5. Найдите наименьшее основание позиционной системы счисления *Y*, при котором $225_x = 14_y$.

Решение: Запишем формулу перевода $2 \cdot x^2 + 2 \cdot x^1 + 5 = 1 \cdot y^1 + 4$, или $y = 2x^2 + 2x + 1$. Заметим, что в числе с основанием *x*, присутствует цифра 5, значит, $x \geq 6$. Подставляя вместо *x* число 6, находим наименьшее из возможных $y = 2 \cdot 6^2 + 2 \cdot 6 + 1 = 85$.

Ответ: 85.

Пример 6. Вычислите сколько единиц в двоичной записи десятичного числа 194,125?

Решение: Переведем в двоичную систему счисления числа 194 и 0,5:



В результате получаем $194{,}125_{10} = 11000010{,}001_2$. В двоичной записи числа 194,125 четыре единицы.

Пример 7. Найдите количество пятизначных чисел в восьмеричной системе счисления, в записи которых присутствует только одна цифра 6, учитывая, что никакая нечетная цифра не стоит рядом с цифрой 6.

Решение:

1. Рассмотрим элементы нашего алфавита: {0, 1, 2, 3, 4, 5, 6, 7}.

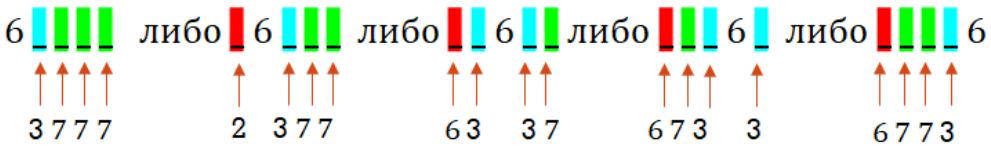
   Из них четные: 0, 2, 4, **6 – всего 3 без цифры 6.**

   Из них нечетные: 1,3,5,7 - **всего 4.**

2. Рассмотрим все случаи расположения цифры 6 в числе:

   6 _ _ _ _ либо _ 6 _ _ _ либо _ _ 6 _ _ либо _ _ _ 6 _ либо _

3. Применим правила комбинаторики в каждом случае:



4. Посчитаем: $3 \cdot 7^3 + 2 \cdot 3 \cdot 7^2 + 6 \cdot 3^2 \cdot 7 + 6 \cdot 3^2 \cdot 7 + 6 \cdot 3 \cdot 7^2 = 2961$

Ответ: 2961.

Связь между двоичной, восьмеричной и шестнадцатеричной системами счисления, а также двоичную арифметику лучше изучать в основной школе, а в старшей только повторить, если это необходимо для подготовки к итоговой аттестации.

В 11 классе вызывают трудности алгоритмы кодирования Хаффмана и Хемминга, если материал не адаптирован для старшеклассников. Это

обязательный материал для среднего общего образования, поскольку результаты изучения входят в предметные результаты по информатике. Рассмотрим содержание этих тем, на изучение которых достаточно двух уроков. Эти два вида кодирования рассматриваются на примере кодирования текста.

Неравномерное кодирование. Алгоритм кодирования Хаффмана

Содержание темы урока

Самый распространенный способ кодирования текста – сопоставление каждой букве некоторого натурального числа, которое потом переводится в двоичный код с помощью двоичной системы счисления. Поскольку один из главных критериев при разработке кода – компактность (от этого зависит время и надежность передачи текста), то способ кодирования должен обеспечивать минимальную длину кода.

Самое известное сопоставление такого рода – ***таблица кодировки***. Таблица кодировки представляет собой сопоставление каждой букве фиксированного числа. В большинстве таблиц кодировки появление буквы считается равновероятным событием, а поэтому длина кода для всех букв одинакова. Такой код называется ***равномерным***. Примером равномерного кода является таблица кодировки ASCII (7 бит на букву, т. е. до 128 символов в таблице. Восьмой бит в расширенной таблице кодировки отводится на представление национальных алфавитов и псевдографики).

Как известно из курса информатики основной школы, равномерное кодирование текста сохраняет его статистические свойства. Проще говоря, какие бы символы мы не использовали для записи текста, их количество останется тем же самым (одни символы перешли в другие). Именно поэтому простое перекодирование (подстановка) не является стойким шифром – даже если мы никому не сообщали нашу таблицу перекодирования – ее нетрудно восстановить по достаточно длинной части текста – просто подсчитав частоты отдельных символов и соотнеся их с известными частотами для букв языка.

Очевидно, что будет выгодно при записи текста отводить для часто встречающихся букв меньше бит, а для редко встречающихся – больше. Такой код будет ***неравномерным***. Самый известный пример такого кодирования –

кодирование Хаффмана[1]. Это кодирование применяется при сохранении или передаче текстов для его ***сжатия***.

Рассмотрим алгоритм кодирования Хаффмана

В этом алгоритме каждая буква будет обозначаться последовательностью 0 и 1, которую в алгоритме называют ***словом***. Вместо вероятности используем частоту появления каждой буквы (хотя частота не совсем точно ей соответствует, но удобна при подсчете и не искажает смысл алгоритма).

Общее описание алгоритма построения таблицы кодирования Хаффмана для конкретного текста выглядит так:

1. Строится таблица частот каждой буквы, затем таблица упорядочивается по убыванию вероятности использования каждой буквы.

2. Выбираются две наименее вероятных буквы и объединяются в одну «псевдобукву». Ее частота (и вероятность) – сумма двух частот.

3. Менее вероятная исходная буква обозначается 1, более вероятная – 0. Эти значения добавляются к их кодам (для первого шага – буквы получат коды 0 и 1, а дальше коды будут постепенно расти).

4. Операция повторяется, начиная с шага два до тех пор, пока не останется одна «метабуква» с вероятностью 1 (частотой равной количеству букв).

Результатом будет своеобразное дерево, показывающее, как были объединены буквы. На каждом шаге объединения добавлялись к кодам букв (словам) биты, и к самым редким буквам (с которых начали) их добавили больше всего.

Конечно, кодовая таблица зависит от исходного текста, в котором были подсчитаны буквы. Поэтому либо используют заранее подсчитанную таблицу для типичного текста (например, так поступают при сжатии изображений), либо таблицу кодов пишут перед упакованным сообщением.

Например, имеется некоторый текст, в котором встречаются буквы:

Буква	Частота
a	45
b	43
d	16
c	12
e	9
f	5

[1] Сходным образом работал и более старый код Шеннона – Фано, но в некоторых случаях он менее эффективен и уже не используется.

Наше дерево будет выглядеть как на рисунке 10.

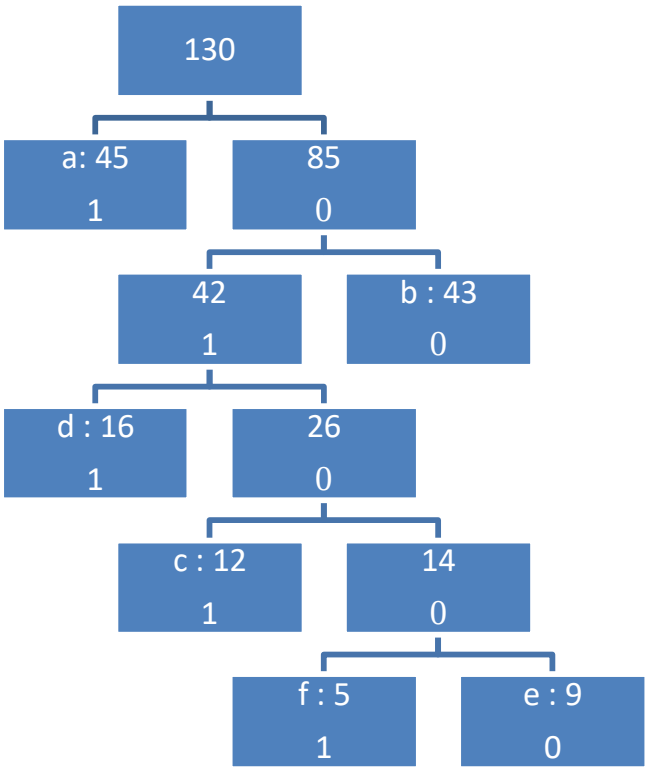


Рис. 10. Кодирование по Хаффману (дерево)

Таблица 11

Буква	**Код**	**Частота**	**Битов обычного кода**	**Битов кода Хаффмана**
a	1	45	360	45
b	0	43	344	43
d	11	16	128	32
c	101	12	96	36
e	0001	9	72	36
f	1001	5	40	25
Всего			*1040*	*217*

Выгода использования кодирования по Хаффману очевидна, если сравнить два последних столбца таблицы.

Задание. Постройте таблицу кодирования Хаффмана для текста, в котором встречаются следующие буквы:

Буква	Частота
A	57
B	49
C	32
E	19
F	9
R	4

Решение: Строим дерево как в предыдущей задаче (рис. 11).

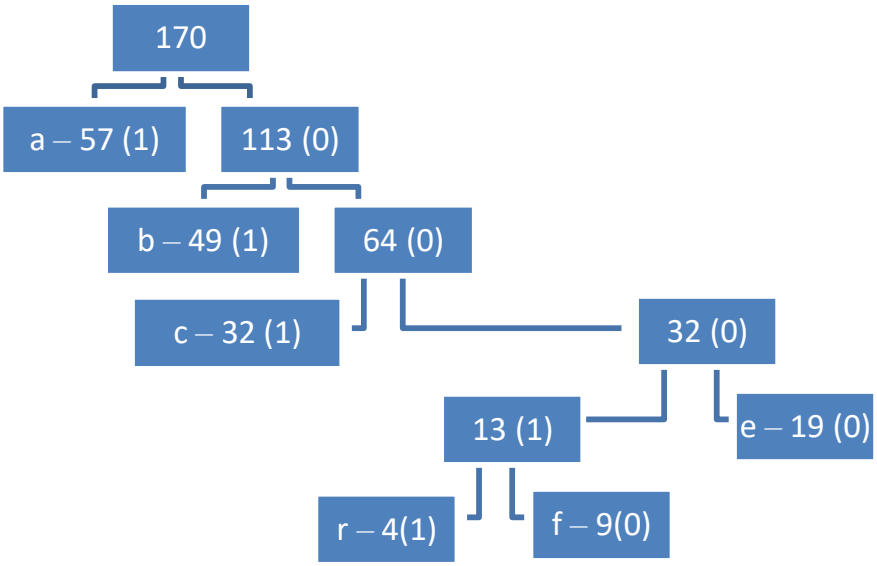


Рис. 11. Построение дерева

Начиная от корня, записываем коды букв.

Буква	Код
a	1
b	01
c	001
e	0000
r	00011
f	00010

Для иллюстрации подсчитаем повышение эффективности при передаче такого сообщения (табл. 11).

Таблица 11

Буква	**Частота**	**Количество бит**	
		в обычном коде	**в коде Хаффмана**
a	57	456	57
b	49	392	98
c	32	256	96
e	19	152	76
r	9	72	45
f	4	32	20
Всего		*1360*	*392*

После изучения алгоритма Хаффмана рекомендуем рассмотреть различные алгоритмы сжатия или упаковки файлов. Несмотря на преимущества неравномерного префиксного кодирования Хаффмана, у него есть недостатки. Вместе с данными необходимо сохранять дерево, иначе распаковка данных будет невозможна. К настоящему времени придумано множество методов сжатия и реализовано в виде алгоритмов и программных средств – архиваторов. Все методы сжатия делят на две большие группы: алгоритмы сжатия без потери информации и с потерями. Лучше всего алгоритмы сжатия объяснять на примере обработки изображений. Первая группа методов применяется тогда, когда исходный файл с изображением нужно восстановить без каких-либо искажений: например, при подготовке качественной печати, в научной графике и т. д.

Самый простой пример алгоритма сжатия без потерь – метод RLE (англ. *Run-length encoding, RLE*), кодирование путем учета длин повторов. Принцип его работы очень прост – записывается цвет точки и количество его повторов до другого цвета.

Если длинных последовательностей много, то результат будет очевиден, если повторов нет – размер изображения даже вырастет за счет добавления значений повторов.

В современных системах используют чаще всего более сложные алгоритмы, построенные на поиске повторяющихся последовательностей, а не просто цветов. Эффективны такие методы при сжатии чертежей, контурных рисунков, текста и т. п.

Очевидно, что для изображений с малым количеством повторов (практически все фотографии) такие методы будут неэффективны – цвета рядом

хоть чуть-чуть, но различаются. Поэтому метод RLE в настоящее время редко используется.

Вторая группа методов получила название сжатия с потерями: изображение сжимается этими методами с искажением, то есть потерей части информации. Несмотря на формальные потери, в реальных условиях человек воспримет это искажение как допустимое (иногда вообще не заметит).

В таких методах сжатия, например изображений, используется несколько приемов:

1. Некоторые компоненты цвета человек воспринимает хуже других, поэтому можно отвести на них меньше битов – все равно разница будет малозаметна.

2. Если разница в яркости между соседними пикселями невелика, то их цвета можно приблизить друг к другу – разница будет не очень заметна.

В наиболее популярном алгоритме сжатия изображений с потерями – JPEG изображение разбивается на квадраты 8 × 8 пикселей, внутри этих квадратов выполняется несколько преобразований.

При выполнении сжатия с потерями, как правило, можно задать коэффициент сжатия – и чем выше этот коэффициент, тем меньше будет сжатое изображение, но тем хуже будет его качество при восстановлении. Сильно сжатое изображение будет «рассыпаться» на квадраты

Коды алгоритмов сжатия достаточно сложные, их не рекомендуется рассматривать в школе. Предметные результаты сформулированы на уровне понимания принципов работы простых алгоритмов сжатия данных, в материалах государственной итоговой аттестации нет вычислительных задач по теме. Поэтому можно ограничиться решением задач на кодирование с выполнением условия Фано.

Контроль и восстановление после ошибок. Код Хемминга

Содержание темы урока

Одна из характеристик любой линии связи – это вероятность возникновения ошибок при передаче сигнала. Ошибка может возникнуть в результате помех, нарушения синхронизации, сбоев оборудования – причин много.

С учетом того что вероятность возникновения ошибки существует всегда, необходимы методы, которые позволяют такие ошибки выявлять и по возможности корректировать их. При большом объеме передаваемых данных (или совершаемых операций) ошибки возникать будут обязательно.

Наиболее частый случай повреждения данных – повреждение одного из битов в слове. Возникновение ошибки в двух и более битах одного слова маловероятны. Самым распространенным методом поиска и исправления ошибок при передаче данных по каналам связи является специальный метод кодирования данных, получивший название **код Хэмминга**.

Суть этого метод состоит в том, что для выявления и коррекции ошибок мы добавляем к битам основного сообщения контрольные биты. Сообщение делится на отдельные слова фиксированной длины, состоящие из двух частей – биты сообщения и контрольные биты.

Позиции контрольных битов – степени числа 2: 1, 2, 4, 8 и т. д.

Строгое определение кода Хэмминга требует, чтобы соотношение количества информационных и контрольных битов было оптимальным, т. е. максимум информационных и минимум контрольных. Используем вариант, в котором этот принцип соблюдается: 7 битов информационных и 4 контрольных.

При формировании кода рассчитывают ***контрольную сумму*** – т. е. последовательность битов, сравнивая с которой будет возможно обнаруживать и исправлять ошибки.

Контрольная сумма формируется как результат операции XOR («исключающее ИЛИ») над позициями ненулевых битов.

Например, необходимо передать букву g. Ее код в таблице ASCII – 67_{16}, т. е. 1100111_2.

Разместим информационные биты в кодированном слове, начиная с младшего разряда.

Позиции	**Биты**	**Пояснение**
1	*	Под контрольный бит
2	*	Под контрольный бит
3	1	С младшего разряда начинаем
4	*	Под контрольный бит
5	1	
6	1	
7	0	
8	*	Под контрольный бит
9	0	
10	1	
11	1	

Контрольные биты мы пока не рассчитали. Для их расчета вычислим сумму, выполнив побитовую операцию XOR с позициями 3,5,6,10 и 11 (позиции ненулевых битов, для их представления нужно 4 двоичных разряда. Количество необходимых разрядов определяем по максимальному числу). Результат – 1 (0001_2).

Позиция 3	0	0	1	1
Позиция 5	0	1	0	1
Позиция 6	0	1	1	0
Позиция 10	1	0	1	0
Позиция 11	1	0	1	1
Результат XOR	0	0	0	1

Размещаем полученные значения в таблицу и получаем итоговый код.

Позиция	Бит
1	1
2	0
3	1
4	0
5	1
6	1
7	0
8	0
9	0
10	1
11	1

Если принимающая сторона подсчитывает контрольную сумму для всего полученного слова, то при правильной передаче результат будет 0.

Контрольная сумма рассчитывается как побитовая операция XOR для ненулевых позиций итоговой таблицы, т. е. для позиций 1, 3, 5, 6, 10 и 11.

Предположим, при передаче возникла ошибка, и мы получили в пятой позиции бита вместо 1 – 0. При расчете контрольной суммы позиция 5 будет нулевой и в подсчете не участвует, тогда вместо 0, в контрольной сумме получим: 101, т. е. 5 – **номер неверно переданного бита**. Остается его инвертировать и слово будет исправлено.

Конечно, для случая, в котором повреждены два бита результат исправления по такому правилу будет не просто неверным, он будет даже хуже, чем просто ошибка. Но рассчитаем вероятность такого совпадения: пусть вероятность ошибки в 1 бите слова – 10^{-5} (в среднем одна ошибка на 10 тыс. бит, т. е. довольно часто). Тогда вероятность совпадения двух таких событий в одном слове – 10^{-10}, т. е. одно такое слово на 10 млрд бит, т. е. такое событие можно считать маловероятным, что нам и требовалось.

Еще раз напомним, что строгий код Хэмминга использует другие соотношения, а приведенный пример иллюстрирует саму процедуру

Задание. Рассчитайте биты сообщения с кодом коррекции Хэмминга для передачи сообщения из двух букв – i и n (в ASCII-коде – 89 и 94, по 8 бит на код).

Решение: В исходном сообщении 2 буквы, по 8 бит – итого 16 бит. Добавляя необходимое количество битов (в позициях 1, 2, 4, 8, 16), получим (16 + 5) длину сообщения с контрольными битами – 21 бит.

Запишем их в таблице, как и в прошлый раз, обозначив пока значения контрольных бит знаком «*».

Поскольку битов больше, чем в прошлой задаче, расположим таблицу горизонтально. В первой строке – номера битов, начиная с минимального разряда, во второй – их значения.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
*	*	1	*	0	0	1	*	1	0	1	0	0	1	1	*	1	1	0	1	0

Выполним XOR с позициями ненулевых битов (3, 7, 9, 11, 14, 15, 17, 18, 20) получим 5 контрольных битов.

Позиция 3	0	0	0	1	1
Позиция 7	0	0	1	1	1
Позиция 9	0	1	0	0	1
Позиция 11	0	1	0	1	1
Позиция 14	0	1	1	1	0
Позиция 15	0	1	1	1	1
Позиция 17	1	0	0	0	1
Позиция 18	1	0	0	1	0
Позиция 20	1	0	1	0	0
Результат XOR	1	0	0	0	0

Запишем их в контрольные биты.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
0	0	1	0	0	0	1	0	1	0	1	0	0	1	1	1	1	1	0	1	0

Проверим результат, рассчитав контрольную сумму при точной передаче. Фактически, мы можем рассчитать ее, просто выполнив XOR результата расчета ($10\ 000_2 = 16_{10}$) с позиции нового ненулевого бита – 16 (остальные контрольные биты – 0).

16 XOR 16 = 0

Поскольку в материалах государственной итоговой аттестации нет заданий на кодирование Хаффмана и Хемминга, можно ограничиться только разбором нескольких заданий и небольшой проверочной работой.

По тематическому разделу **«Теоретические основы информатики»** в 11 классе сложными темами являются темы **«Моделирование»** и **«Искусственный интеллект».** В углубленном курсе информатики теоретический и практический материал по этим темам не должен дублировать то, что было рассмотрено в основной школе. С помощью достаточно серьезной теоретической части, посвященной системному подходу в моделировании, следует выйти на динамические модели и исследовать несколько моделей в среде имитационного моделирования. При необходимости использовать для этого резервные часы.

Моделирование востребовано во многих отраслях профессиональной и научной деятельности человека и считается одним из методов научного познания. Деятельность по моделированию является основной для информатики, поскольку в информатике как научной области изучаются общие методы и средства создания и использования *информационных моделей.*

В общем виде, под *информационной моделью понимают описание объекта моделирования на каком-либо формализованном языке* с тем, чтобы исследовать в дальнейшем полученную модель с использованием компьютера. Иногда используют термин «компьютерная модель», поскольку в информатике основной инструмент исследования моделей – компьютер. Для информатики важно, каким образом модель описана, в зависимости от этого выбирают способы исследования модели.

Создание моделей позволяет:

1) описывать характеристики реальных объектов, их взаимосвязь;

2) исследовать поведение объекта как поведение его характеристик, исключив несущественные факторы;

3) предсказать поведение моделируемого объекта, причем предсказать его и для всех возможных вариантов развития событий.

В целом ряде случаев модель – единственная возможность провести исследование и изучение объектов без нанесения существенного вреда (например, устойчивость здания явно безопаснее проверять на математической модели, без попыток его построить) или неприемлемых затрат.

Например, построение математической модели здания или механизма позволяет заранее узнать поведение исходного объекта в критических условиях, например, состояние здания атомной электростанции после того, как в нее врежется самолет. Или, что более реалистично, поведение высотного здания в условиях сильного ветра. Очевидно, что проверять такие вещи на практике нерационально.

Таким образом, повторяются цели моделирования и понятие информационной модели, акцентируется внимание обучающихся на то, что моделирование – научный метод познания, а компьютерное моделирование – его частный случай.

На подготовку любой модели существенно повлияет то, каким является исходный, реальный объект моделирования. На уровне среднего общего

образования в моделировании следует выйти на исследование динамических моделей, ведь реальные процессы в жизни не статичны.

Один из наиболее универсальных и активно использующихся в моделировании подходов получил название ***«системный подход».*** В его основе лежит представление о реальном мире как о взаимодействующих комплексах объектов. В зависимости от целей, можно выделить некоторые интересующие объекты, определить их связи и создать модель этого комплекса – изучаемого крупного объекта (системы).

Для этого необходимо, с одной стороны, выделить «границы» объекта (системы), а с другой – определить его внутреннее устройство и его взаимодействие с внешним миром, т. е. всеми остальными объектами, которые в создаваемой модели будут рассматриваться как условия.

Человеку трудно воспринимать (и исследовать) сложные объекты как что-то неразделимое, да еще и непрерывно развивающееся. Поэтому для создания таких моделей необходимо определить совокупность *отдельных* частей, которые можно исследовать *дискретно*. При этом, конечно, необходимо сохранить понимание того, что все-таки отдельные части связаны, а в реальности процессы идут непрерывно и чаще всего параллельно.

Для описания таких взаимосвязанных объектов в некоторой внешней среде вводят термин «система». В литературе существует более 40 разных определений этого термина. Можно взять за основу достаточно простое определение. ***Система*** *– это совокупность связанных между собой в единое целое отдельных частей (элементарных, неделимых объектов, процессов, явлений), причем в виде целого, обладающую свойствами, которыми элементы по отдельности не обладают.*

Отдельная, неделимая часть в этом случае называется ***элементом системы***. Неделимость элемента системы на самом деле означает, что для конкретного случая внутренняя структура элемента не имеет значения, не оказывает влияния на модель в целом.

Тем не менее часто при создании (или описании) элемента системы нам приходится и его рассматривать как совокупность связанных объектов. В этом случае элемент называется ***подсистемой***.

Система также включает в себя компоненты, осуществляющие взаимодействие между элементами системы, а также между системой и окружающей средой. Такие компоненты называются ***связями***.

Связи – фактически и есть то, что отличает систему от просто набора отдельных частей. Как и для самого понятия модели, для понятия связи существует масса определений.

Фактически, ***связь*** *– это компонент системы, осуществляющий взаимодействие одного компонента с другим компонентом (компонента и среды, компонента и системы в целом).*

При моделировании важно представлять, какие связи придется отражать в будущей модели. Возможно, как обычно, рассмотреть несколько классификаций.

По направлению связи: исходящие, входящие, двусторонние.

По содержанию: материальные, энергетические, информационные.

По порядку:

1. *Связи первого порядка* – связи, определяющее поведение и структуру системы.

2. *Связи второго порядка* – дополнительные, увеличивающие синергетический эффект, но не необходимые.

3. *Связи третьего порядка* – случайные, излишние или противоречивые связи.

Относительно устойчивая система связей между элементами системы называется ***структурой системы***. Понятие устойчивости требует формализации, но достаточно считать, что устойчивой структура будет, если в течение исследования/использования модели в ней не произойдет изменений, существенных для сохранения подобия модели реальности.

Нужно отметить, что система – это по определению комплекс, выделяющийся из окружающей действительности. Выделяя системы в реальном мире, можно найти в них общие черты «типовых» систем. Например, выделяют достаточно «крупные»: биологические, общественные и технические системы. Предложите обучающимся привести свои примеры, и будет заметно, насколько различаются представления каждого из них об окружающих его системах.

В соответствии с положениями системного подхода – выделяя из общей среды собственно систему, следует описать ее на некотором формальном языке. Это могут быть коды, графы и пр. После этого изучение реального объекта может быть заменено (по крайней мере, частично) изучением модели. При реализации системного подхода важно: выделить возможно более «мелкие» элементы

системы, связи между ними, описать их на некотором формальном языке и «собрать» из выявленных компонентов модель реального объекта.

Процесс выделения компонентов и связей носит название ***системного анализа***, а сбора модели из компонентов – ***синтеза***.

В зависимости от того как система описывается в модели, можно предложить несколько классификаций. В частности, система может быть описана как:

1. ***Модель «черного ящика».*** В этом случае игнорируется внутреннее устройство системы, она описана в виде набора входов и набора выходов. Соответствие устанавливается между определенной комбинацией входных и выходных параметров.

2. ***Модель состава системы.*** При таком подходе система описывается как набор параметров входящих в нее элементов, игнорируются связи между ними.

3. ***Структурная модель.*** В этом случае в описании используется не только описание элементов системы, но и связей между ними. Связи – это не только формальное указание, но и самостоятельный компонент модели, оказывающий влияние на ее поведение.

4. ***Динамическая модель.*** В этом случае описываются структура и состав системы, а также учитываются изменения системы и ее параметров во времени. Поскольку целью моделирования в этом случае является изучение не статического состояния, а динамики развития, то и модель называется динамической.

Несмотря на то что разнообразие систем очень велико и системный подход один из наиболее универсальных, можно отметить несколько закономерностей, характерных для всех видов систем.

Эмерджентность (от англ. emergence – появление) – появление в системе новых качеств, которые у ее составляющих по отдельности отсутствовали. Например, набор деталей вертолета, каждый из которых сам по себе летать не может.

Целостность – изменение в одном элементе системы вызывает изменения в других ее элементах. Пример: недостаток смазки в движущихся частях поезда приведет к снижению срока их службы и замедлению движения.

Как следствие, в системах возникают *побочные эффекты* – внося изменения в часть системы, и на те ее части, которые не должны были

затрагиваться. Таким образом, в пределах системы нельзя ограничить воздействие одной частью.

В зависимости от характера связей между элементами системы (и их наличия) система на изменения может реагировать по-разному.

Два крайних варианта поведения:

1. ***Аддитивность*** – так ведут себя объекты, в которых элементы не связаны между собой. Поведение такого объекта – сумма поведений его частей. Например, площадь ромба равна сумме площадей прямоугольных треугольников, из которых может состоять ромб; цветовая схема, часто используемая для представления изображения, – цвет как сумма трех цветов (точнее, светимости).

2. ***Синергизм*** – такой объект ведет себя как единое целое, все элементы связаны и в результате эффект не суммируется, а умножается. Например, два куска радиоактивного материала в соединении дают выделение энергии, превосходящее излучение энергии простого суммирования отдельных кусков или объединенное действие двух лекарственных препаратов может быть более сильным, чем сумма действий этих двух лекарств при их раздельном приеме.

Любая реальная система, конечно, будет где-то между полной аддитивностью и синергизмом. Но это условное положение редко остается постоянным с течением времени, поэтому для описания изменения этих свойств в системе вводят понятия:

– ***факторизации*** – т. е. прогрессирующего деления системы на части (элементы или подсистемы);

– ***систематизации*** – росту количества и жесткости связи элементов системы.

Одним из важных свойств систем является их ***иерархичность*** – любая система может быть представлена в виде соподчиненных объектов, такая система называется ***иерархией***. В иерархии более высокий уровень объединяет все объекты нижнего уровня и оказывает на них направляющее воздействие. Например, для архитектора дом со всеми коммуникациями – одна система, а для электротехника системой является общая схема проложенных электрических кабелей, а сам дом – это внешняя среда.

Иерархичность – свойство, которое очень часто используется при анализе систем как способ описать порядок взаимодействий в системе или выявить причины того или иного поведения.

Перед рассмотрением имитационного моделирования рекомендуется вернуться к математическому моделированию на примерах моделей популяционной динамики. Достаточно часто в моделировании встречаются задачи, сводящиеся к исследованию поведения системы, состоящей из большого количестве несложных объектов. Например, моделей для предсказания численности популяции каких-либо животных учеными было создано довольно много. Самая первая – это модель Фибоначчи (размножение кроликов, в котором общая численность – это два текущих поколения), модель Мальтуса (неограниченного экспоненциального роста), модель Ферхюльста (рост, ограниченный ресурсами) и, наконец, предложенная в начале XX в. модель Вольтера-Лотки («хищник – жертва»). Этот материал присутствует в учебниках информатики углубленного уровня для 10–11 классов. Математическое моделирование на примерах моделей популяционной динамики рассматривается без вывода формул. Формулы, вернее усложнение модели за счет добавления новых параметров, влияющее на численность популяции, просто анализируются обучающимися. Совместно с обучающимися приходим к выводу о невозможности в приемлемое время исследовать математическую модель и использовать результаты в практических целях. Возникают идеи о том, что хорошо бы процесс исследования как-то автоматизировать.

Итак, если описать взаимодействие элементов системы математически сложно или невозможно, то можно воспользоваться общим методом под названием ***имитационное моделирование***.

Специально созданные компьютерные программы позволяют имитировать поведение системы из множества простых элементов, моделируя их поведение и взаимодействие, рассматривая при этом результаты для всей системы в целом.

Такие модели называются ***имитационными***, т. е. имитирующими поведение системы. Применяя их, можно изучить поведение системы, не имея общих аналитических соотношений (без сложной математики), проследить поведение системы в динамике (т. е. на каждом шаге процесса), исследовать поведение системы, в которой возможны случайные вариации поведения объектов.

Существует несколько видов имитационных моделей.

1. ***Дискретно-событийные модели.*** Такие модели описывают поведение системы как набор последовательных событий. Например, так можно описать

рост известного растения (процесс состоит из хорошо известных стадий), работу предприятия (его работа описана инструкциями и регламентами). В такой системе описываются *состояния* (например, ожидание поступления на склад автосервиса запасной части) и *события*, т. е. изменения состояния.

2. ***Агентные модели.*** Такие модели построены в виде набора взаимодействующих отдельных объектов (агентов), каждый из которых имеет какие-то заданные правила поведения и на основе поступающих сведений принимает решение об их применении. Например, такой агент может описывать поведение особи в стаде, человека в толпе или очереди и т. д. Поведение всей системы будет складываться как результат взаимодействия агентов, но спрогнозировать его (например, поведение толпы на выходе из здания) математическим соотношением трудно, слишком много параметров необходимо учесть.

3. ***Модели системной динамики.*** Этот метод предназначен для исследования поведения достаточно сложных систем с большим количеством обратных связей и зависимых параметров (например, крупного города, большого производства, демографические модели и т. д.). При таком виде моделирования систему представляют в виде взаимодействующих объектов. Каждый объект на самом деле может быть очень крупной системой, но при моделировании мы считаем его единым целым, пропускающим через себя, формирующим или поглощающим потоки разного рода – финансовые, материальные и т. п.

Для работы с имитационными моделями можно использовать бесплатную образовательную версию AnyLogic: https://www.anylogic.ru/downloads/ (для имеющейся на компьютере операционной системы, возможно, это будут личные ноутбуки обучающихся). Преимущества данной среды над другими:

1) единственная среда, реализующая все три вида моделей: агентное, дискретно-событийное и системную динамику;

2) возможность работать без сложной математики и программирования;

3) бесплатная русифицированная образовательная версия – Personal Learning Edition, рассчитана на тех, кто только начинает знакомство с данной средой;

4) большое количество справочного материала по AnyLogic (видеоуроков, литературы, готовых моделей).

На начальной странице программы AnyLogic размещено восемь учебных пособий на все три вида имитационных моделей. Выполнение этих работ поможет освоить среду и станет основой для проектных исследовательских работ. Идеи работ возникают при разборе контекста заданий, изменении исходных параметров, желании решить свою проблему современным инструментом имитационного моделирования. Построение имитационных моделей систем из разных сфер жизнедеятельности общества (модель работы школьной столовой, техноцентра, аэропорта, бизнес-центра, распространения эпидемий и т. д.) послужит хорошим примером к повышению мотивации обучающихся в освоении процесса моделирования.

Практические работы и идеи проектов в заданиях к этим работам предложены в электронной версии учебного пособия: И.А. Калинин, Н.Н. Самылкина, П.В. Бочаров «Информатика. Углубленный уровень: задачник-практикум для 10–11 классов» [4]. Для выполнения работ использовались более ранние версии продукта, но тематика работ социально значима для старшеклассников. Практикум есть в свободном доступе по ссылке: https://files.lbz.ru/authors/informatika/8/kal-zp10-11.pdf. В нем представлены четыре практические работы: «Безопасность в школе», «Оптимизация работы поликлиники», «Эпидемия», «Работа сотовой компании», которые могут составить основу индивидуальных проектов.

Умение строить и исследовать сложные модели реальных производственных и социальных процессов в среде имитационного моделирования является одним из основных практических результатов при изучении данной темы. При внимательном анализе примерной программы можно увидеть, что темы моделирования, программирования, искусственного интеллекта и информационных технологий пересекаются или интегрируются, а возможность переставлять темы в рамках одного года по-прежнему остается у учителя. Это следует учитывать при комплектации уроков теории и практики. Зачастую, изучая одну тему, параллельно рассматриваются еще несколько, особенно в части практической реализации на Python или в каком-то программном пакете. Учителю при планировании следует указывать какие дополнительно темы были охвачены и за счет каких разделов и каких практических занятий это сделано, выделив эти моменты в примечания.

В связи с понятием *системы* возникает проблема *управления системой*: выделение управляющей и управляемой системы и рассмотрения их

взаимодействий, т. е. решение управленческих задач. Причем использование различных моделей среды имитационного моделирования в основном для задач прогнозирования позволяет сразу и оптимизировать функциональные процессы и минимизировать риски от субъективных управленческих решений.

Управлением называют особый вид деятельности, направленный на поддержание системы (т. е. сохранение ее элементов и связей) в продуктивном состоянии, т. е. в состоянии, достигающем некоторых заранее определенных целей, путем организации поведения и взаимодействия элементов между собой и внешним миром. Для живого организма эта цель – выживание и размножение, для технической системы – выработка некоторого продукта и т. п.

Эта деятельность включает в себя выработку решений (т. е. нужных воздействий), их передачу, выполнение и получение результатов (т. е. формирование представления о том, что получилось).

В моделировании и создании таких систем выделяют две основных части: субъект управления (что управляет) и объект управления (чем управляют). Эти две части связаны между собой и поведение объекта управления в большей части определяется субъектом. Нужно заметить, что в такой системе необходима минимум одна связь, т. е. компонент взаимодействия субъекта с объектом.

Связь эта информационная, т. е. фактически сообщения, управляя поведением объекта, становятся ***командами***. При этом ни природа связи, ни вид сообщений существенного значения не имеют. Такая связь в управляемой системе называется *прямой связью*.

Часто предусмотрена и *обратная связь*, т. е. сообщения о результатах исполнения команд от объекта субъекту. Эта связь, очевидно, тоже информационная. Системы без обратной связи называют *разомкнутыми*, а системы с обратной связью – *замкнутыми*.

Такой подход универсален: нам не требуется для описания процесса оговаривать природу и способ связи объекта и субъекта, он позволяет моделировать и изучать процессы управления в самых разных системах – технических, биологических, социальных.

Разработка технических систем автоматического управления с обратной связью стала причиной возникновения целого научного направления – ***кибернетики***, в котором самые разные системы с управлением рассматриваются с помощью такого подхода. Основное положение кибернетики гласит, что

управление – *процесс целиком информационный* (поскольку воздействие, оценка результатов и выработка новых команд – это процесс обработки и передачи некоторой информации) и протекает он по общим принципам в любых управляемых системах.

Системы управления – одно из самых старых и распространенных применений компьютеров[1]. Фактически современные компьютеры очень часто используются как универсальные субъекты управления, которые можно стандартными средствами связать с любыми объектами – от периферийных устройств до станков и сложных механизмов.

Использование этого подхода дает возможность создавать самые разные системы, управляющие сложными процессами или помогающие такое управление организовать. Поскольку, как мы уже указали, управление – информационный процесс, то с этим подходом мы получаем возможность создавать системы управляемые с минимальным вмешательством человека, построив достаточно универсальную модель восприятия, анализа его содержания, синтеза и выработки решения.

Итоговой целью деятельности является создание для возможно более широкой предметной области такой системы, в которой место человека как элемента, принимающего решения, занимает «информационный автомат», т. е. воспроизводимый технический объект.

Создание такого «универсального устройства принятия решений», включая и анализ окружающего мира и определение целей, и поиск средств их достижения – **сильного искусственного интеллекта**, одна из классических задач информатики, не имеющих решения до сих пор.

Появление устройств, выполняющих (и выполняющих очень эффективно) операции и действия, ране доступные только людям, заставило задуматься: что еще и каким образом может быть автоматизировано? Как на самом деле надо принимать решения? Как действует человеческое мышление? Причем ответ на эти вопросы должен быть не отвлеченным описанием, а позволяющим действительно решить группу технических задач, т. е. фактически задачи моделирования разных аспектов деятельности и мышления человека.

При решении этих задач выделилось два сообщающихся направления: разработка и применение автоматических систем (в данном случае – специально

[1] Автоматические системы управления – один из источников появления универсальных вычислительных машин.

спроектированных технических комплексов), т.е. систем, которые на некоторых участках заменяют деятельность человека, и разработка и применение систем вспомогательных (автоматизированных), предназначенных для решения задач вместе с человеком (информационные системы с интеллектуальными алгоритмами).

Таким образом, обучающимся раскрываются взаимосвязи темы динамического моделирования, функционирования сложных систем, управления этими системами и роль искусственного интеллекта в этих процессах.

Современными примерами применения систем с использованием **слабого** (т. е. пока не универсального) **искусственного интеллекта** являются различные системы распознавания и сегментации изображений – применяемые и при контроле продукции, и в системах обеспечения безопасности, и в системах диагностики заболеваний, распознавание текста на иллюстрациях и в устной речи, синтез речи с заданными характеристиками и т.д.

Активно обсуждаемыми в настоящий момент являются генеративные системы, позволяющие создавать информационные объекты по их описанию. Например, это системы автоматизированного перевода, системы реферирования, вопросно-ответные системы, генерация изображений по описанию и прочие аналогичные интеллектуальные информационные системы.

Активное развитие таких систем и их сравнение с результатами «человеческой» деятельности – хорошая тема для обсуждения с обучающимися. Фактически, работы в этой области начались практически одновременно с появлением первых компьютеров. Несмотря на то что «машинный разум» до сих пор не создан, во время исследований было получено много важных результатов, обогативших и науку, и технологию.

Но история этого научного направления такова, что до сих пор нет даже единого определения основного понятия. Сейчас есть множество разных подходов к изучению и понимаю искусственного интеллекта (ИИ). Наиболее важным является выделение двух областей.

- **Искусственный интеллект как наука** – наука о проектировании сознания, изучении и моделировании человеческого мышления, вычислимом поиске смысла в данных.
- **Искусственный интеллект как технология** – это проектирование систем, позволяющих обрабатывать данные интеллектуально (включая

самообучение, получение нетривиальных данных, построение теорий с помощью математической логики и т. д.), а также интеллектуальные алгоритмы**.**

Искусственный интеллект продолжает постоянно развиваться как научная и одновременно прикладная область. В 2020 г. дочерняя компания Google Deep Mind создала модель, интеллектуальные алгоритмы которой предсказываются сворачивание белков, что ранее считалось одной из 125 величайших задач, стоящих перед человечеством. Во время пандемии искусственный интеллект научили определять заражение вирусом COVID-19 раньше проявляемых явных признаков с помощью мобильного приложения. В то же время компания Alibaba в Китае создала систему диагностирования вируса на основе распределенных вычислений и машинного обучения, а затем представила программу Alipay Health Code, на основе больших данных регулирующая доступ зараженных лиц к общественным местам, что способствовало локализации очагов заражения. Пандемия повлияла и на внедрение и развитие интеллектуальных сервисов, связанных с командной работой и с автоматизацией процессов на производстве.

Сегодня мы чаще всего в контексте искусственного интеллекта говорим о машинном обучении (Machine Learning), науке о данных (Data Science) и глубинном анализе данных (Data Mining).

Искусственный интеллект и его применение давно перестали быть достоянием секретных лабораторий. В настоящее время в гаджеты и компьютеры встраиваются специальные сопроцессоры с искусственным интеллектом, в том числе в смартфонах, планшетах, колонках, носимых гаджетах, маршрутизаторах и сетевом оборудовании и т. д. В последнее время, примерно в течение десяти лет, наблюдается значительный рост стартапов, связанных с искусственным интеллектом и разработкой продуктов для образования. По данным HolonIQ, было два пика возникновения таких стартапов: 2013–2014 гг. (основная тема – контроль знаний и управление образовательной средой) и 2017–2018 гг. (основная тема – прокторинг и корпоративное обучение), характеризующиеся разной тематической направленностью. Область постоянно трансформируется в зависимости от потребностей пользователей. Сегодня мы наблюдаем рост проектов, связанных с анализом больших данных, формирующих цифровой след обучающегося и преподавателя. Проекты, связанные с машинным обучением,

в основном направлены на персонализацию обучения (включая составление индивидуального образовательного маршрута).

В Рабочей программе по информатике углубленного уровня терминология, связанная с искусственным интеллектом, присутствует практически в каждом тематическом разделе. В разделе «Теоретические основы информатики» по теме «Моделирование» предполагается некий обзор сервисов, содержащих элементы искусственного интеллекта, который вкратце приведен выше. В таблице 1 можно увидеть предметное требование, связанное с умением классифицировать задачи анализа данных, но в тематическом планировании Рабочей программы по этому тематическому разделу не выделены темы практических работ и время ни в 10, ни в 11 классе. При этом теоретический материал по теме «Анализ данных» в объеме 8 часов предлагается изучать в тематическом разделе «Информационные технологии», но все практические задачи предлагается решать пока в табличном процессоре. Таким образом, учителю информатики дается некоторое время на освоение языка Python со специализированными библиотеками для решения задач искусственного интеллекта. Тем педагогам, кто не испытывает затруднений с программированием, следует использовать следующий учебный материал: И.А. Калинин, Н.Н. Самылкина, А.А. Салахова. Искусственный интеллект. 10–11 классы: учебное пособие. – М.: Просвещение, 2023. – 144 с.) [5]. Издание можно найти по ссылке: https://shop.prosv.ru/iskusstvennyj-intellekt--10-11-klassy21811

Вернемся к этой теме в разделе «Информационные технологии».

Методические рекомендации по изучению тематического раздела «Алгоритмы и программирование»

Тематический раздел «Алгоритмы и программирование» на углубленном уровне представлен в 10 классе следующими темами:

1. Введение в программирование (16 ч).
2. Вспомогательные алгоритмы (8 ч).
3. Численные методы (5 ч).
4. Алгоритмы обработки символьных данных (5 ч).
5. Алгоритмы обработки массивов (10 ч).

В 11 классе предлагается изучать следующие укрупненные темы:

1. Элементы теории алгоритмов (6 ч).
2. Алгоритмы и структуры данных (28 ч).
3. Основы объектно-ориентированного программирования (16 ч).

На изучение раздела в 10 классе выделяется 44 часа, в 11 классе – 50 часов.

В ФПР по информатике в 10 классе предусмотрено 22 практические работы, в 11 классе – 17 практических работ. По факту каждый урок по данному тематическому разделу можно считать практическим, поскольку изучаемые алгоритмы реализуются на выбранном языке программирования и требуют отладки и анализа полученных результатов.

Предметные результаты на двух уровнях изучения данного раздела представлены в таблице 1.

Программирование в настоящее время является самой важной грамотностью, определяющей успешность самореализации каждого в повседневной цифровой реальности и успешность будущей профессиональной деятельности выпускников. Именно поэтому изучению программирования уделяется такое внимание в различных государственных программах подготовки кадров для цифровой экономики и нормативных документах на всех уровнях образования. В ФПР по информатике углубленного уровня на программирование выделено 94 часа за два года обучения при резерве учебного времени 42 часа, которое также можно использовать частично или полностью на программирование. Следует обратить внимание, что фактически все темы курса информатики интегрируются с темой программирования. Это видно и по материалам государственной итоговой аттестации выпускников 11 класса, где большинство задач решаются программированием и часто это

самое оптимальное из предлагаемых способов решения. Как во все времена изучения информатики, за учителем остается выбор языка программирования. Список рекомендуемых языков достаточен для достижения предметных результатов по информатике углубленного уровня. С выбором языка программирования связана последовательность изучения тем данного раздела и возможность использования выбранного языка программирования для поддержки изучения других разделов курса информатики углубленного уровня.

Исходя из того что язык программирования должен быть применен при решении задач на ЕГЭ и соревнованиях олимпиадного уровня, рассмотрим особенности рекомендованных для этих задач языков.

Язык программирования Java в настоящее время активно используется при разработке программного обеспечения для различных мобильных устройств и встраиваемых систем (в силу своей кроссплатформенности).

При его выборе следует учесть, что для его использования необходимо будет провести достаточно сложную и комплексную подготовку рабочего места, включающую в себя установку необходимых программных средств (базовая платформа средств разработки не содержит), дополнительных библиотек и их настройку. При этом требования к аппаратному обеспечению (особенно оперативной памяти) будут повышены.

Также следует понимать, что даже простая программа будет представлять собой специфическое приложение и содержать целый ряд отсылок к непростым для понимания концепциям в профессиональном программировании, в особенности базовой для языка концепции объектно-ориентированной разработки.

В отличие от Java, язык C++ появился и разрабатывается как универсальный, компилируемый язык. При использовании языка C++ следует помнить, что язык развивается достаточно давно, имеет целый ряд серьезно отличающихся друг от друга стандартов. Язык ориентирован на разработку крупных и сложных приложений, поэтому многие из его особенностей и даже базовых средств также требуют понимания достаточно непростых концепций в профессиональном программировании, которые при решении обычных школьных задач не будут востребованы. Также следует помнить, что, несмотря на развитую стандартную библиотеку (STL), решение прикладных задач

для слабо подготовленного пользователя будет существенно затруднено объемом и сложностью кода, который потребуется для обращения даже уже разработанным библиотекам (код на языке C++, как правило, в 2–3 раза больше, чем код на «скриптовом» адаптированном языке, что компенсируется для профессионалов универсальностью и скоростью работы готовых программ).

Отказ от сложных для понимания конструкций (например, библиотек шаблонов) приведет к тому, что язык превратится в диалект C, причем, как и в случае Java, со странно выглядящими артефактами в программах.

Язык C# разработан корпорацией Microsoft для разработки программного обеспечения под ОС Windows. Его основной рекомендованной средой разработки является фирменная среда Visual Studio, и ориентирован он на разработку программ, опирающихся на фирменную же исполняющую среду CLR.

С учетом отказа основного разработчика от работы с организациями в Российской Федерации в современных условиях его использование представляется невозможным. Кроме того, как и Java, язык ориентирован на разработку именно приложений – что влечет за собой усложнение программного кода при решении простых учебных задач в соответствии с ФРП по информатике углубленного уровня.

Все это затрудняет последовательную реализацию данного тематического раздела в соответствии с программой по информатике углубленного уровня.

Оптимальным считаем выбор языка программирования Python из-за его изначальной ориентации на обучение программированию, расширенных возможностей, связанных с применением достаточного большого количества специализированных библиотек для решения самых разных прикладных задач, в частности, задач искусственного интеллекта.

Открытый характер языка, разнообразие и доступность библиотек и средств разработки, легко читаемый характер программ, свобода реализации программ во всех основных концепциях профессионального программирования, применение для решения самых разных прикладных задач позволяют считать его лучшим выбором для обучения программированию в школе. Мы вступаем в этап изучения школьной информатики, где используется профессиональный язык программирования Python и с его помощью можно решать задачи всех разделов курса информатики.

Последовательность изучения программирования на языке Python лучше определить учителю самостоятельно исходя из возможностей обучающихся. Эта последовательность, скорее всего, не будет совпадать с описанной в программе версией из-за особенностей языка. Но в пределах темы и даже одного года менять последовательность изучения тем в содержании курса информатики разрешено.

В настоящее время идут непростые процессы перехода школ на обновленные ФГОС общего образования, а в информатике происходит очередная смена языка программирования на более современный вариант.

Вполне возможна ситуация, что обучающиеся в основной школе изучали Pascal. Учителю придется начинать язык Python с нуля, но используя аналогию с ранее изучаемым языком, можно ускорить процесс. Именно для такого случая предназначено издание: А.В. Щерба. Программирование на Python: первые шаги. – М. : Лаборатория знаний, 2022 [14], https://pilotlz.ru/books/276/11128/. В этом пособии большое количество практических заданий для отработки материала, подача теории в сравнении со структурным языком Pascal, акценты на синтаксис и его отработка позволят выйти на достаточный уровень подготовки к сдаче единого государственного экзамена в соответствии с предметными требованиями ФРП по информатике.

Для варианта изучения двух других языков в сравнении Python и C++, а также для изучения одного из предложенных доступно издание из четырех частей: К.Ю. Поляков «Программирование. Python. C++» (М.: Бином. Лаборатория знаний, 2022) [8–11]. Издание можно начать использовать с 8 класса и продолжать в 10 и 11 классах. Ссылки на все четыре части:
https://catalog.prosv.ru/item/48891; https://catalog.prosv.ru/item/48892;
https://catalog.prosv.ru/item/48893; https://catalog.prosv.ru/item/48894

Следует отметить, что с точки зрения решения школьных задач в области программирования каких-либо существенных выгод изучение двух языков одновременно не принесет. Безусловно, профессиональные средства и знание внутренних механизмов (например, организации стека вызовов функций), позволяют разрабатывать программные комплексы на нескольких языках (в частности, многие из библиотек Python реализованы на языках C и C++), но затраты времени на изучение этого программного кода (как правило, весьма объемного и связанного с внешними средствами и еще более объемными

библиотеками ОС) не будут оправданы. При этом будет потрачено время, которое отводилось на изучение других тем, также отраженных в школьном курсе информатики.

Не менее существенным затруднением может оказаться использование сложных и не изучаемых в школьном курсе алгоритмов и подходов (например, реализация умножения больших чисел и возведения в степень может сильно отличаться от школьного представления и требовать знания алгебры на уровне специализированного университетского курса, например, Python использует алгоритм Карацубы).

Изучение построения специализированных библиотек оправдано для профессионалов, но даже им рекомендуют изучать код не всех библиотек, а библиотек в их области специализации. Представляется оптимальным использовать уже готовые средства (библиотеки), обращая внимание на их алгоритмическую базу и реализуемый функционал.

В случае продолжения изучения языка программирования Python, начатого в основной школе, следует ориентироваться и всячески поддерживать мотивацию обучающихся. Именно мотивированные обучающиеся выбирают углубленный уровень изучения информатики, чтобы продолжить свое профессиональное образование в области информатики и информационных технологий. Такие обучающиеся ориентированы не столько на сдачу единого государственного экзамена, сколько на победу в олимпиадах, связанных с программированием, или победу в научно-технологических конкурсах, проводимых вузами вместо дополнительных вступительных испытаний. Вариант подготовки к олимпиаде, связанной с программированием, самый продуктивный для будущего ИТ-специалиста. Тогда и при выборе специализированных библиотек для изучения Python будет возможность ориентироваться на выбранную специализацию в информационных технологиях, например: искусственный интеллект или криптография (информационная безопасность). В настоящее время эти два направления специализации наиболее востребованы в отрасли информационных технологий. Ориентируясь на эти направления специализации, можно готовиться к олимпиадам, начиная с основной школы на базе языка Python. Учитывая ограниченные образовательные ресурсы по программированию на Python, следует обратить внимание на цифровые ресурсы организаций и вузов, на базе которых проводятся олимпиады по программированию или по указанным

специализациям. Например, альянс в сфере искусственного интеллекта предлагает материалы для уроков и подготовки к олимпиадам и конкурсам на сайте: https://a-ai.ru/school/

Министерство просвещения Российской Федерации в рамках Федерального проекта «Искусственный интеллект» национальной программы «Цифровая экономика Российской Федерации» с 2021 г. проводит Всероссийскую олимпиаду по искусственному интеллекту для обучающихся 8–11 классов общеобразовательных организаций [7]. Материалы олимпиады размещены по ссылке: https://olimp.edsoo.ru/

В России вузами проводятся еще две такие олимпиады – это олимпиада НТИ и Университета Иннополис. Их особенностями являются, в частности, ориентация на командную работу, а олимпиада Минпросвещения России является индивидуальным соревнованием, дающим дополнительные баллы для поступления в вуз по профилю.

Задачи первого отборочного этапа проверяют сформированность базовых навыков решения задач на языке Python. Это пока единственный доступный обучающимся язык программирования, для которого есть полный набор средств решения задач искусственного интеллекта.

На этом этапе отбираются участники, владеющие в необходимом объеме навыками решения задач на языке Python, с использованием его стандартных библиотек (например, регулярных выражений). Важно отобрать тех, кто не просто технически освоил приемы программирования, но и осознанно применяет его специальные возможности, может выдвинуть правдоподобные предположения для возможного решения и добиться результата. Анализ того, что проверяется на отборочном этапе, поможет учителю выстроить последовательность изучения возможностей Python с ориентацией на участие в олимпиаде. Проверяются:

– навыки ввода данных и вывода результатов;

– использования файлов для ввода и вывода;

– обработки числовой информации;

– обработки текстовой информации, в том числе работы с различными кодировками;

– навыки организации хранения данных в памяти, планирования структуры хранения данных для последующей обработки этих данных;

– некоторые навыки решения переборных задач.

Этот материал уже должен быть отработан в основной школе, а в старшей должен использоваться для решения конкретных олимпиадных задач из банка заданий по ссылке: https://olimp.edsoo.ru/ В настоящее время здесь собраны задания двух проведенных олимпиад, вебинары разработчиков с разбором заданий на Python, а в периодических научно-методических изданиях «Информатика в школе» (https://infojournal.ru/school/) и «Информатика и образование» (https://infojournal.ru/journals/info/info_01-2023/) авторами заданий дается полный разбор с кодом программ.

На втором, основном, этапе олимпиады предлагаются задания на использование специализированных библиотек, ориентированных на работу с числовыми, текстовыми и графическими данными, с ориентацией на подготовку и анализ наборов данных для последующего решения задач интеллектуальной обработки. Предусматривается возможность использования специализированных библиотек Numpy, Pandas, Pillow, sklearn, NLTK, Apyory. Задания предусматривают оценку уровня знакомства школьников с возможностями этих библиотек, умение применить их для решения ранее не встречавшейся задачи.

Решаются следующие классы задач:

– классификации;

– кластеризации;

– выявления ассоциативных правил;

– регрессии.

Предполагалось, что участники в своем решении:

– анализируют или формируют набор данных для решения задачи;

– выбирают модель для ее решения;

– формируют обучающий и (при необходимости) тестовый набор данных;

– проводят обучение и при необходимости тестирование модели;

– готовят программу, применяющую модель для формирования ответа и выводящую ответ в стандартный поток вывода.

Задачи не предусматривают самостоятельной реализации участниками алгоритмов перечисленных классов. Все необходимые для решения модели и средства имеются в составе библиотек, но ими необходимо грамотно воспользоваться.

Таким образом, к заключительному этапу участники подготовлены к необходимости разработки программной модели обработки данных

в соответствии с заданием. На третьем этапе участникам предлагается два задания: машинное зрение (сегментация изображений) и обработка текста на естественных языках (задача оценки тональности отзыва), где необходимо самостоятельно выбрать метод решения и реализовать его в модели. Организационной особенностью олимпиады является возможность обучающегося работать совместно со своим учителем информатики в дистанционном формате первых двух этапов олимпиады. Это возможность учителю нарастить свои навыки в области решения практических задач в области искусственного интеллекта. Для опытного учителя – это возможность выстроить индивидуальный маршрут подготовки к соревнованиям для перспективных обучающихся.

Обращаем внимание на олимпиаду по информатике и компьютерной безопасности имени И.Я. Верченко, проводимую в Институте криптографии, связи и информатики Академии Федеральной службы безопасности Российской Федерации. На сайтах: https://ikb.mtuci.ru/ и http://v-olymp.ru/ представлен архив заданий с решениями за более чем десятилетний период времени. Участие в олимпиаде индивидуальное, с льготами при поступлении в вуз по профилю олимпиады. Полезность этой олимпиады для учителя информатики в том, что можно скоординировать с учителем математики последовательность подготовки обучающихся к этой олимпиаде, акцентировать внимание на математические приемы решения задач и их реализацию на языке Python. Олимпиада полностью опирается на школьные знания и навыки, не проверяет специфических для темы умений, как в других олимпиадах по этой теме.

Необходимым условием для успешного решения олимпиадных задач является знание и использование в программе регулярных выражений. Это одно из самых известных и универсальных средств обработки текстов, применяемых для решения задач. *Регулярные выражения* – это механизм описания и преобразования шаблонных последовательностей символов. В том или ином виде (называемом диалектами регулярных выражений) механизм реализован и применяется во множестве стандартных библиотек языков программирования и самых разных информационных системах.

С теоретической точки зрения, регулярные выражения – это описание конечного автомата по преобразованию символов. Определенно здесь можно заметить сходство с машиной Тьюринга и впоследствии обратить внимание обучающихся на этот факт. Регулярные выражения применяются для поиска

(выборки) шаблонов из текста, проверки соответствия символьного выражения шаблону или преобразования шаблона. К таким задачам можно отнести задачу проверки правильности ввода (например, введен ли номер телефона или адрес электронной почты), преобразование шаблонных выражений (например, дат), разбиение или выделение значащих элементов из текста и т. д.

Вне зависимости от используемого механизма, основными элементами их составления будут текстовые последовательности, множества символов, количество их повторений в определенных позициях, допустимые варианты и средства их группировки.

Приведем несколько простых примеров для практической работы.

1. Выделение из текста целых неотрицательных чисел. Для этой задачи мы описываем целое число как последовательность символов из одной или более цифр: [0–9]+. Точно такой же смысл будет иметь выражение *d*+. В этом выражении мы обозначаем символ класса (т. е. входящий в набор цифр внутри скобок) и добавляем квантификатор + (указание, что символ должен встречаться один или более раз). Такое регулярное выражение выделит все целые числа, но если в тексте есть числа дробные, то и их части, а также значимую часть отрицательных чисел. Чтобы избежать захвата отрицательных чисел, добавим так называемые незахватывающие проверки: исключим число с минусом и цифрой после (т. е. отрицательные), исключим цифру с точкой или запятой после (исключим дробные числа) и исключим точку или запятую с цифрой после. Получим следующее выражение:

(?<![-\d])(?<!\d[.,])(\d+)(?![.,]?\d)

Пример несложной программы на языке Python, которая получит с клавиатуры имя файла и напечатает все целые числа, используя предложенное выражение:

```
import re

with open( input('Введите имя файла'),'r' ) as textSearch:
    text = textSearch.read()
    for res in re.findall('(?<![-\d])(?<!\d[.,])(\d+)(?![.,]?\d)', text):
        print( res )
```

2. Найти и преобразовать в тексте все даты в формате «ГГГГ-ММ-ДД» в привычный нам формат «ДД.ММ.ГГГГ». Для упрощения задачи предположим, что количество разрядов всегда такое, как в формате.

Выражение для поиска будет состоять из трех групп цифр, разделенных знаком «-»: (\d{4})-(\d{2})-(\d{2}) Мы указываем после обозначения допустимых знаков их точное количество – в фигурных скобках.

Выражение для замены будет состоять из ссылок на выделенные нами группы: \3.\2.\1

На языке Python программа может выглядеть так:

```
import re
with open( input('Введите имя файла'),'r' ) as textSearch:
    text = textSearch.read()
    replaced = re.sub('(\d{4})-(\d{2})-(\d{2})','\3.\2.\1',text)
```

Результат преобразования в строковой переменной replaced можно использовать как угодно.

Подготовить регулярное выражение и оценить, как оно будет работать с различными текстами на нужном вам диалекте, можно на общедоступном ресурсе по адресу: https://regex101.com/

В 11 классе в соответствии с ФРП по информатике углубленного уровня по теме «Элементы теории алгоритмов» отводится 6 часов учебного времени, один из которых должен быть отведен на практическую работу по составлению программы для машины Тьюринга. Предлагаем содержательный материал по этой теме. Учитывая аналогию в работе регулярных выражений и машины Тьюринга, можно считать эти практические работы взаимозаменяемыми.

Алгоритм и его свойства. Алгоритмически неразрешимые проблемы

Содержание темы урока

Предлагаемый урок-дискуссия начинается с повторения, когда еще раз на другом возрастном уровне рассматривается объяснение понятия «алгоритм» и реализуется попытка подойти к его строгому определению. Обучающимся предлагается рассмотреть три определения алгоритма. Первое определение, данное Д. Кнутом: *«Конечный набор правил, который определяет последовательность операций для решения конкретного множества задач и обладает пятью свойствами: конечность, определенность, ввод, вывод, эффективность»*, в котором можно дать современную трактовку свойств или несколько трактовок этих свойств одновременно. Вместе с обучающимися сравниваем определение Д. Кнута со словесными определениями, принадлежащими российским ученым А. А. Маркову и А. Н. Колмогорову.

1. *«Алгоритм – это всякая система вычислений, выполняемых по строго определенным правилам, которая после какого-либо числа шагов заведомо приводит к решению поставленной задачи»*. (А. Н. Колмогоров)

2. *«Алгоритм – это точное предписание, определяющее вычислительный процесс, идущий от варьируемых исходных данных к искомому результату»*. (А. А. Марков)

Методический прием сравнения различных определений и трактовок понятий позволяет обучающимся освоить аналитические приемы и формулировку обоснованных выводов. Введение нового материала происходит в формате дискуссии, когда обучающиеся активные участники обсуждения и самостоятельно формулируют выводы из обсуждения.

Обучающиеся высказываются обычно следующим образом: «Первое определение отличает наличие в нем свойств, его автор настаивает на том, что алгоритмом является лишь та последовательность действий, в которой выполняются перечисленные свойства. В современной трактовке свойства алгоритма звучат немного иначе, но они все равно должны выполняться». Соответственно, повторяем материал, изученный в основной школе, и обсуждаем подробно свойства алгоритма.

1. *Результативность* (конечность). Если исходные данные определены верно, то алгоритм будет выполнен за конечное число шагов – мы либо получим ответ, либо установим, что его нет.

2. *Детерминированность* (определенность – точность и понятность). Каждая команда в последовательности имеет одно и только одно значение. Команда входит в список допустимых команд исполнителя.

3. *Понятность и дискретность* (ввод и вывод). Алгоритм получает исходные данные и сообщает о результатах работы, а каждый последующий шаг алгоритма определяется предыдущим шагом.

4. *Эффективность* алгоритма определяется количеством действий, совершаемых исполнителем алгоритма для решения задачи, и объемом памяти, который требуется ему для выполнения этих действий.

Рассматривая алгоритмы, как правило, указывают и еще одно свойство – *массовость*. Необходимо обсудить примеры, подходящие под эти определения алгоритма, и проверить выполнимость свойств алгоритма. Прием сравнения на конкретных примерах и в этом случае неплохо работает. Например, можно

сравнить алгоритм Евклида и кулинарный рецепт. Первый пример – алгоритм, так как позволяет для любых двух чисел найти наибольший общий делитель и обладает всеми перечисленными свойствами. А второй пример не может считаться алгоритмом, поскольку есть проблемы со свойствами, например порядок действий можно варьировать, с «понятностью» не все в порядке (в рецепте «щепотка» – это сколько в граммах?), – результат получаем не всегда одинаковый.

Два других из приведенных выше определений алгоритма относятся только к вычислительным действиям. При этом для них также можно найти примеры, не являющиеся алгоритмами, поскольку требуют уточнения системы команд для конкретного исполнителя алгоритма или дополнительного включения числовых данных. Например, задачи на выполнение геометрических построений (проведение перпендикуляра к прямой в заданной точке).

Вместе с обучающимися подходим к вопросу: в чем же основная проблема, мешающая однозначно и точно определить «алгоритм»? Основной проблемой в приведенных определениях является их неформальность, неоднозначность, опора на естественный язык и опыт. Эти определения не дают возможности изучать алгоритмы в целом, как математическую абстракцию. А раз нет возможности изучить их, то нельзя быть уверенным, что набор действий действительно «решает» задачу – нет способа доказать или опровергнуть решение.

На рубеже прошлого столетия математики стали обсуждать вопрос возможности создания общего метода, определяющего разрешима произвольно составленная задача или нет. С формальной точки зрения речь шла о том, можно ли путем строгих, однозначных преобразований выяснить, истинно логическое высказывание или нет. Строгое определение алгоритма было необходимо как раз для того, чтобы сформулировать или хотя бы описать такой метод.

Алан Тьюринг в 1936 г., решая проблему разрешимости, предложил строгое математическое определение алгоритма, используя которое доказал невозможность создания универсального метода определения истинности или ложности произвольного высказывания в формальной системе аксиом.

Его определение алгоритма, строгая формальная конструкция преобразования символов, и называется машиной Тьюринга. Машина Тьюринга – формальная логическая конструкция, позволяющая описать преобразование цепочек символов по некоторым правилам – программе.

Машина Тьюринга состоит из бесконечной ленты с ячейками и управляющего устройства-считывателя. В ячейки на ленте записываются символы некоторого алфавита. Считыватель может находиться в одном из точно определенных состояний (внутренних). Работа машины описывается набором правил.

Каждое правило описывается исходными условиями (состоянием считывателя и буквой в текущей позиции) и результатом: новым состоянием считывателя, новым символом для текущей ячейки и сдвигом: влево, вправо или на месте.

Начальные данные задаются (вводятся) как начальное состояние букв на ленте, позиция управляющего устройства и его состояние, а также условие остановки.

Пример задания на составление программы для машины Тьюринга: на ленте есть слово, состоящее из символов #, $, 1 и 0. Требуется **заменить все символы # и $ на нули**. В момент запуска головка находится над первой буквой слова слева. Завершается программа (условие остановки) тогда, когда головка оказывается над пустым символом после самой правой буквы слова.

1	**#**	**$**	**1**	**0**	**#**	

В ячейке 1

Не менять символ, сдвинуться вправо, не менять состояние.

1	**#**	**$**	**1**	**0**	**#**	

В ячейке #

Записать ноль, сдвинуться вправо, не менять состояние.

1	**0**	**$**	**1**	**0**	**#**	

В ячейке $

Записать ноль, сдвинуться вправо, не менять состояние.

1	**0**	**0**	**1**	**0**	**#**	

В ячейке 1

Не менять символ, сдвинуться вправо, не менять состояние.

1	**0**	**0**	**1**	**0**	**#**	

В ячейке 0

Не менять символ, сдвинуться вправо, не менять состояние.

1	**0**	**0**	**1**	**0**	**0**	

В ячейке #

Записать ноль, сдвинуться вправо, не менять состояние.

1	**0**	**0**	**1**	**0**	**0**	

В ячейке нет символа, условие остановки

Не менять состояние.

Для закрепления темы рекомендуется сделать модификации этого задания, задавая другие позиции головки считывателя.

Машина Тьюринга приводится лишь в качестве примера строгого определения алгоритма. Тем учителям, которые считают необходимым этот материал изучить с обучающимися более подробно (мы разделяем позицию его полезности для развития алгоритмического мышления), советуем обратиться к учебному пособию: Е.В. Андреева, Л.Л. Босова, И.Н. Фалина. Математические основы информатики. Элективный курс. – М. : БИНОМ. Лаборатория знаний, 2005. – 328 с.

Отталкиваясь от описания работы машины Тьюринга, вводятся понятия *вычислимая функция* и *алгоритмически неразрешимая проблема* на уровне объяснения примеров.

Во-первых, нужно определить, что такое «алгоритмически разрешимая проблема». Алгоритмически разрешимая проблема (обозначим ее П) – это набор индивидуальных типовых задач, отличающихся конкретными исходными условиями (обозначим каждую из них α), для которых можно построить машину Тьюринга, вычисляющую следующую функцию:

$$f(\alpha)=\begin{cases}1, \alpha \in \Pi \ разрешима \\ 0, \alpha \notin \Pi \ неразрешима\end{cases}$$

То есть проблема, для которой такую машину построить нельзя, будет *алгоритмически неразрешимой*.

Пример такой проблемы: «Функция всюду определена», т. е. создание алгоритма, который может выяснить определена ли функция всюду или нет.

Предположим, что мы пронумеровали все вычислимые (алгоритмически разрешимые) функции и обозначили их $f_x(x)$, где x – натуральное число, номер машины. Расположить машины по порядку можно, например, упорядочив их по алфавиту.

Тогда можно, например, написать такую характеристическую функцию:

$$g(x) = \begin{cases} f_x(x) + 1, \text{ если } f_x(x) \text{ всюду определена;} \\ 0, \text{ если } f_x(x) \text{ не всюду определена.} \end{cases}$$

$g(x)$ – так называемая диагональная функция, такая функция всюду вычислима и определена, так что имеет некоторый номер y.

Поясним, почему для примененного метода выбран термин *диагональный*. Для каждой конкретной задачи из исходной проблемы мы можем построить машину Тьюринга (по условию задачи – это вычислимые функции). Расположив их по порядку, мы даем им номера и можем создать бесконечную таблицу.

	0	1	2	3	...
f_0	$f_0(0)$	$f_0(1)$	$f_0(2)$	$f_0(3)$	...
f_1	$f_1(0)$	$f_1(1)$	$f_1(2)$	$f_1(3)$	...
f_2	$f_2(0)$	$f_2(1)$	$f_2(2)$	$f_2(3)$	...
f_3	$f_3(0)$	$f_3(1)$	$f_3(2)$	$f_3(3)$	...
...	...	...	...	...	...

При построении функции g для определения значений в точках x выбирались диагональные элементы таблицы $f_0(0)$, $f_1(1)$, $f_2(2)$, … . Выбранные значения изменялись так, чтобы обеспечить отличие $g(x)$ от $f_x(x)$.

Если функция имеет некоторый номер y, то можно записать, что $g(y) = f_y(y) = f_y(y) + 1$. Это очевидное математическое противоречие, т. е. такая функция не существует.

Таким образом, мы видим, что не все проблемы алгоритмически разрешимы. И одна из существенных задач теории алгоритмов – определить, является ли некоторая проблема разрешимой в принципе.

Далее необходимо ввести понятие сложности алгоритма. К этому понятию подходим, обсудив необходимость сравнения различных алгоритмов. Поскольку

многие задачи решаются разными алгоритмами, возникает необходимость выбора наиболее эффективного из них.

Вначале вводятся критерии оценивания алгоритмов: количество действий и объем требуемой оперативной памяти.

Оценка количества действий в алгоритмах – основа для классификации алгоритмов по сложности. *Сложность алгоритма – это примерная оценка количества ресурсов (шагов исполнения и/или памяти), которые необходимо затратить на решение задачи с помощью этого алгоритма*. Затраты ставятся в зависимость от количества «единиц» входных данных.

Сгруппировав задачи по примерно одинаковым затратам, мы можем ввести *классы сложности*.

Вот некоторые из классов сложности:

- O(1) – количество действий постоянно и не зависит от количества входных данных. В подавляющем большинстве случаев это удачные алгоритмы, поскольку время их исполнения хорошо предсказывается.

Пример: вычисление суммы N первых элементов арифметической прогрессии по первому элементу прогрессии и разности. Вне зависимости от N, количество действий не изменяется – вычисления ведутся по формуле.

- L – O(log (n)) – количество действий сравнимо с логарифмом[1] n. Это быстрые алгоритмы, т. е. время их исполнения при увеличении объема данных растет медленно.

Пример: поиск значения в отсортированной последовательности методом деления пополам. В зависимости от количества элементов, количество шагов поиска будет пропорционально O(log (N)).

- P – O(P(n)) – количество действий не превосходит некоторого многочлена от n. Такие алгоритмы тоже считаются быстрыми. Скорость их решения сравнительно мало зависит от входных данных.

Пример: поиск пути в некотором графе.

- NP – задачи, сложность решения которых существенно зависит от количества данных. Но если есть некоторые дополнительные сведения, то задача становится задачей из предыдущего класса.

Пример: задача о сумме подмножеств заключается в нахождении (хотя бы одного) непустого подмножества некоторого набора чисел, чтобы сумма чисел

[1] Основание логарифма не имеет значения. Всегда можно преобразовать число, например, к основанию 2, вынеся некоторую константу.

этого подмножества равнялась нулю. Для этой задачи есть точный псевдополиномиальный (т. е. приближенный алгоритм) и алгоритм поиска примерного решения – полиномиальный.

- EXPTIME – $O(2^{p(n)})$ количество действий возрастает по некоторой экспоненте. Это алгоритмы вычислительно-сложные, поскольку с ростом количества данных их затраты растут очень быстро.

Пример: задача дискретного логарифмирования (обращения степени целого числа по некоторому модулю) – на сложности этой задачи основана криптография с открытым ключом.

Классификация алгоритмов по занимаемому ими во время работы пространству памяти аналогична.

Методические рекомендации по изучению тематического раздела «Информационные технологии»

Тематический раздел «Информационные технологии» на углубленном уровне представлен в 10 классе следующими темами:

1. Обработка текстовых документов (6 ч).
2. Анализ данных (8 ч).

В 11 классе предлагается изучать следующие укрупненные темы:

1. Компьютерно-математическое моделирование (8 ч).
2. Базы данных (10 ч).
3. Web-сайты (14 ч).
4. Компьютерная графика (8 ч).
5. Трехмерное моделирование (8 ч).

На изучение раздела в 10 классе выделяется 14 часов, в 11 классе – 48 часов.

В ФРП предусмотрено 18 практических работ в 10 классе и 20 практических работ в 11 классе.

Предметные результаты на двух уровнях изучения данного раздела представлены в таблице 1. В обновленных ФГОС СОО и ФРП по информатике углубленного уровня тематический раздел «Информационные технологии» претерпел изменения в сравнении с содержательной линией «Информационные технологии», включавшей в себя шесть технологических направлений, по которым рассматривалась фундаментальная теоретическая часть и прикладная составляющая, реализованная в программных продуктах. В существующем тематическом разделе фундаментальная теоретическая составляющая информационной технологии отсутствует, она изучается обобщенно в тематическом разделе «Теоретические основы информатики». Прикладная составляющая в виде практики работы в программных пакетах по обработке текста, числовых данных, графики, мультимедиа, компьютерное моделирование и анализ данных в ходе такого моделирования изучается в тематическом разделе «Информационные технологии». На обработку текстовых документов выделяется 6 часов, где уделяется внимание средствам автоматизированной работы с текстом: проверки орфографии и грамматики, стилевому оформлению, нумерации страниц, сноскам и оглавлению, средствам

коллективной работы с текстом, рецензированию и реферированию в соответствии с действующими стандартами оформления документов. Впервые введены темы компьютерной верстки текста. Эта часть материала может быть ориентирована на профессиональное самоопределение старшеклассников с учетом ограниченных возможностей здоровья. Здесь важно показать, что деятельность в отрасли информационных технологий может быть связана не только с программированием. Фактически, современный цикл подготовки издания представляет собой разработку компьютерной модели будущего издания. При этом множество технических понятий и терминов при подготовке издания пришли (точнее, остались) от наработанных за столетия художественных и технологических решений. Современные технические возможности позволяют быстро и сравнительно дешево превратить модель в реальное издание. Обучающимся интересна краткая история типографского дела, которая должна укрепить мысль о том, что компьютеры только ускорили процесс книгопечатанья, традиции и стандарты которого сформировались за несколько столетий до появления компьютеров.

Типографские традиции и стандарты. Компьютерная верстка текста

Содержание темы урока

Книгопечатание было изобретено дважды – в Китае (ориентировочно, по китайским источникам, в X в.) и в Европе (Гуттенберг, 1440 г.). Первопечатные книги (инкунабулы) были сделаны практически как копия рукописных книг методом ксилографии – зеркальное отражение страницы вырезалось на специальных досках (***матрицах***), на выступающие части наносилась краска. К доске прикладывалась бумага, краска отпечатывалась на ней. В зависимости от того вырезались ли буквы как углубления или как выступающие части страница была в основном черной или белой. Метод существует до сих пор под названием *ксилография*. Способ позволял получить устойчивый отпечаток только с одной стороны листа, поэтому все первопечатные книги – односторонние.

Ксилография применялась в Европе примерно с 1400 г., в основном для однолистовых изданий, как способ дешевого и быстрого размножения. Такой способ позволял тиражировать книги гораздо быстрее, чем при переписывании, но доски довольно быстро портились, а подготовка их была очень трудоемким

процессом. Поэтому отсчет современного книгопечатания начинают с применения наборного станка и пресса.

Суть изобретения Иоганна Гуттенберга – создание наборной печати. При этом методе матрица набирается из заранее подготовленных блоков с буквами, пробельных (для пробелов между словами) блоков, а позднее и блоков-украшений. Рисунки встраиваются в книгу отдельно. Набранная матрица прикладывается к странице с помощью пресса.

Буквы для типографий отливались специально, поэтому для набора больших объемов текста требовалось много одинаковых букв, причем очень ценилось их стилистическое единство. Очевидно, что использование разного размера букв требовало резкого увеличения количества имеющихся литер.

Со временем стандартизировались размеры букв и страниц, сложилась строгая классическая типографская традиция. С ростом требований к скорости и объемам печати, совершенствованием машин и механизмов типографии перешли (для крупных тиражей) к методам, при которых пресс и наборные матрицы заменили барабаны. Первоначально страницы на барабанах гравировались.

Современным методом является метод офсетной печати, при котором изображение формируется на барабане-исходнике, краска переносится на барабан-посредник, а уже оттуда на носитель.

Точность современных механизмов позволяет формировать таким образом и многоцветные изображения – разделив их по цветам на четыре барабана. Малотиражные издания и сейчас чаще всего размножаются с помощью печатной аппаратуры.

Долгое время в типографском деле не было единой системы мер и стандартов. Впервые с идеей введения единой стандартной единицы измерения выступил французский печатник Пьер Симон Фурнье (1737). Позже эта система была уточнена (были, в частности, исправлены ошибки, возникшие при переизданиях) Франсуа-Амбруазом Дидо (1783).

В основе системы мер лежал французский королевский фут – 324,84 мм. Один фут равнялся 12 дюймам, один дюйм – 12 линиям, а линия – 12 пунктам. В Америке в 1879 г. (позже и в Англии) Нельсоном и Хоуксом был введен пункт, равный 0,3514 мм (0,0318 английского дюйма) [13].

С тех пор основной типографской единицей является ***пункт*** (от лат. punctum – точка), но фактически у него есть два основных значения и несколько округлений. Очевидно, что при отсутствии возможности высокоточных вычислений (на отливках это практически было невозможно) всегда использовались некоторые округления.

В России принятой является европейская, т. е. французская и немецкая система (1 пункт = 0,376 мм). Но применяется и другой вариант, положенный в основу языка представления PostScript и ровный $^1/_{72}$ английского дюйма, т. е. 0,013(8) дюйма (1 пункт = 0,3514 мм).

Основным и наиболее важным средством определения внешнего вида текста является **шрифт** (от нем. Schreiben – писать). Шрифт – это графический рисунок букв, цифр и символов, обладающий общими для всех символов стилистическими особенностями изображения. Шрифт – средство визуализации букв. Кегль (кегель) – размер шрифта – предельная высота большой буквы и окружающих ее пробелов (термин введен для описания высоты площадки литеры при наборе с помощью типографской кассы). Чаще всего задается в типографских пунктах (1 пункт = $^1/_{72}$ дюйма = 0,375 мм). По историческим причинам некоторые размеры имеют собственные названия: 8 пт – петит, 9 пт – боргес, 10 пт – корпус, 12 пт – цицеро. Собственно, величина типографского пункта подбиралась исходя из размера «цицеро». Совокупность всех возможных размеров и вариантов написания шрифта называется **гарнитурой**. Гарнитуры имеют имена, по которым часто называют и конкретный шрифт.

По общим чертам рисунка различают три основных вида шрифтов.

Рубленые шрифты (гротески). Для этих шрифтов характерно угловое соединение штрихов. Чаще всего у таких шрифтов нет засечек. Такими шрифтами часто набирают большие объемы текста.

Антиквенные шрифты (Таймс). Происходят от созданного Альбрехтом Дюрером шрифта «Антиква». В этом шрифте соединения между штрихами сглажены, обязательны засечки. Это наиболее популярная «книжная» группа шрифтов.

Акцидентные (оформительские) шрифты. Шрифты с самым разным рисунком, применяемые для оформительских целей, часто стилизованные под рукописные буквы. Чаще всего они выделены в специальные коллекции шрифтов, которые нужно установить специально.

Шрифт задается для набранного текста и не изменяет самих символов – он только определяет написание каждого символа, исходя из эталонного изображения. Библиотека таких изображений называется просто «шрифтом» при хранении и передачи.

Существует несколько основных способов описания шрифтов (точнее – гарнитуры шрифта) для хранения и использования с помощью компьютеров.

1. *Растровые шрифты.* При таком способе каждая буква описывается отдельно, как некоторая матрица точек. Такой способ позволяет максимально ускорить обработку, но сильно затрудняет изменение размеров или начертаний. Для достижения качества каждый символ такой гарнитуры должен быть отредактирован вручную и хранится отдельно.

2. *Векторные шрифты*. При таком способе описания шрифт задается с помощью некоторых математических кривых, совокупность которых и составляет рисунок символов. Такой шрифт может изменять размеры без потери качества, но с помощью примитивов трудно добиться прорисовывания заполняемых элементов.

3. *Контурные шрифты*. Аналогично векторным, описываются с помощью некоторых математических кривых, но они определяют не символ, а его контур. При использовании контур по определенным правилам заполняется. Именно этот тип шрифтов и является наиболее популярным.

Для использования векторных и контурных шрифтов необходимо выполнение операции, «создающей» шрифт (заданного рисунка, размера и начертания), годного для отображения. Такая операция называется «растеризацией». В состав графических оболочек современных операционных систем входят программы-растеризаторы шрифтов определенного формата.

Растеризация шрифта – достаточно ресурсоемкая операция, поэтому распространение контурные шрифты получили только с массовым применением достаточно мощных компьютеров.

При подготовке любого печатного издания многое зависит от его назначения и формы. Можно выделить несколько самых частых форм печатных изданий.

1. *Плакат* (*листовка, объявление и т.д.*) – это издание из одного листа, причем одностороннее. Чаще всего текста на нем немного, но он должен привлекать внимание.

2. *Буклет.* Издание с небольшим количеством листов (часто – из одного листа), не скрепленных, как правило двустороннее. Листы сгибаются несколько раз и складываются. Способов сложить лист бумаги существует множество, поэтому и верстка буклета – не всегда проста.

3. *Брошюра.* Несколько листов, скрепленных вместе по средней линии. Текст размещается на листе двумя блоками.

4. *Книга.* Стопа листов (часто – брошюр) с общим скрепленным краем.

При печати крупных изданий (книг) печать не выполняется на таких же листах, каким оказывается итоговый формат книги – это было бы непроизводительно. Поэтому печать выполняется на куда более крупном *печатном листе*, который после печати складывают (фальцуют), а потом обрезают края так, чтобы остались только те, что скрепляют брошюру.

Подсчет объема издания чаще всего выполняется в листах, и с учетом меняющихся форматов может быть не очень простым. Как правило, за основу таких подсчетов берется *авторский лист* – примерно 40 000 знаков или 3000 строк стихотворного текста, т. е. 10–12 страниц текста, набранного шрифтом цицеро (12 пунктов) через один интервал.

Рассмотрим некоторые наиболее часто встречающиеся типографские термины.

Полоса набора – участок страницы, на котором размещается весь текст и иллюстрации. Чаще всего это прямоугольник, определяемый размерами и положением на листе.

Верстка – процесс и результат формирования полосы, листа, книжной страницы или разворота из пришедшихся на них элементов: текста, иллюстраций, подписей, украшений и т. д. Словом «верстка» также обозначают подготовленное к печати издание.

Макет – первоначально макетом называлась условное представление всего текста, модель. В такой модели вместо всего текста мог присутствовать кусок, а то и заменитель реального текста так называемая «рыба». Сейчас с внедрением цифрового набора макет – фактически полностью подготовленное, представленное в виде цифровой модели издание. В этом случае «макет» становится синонимом слова «верстка».

Поля – пустое (не заполненное содержанием издания) пространство вдоль края страницы.

Интерлиньяж – степень близости строк, межстрочный интервал; часто задается пропорционально размеру шрифта.

Колонка – элемент развертки текста по вертикали, фигура из определенного количества строк заданного формата при заданном интерлиньяже.

Абзац (от нем. Abzac – отступ) – зрительно выделенный небольшой период текста. Слово применяется уже не только в типографике, но и в лингвистике.

Абзацный отступ – отступ от края колонки в первой строке абзаца, одно из средств визуального выделения начала абзаца. Рекомендуемые величины: в одну кегельную, т. е. на высоту шрифта; «в круглый» – так назывался пробельный кусочек («шпация») квадратного сечения. Меньшее расстояние недостаточно наглядно выделяет начало. Хотя большее расстояние – неоправданная трата места, но чаще всего используют большую величину, заимствованную из машинописного набора, – в три или четыре символа.

Колонэлемент – колонтитул (текст), колонлинейка (горизонтальная линия), колонцифра (номер страницы или части) – элементы, присутствующие на каждой странице, например название книги, текущего раздела (название, конечно, меняется – но назначение остается) и т. д.

Рубрика – заголовки частей издания, текст, выделяющий смысловую область текста.

Выключка – доведение строки до заданного формата или (если строку не «разравнивают» изменением ширины пробелов) расположение незаполненной строки относительно полосы набора – по одному из краев или по центру, красная строка и т. д. Простейшие случаи выключки в текстовых процессорах называются «выравниванием»[1].

Маргиналии – небольшие текстовые блоки, выносимые на поля, используются для специальной нумерации или справочных замечаний.

В целом ряде случаев требования к размещению материалов на страницах заданы действующими стандартами (например, при подготовке научных работ, официальных документов и т. д.). Как правило, документы и другие аналогичные материалы считаются не изданиями, поскольку не предусматривают размножения, а рукописями.

[1] А в специализированных системах так и будет – выключка строк.

Современная верстка страниц осуществляется блоками. Простейший случай – наличие на странице одного единого блока текста. Чуть более сложный – размещение текста в несколько колонок.

Существенная часть работы при размещении блоков – создание единого, цельного образа страницы, причем такого, на который будет не только приятно смотреть, но и удобно читать. Работа по размещению блоков и ее результат называются композицией страницы.

Для этого при формировании страницы можно придерживаться таких правил:

1. Композиция страницы смотрится цельно, когда она ***равновесная***, т. е. когда элементы визуально «уравновешивают» друг друга относительно композиционного центра.

2. Поля должны быть обязательно. Поля визуально ограничивают текст и позволяют читателю легко зафиксировать его на странице.

3. Между отдельными блоками должно быть достаточно пустого пространства, чтобы отделить блоки друг от друга.

4. Важное качество верстки (особенно для многолистовых изданий!) – ее постоянство, предсказуемость. В размерах, в расположении блоков и колонэлементов, в размерах полей и шрифтов. Аккуратно выполненная верстка позволяет читателю сосредоточиться на тексте, не тратить время на поиск и выделение элементов. Предсказуемая верстка легко читается!

5. При выборе размеров блоков и полей наилучшие результаты дает соотношение золотого сечения.

6. Несмотря на то что поля – это ничем не заполненное пространство, они тоже на листе смотрятся как блок и участвуют в балансе страницы – и самостоятельно, и с текстовыми блоками и иллюстрациями.

7. Следует помнить, что и текст и блоки на странице видны как разница между составляющими их областями цвета и фоном. Если фон будет слишком насыщенным, мало будет отличаться по цвету от букв, будет иметь самостоятельный рисунок – визуально выделить буквы и блоки будет сложнее.

Процесс подготовки и тиражирования издания в современных условиях делят на две части: подготовка оригинал-макета (т. е. полностью готовой

к печати копии издания) и собственно тиражирование (т. е. изготовление копий оригинал-макета).

Подготовкой оригинал-макета занимается издательство, а тиражированием – типография. В подготовку оригинал-макета входят: подготовка текста (включая его написание, редактирование, корректуру) и верстка (подготовка оригинал-макета издания).

Процесс подготовки издания полностью компьютеризирован и в процессе его используют:

– собственно *компьютер,* с установленным программным обеспечением, как основной инструмент ввода, правки текста, основное средство подготовки макета;

– *сканеры* как устройства для ввода графических изображений, в том числе текстов – для последующего распознавания;

– *цифровые фотокамеры* – применение этих устройств позволило значительно сократить цикл подготовки изданий, особенно периодических;

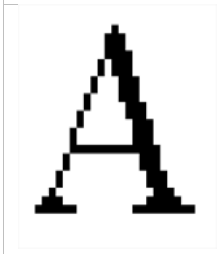
– *принтеры*, чаще всего лазерные, позволяющие получить оригинал-макет в виде, максимально близком к печатному.

Подготовка текста завершается созданием оригинал-макета для отправки в типографию, как правило в виде либо эталонного отпечатка, либо электронной версии в оговоренном формате. В зависимости от того как построен процесс печати, издательство может передать на размножение бумажную копию, либо файл макета в оговоренном формате, либо прозрачные пленки и т. п.

Стоит заметить, что появление и распространение мощных компьютеров, качественных и быстрых печатающих устройств, средств цифровой обработки изображений привело к появлению *настольных издательских систем*, т. е. комплексов, которые позволяют целиком подготовить макет к печати и даже выполнить небольшой (в пределах нескольких сотен) тираж.

Задания по теме, которые можно использовать в ходе изучения материала

1. По изображению определите тип компьютерного шрифта и дайте ему краткое определение.

Картинка	Тип шрифта	Определение
		
		

2. Соотнесите примеры шрифта с его названием и опишите ключевую характеристику шрифта.

Й У В К Е Н Г	Антиквенные шрифты	
J F D K L K B V F	Рубленые шрифты	

3. Дополните предложение.

По происхождению языки можно разделить на … и … .

… – представляет собой последовательность символов, разбитых на абзацы или строки.

… – это графический рисунок букв, цифр и символов, обладающий общими для всех символов стилистическими особенностями изображения.

4. Сопоставьте понятие и его определение.

1. Рисунок шрифта
2. Кегль (кегель)
3. Начертание

A. Это размер шрифта: предельная высота большой буквы и окружающих ее пробелов. Этот термин введен для описания высоты площадки литеры при наборе с помощью типографской кассы

B. Графические особенности, определяющие общность всех элементов шрифта и его отличие от всех других шрифтов

C. Шрифт с общим рисунком, но какими-либо отличительными признаками: более жирный, наклонный, разряженный

В соответствии с ФРП по информатике углубленного уровня в тематическом разделе «Информационные технологии» в 10 классе присутствует тема «Анализ данных» объемом 8 часов. Для изучения этой темы можно использовать следующее учебное пособие: И.А. Калинин, Н.Н. Самылкина, А.А. Салахова. Искусственный интеллект: 10–11 классы. – М.: Просвещение, 2023. – 144 с. [5]. Издание можно найти по ссылке: https://shop.prosv.ru/iskusstvennyj-intellekt--10-11-klassy21811

Указанное учебное пособие представляет собой практикум по искусственному интеллекту, состоящий из 12 кейсов, которые включают в себя:

– описание проблемной ситуации (конкретную проблему и путь ее решения должны предложить сами обучающиеся);

– перечень инструментов (библиотек и фреймворков);

– данные (дата-сеты).

С использованием практикума решаются основные задачи анализа данных: классификация, кластеризация, линейная регрессия, анализ отклонений, составление ассоциативных правил, построение дерева решений. Создается экспертная система и изучается работа нейронной сети. Для реализации разработанных моделей используются дата-сеты из свободно распространяемых источников. Все практические задания выполняются с помощью дистрибутива Anaconda на языке Python с использованием библиотек SciKit-Learn, TensorFlow и Keras, модуля Apriori, пакета PyKnow или experta и другими популярными библиотеками, часто применяемыми в DataScience-проектах.

Для некоторых тем созданы практические материалы, не требующие программирования, представляющие собой игры, в ходе которых демонстрируются общие принципы работы интеллектуальных алгоритмов (например, при кластеризации). Такие задания решаются на доске или в тетради и помогают создать у обучающихся ассоциации с привычными им нецифровыми методами решения задач.

Теоретический материал по каждой из тем дается в таком объеме, чтобы можно было получить общее представление о том, как это работает и зачем все это нужно, а затем выполнить практические задания на применение изученной теории. Кроме общей исторической справки и основных понятий, рассказано о некоторых основных направлениях развития и применения средств

искусственного интеллекта. В приложении кратко излагается история вопроса, чтобы обзорно охватить довольно большой период времени с того момента, когда все начиналось и кем и с чем связан некоторый перерыв и смена акцентов в разработке различных направлений искусственного интеллекта.

В конце практикума предложена программа курса «Основы ИИ» с тематическим планированием на 18 и 32 часа. Это означает, что при отсутствии времени в курсе информатики углубленного уровня можно использовать время на подготовку к олимпиаде по ИИ или проектную и исследовательскую деятельность, которая в школе является обязательной.

В данном пособии предлагаем дополнительно рассмотреть новую тему «Экспертные системы» с практической реализацией двух экспертных систем «Сладкое угощение» и «Виртуальный доктор».

Любая экспертная система направлена на формирование логического вывода в результате цепочки рассуждений (точнее, логических следствий).

Для этого экспертные системы должны иметь «представление» о своей предметной области – базу знаний. Базы знаний у экспертных систем состоят из трех основных частей:

– *фактов*, т. е. утверждений, зафиксированного состояния предметной области. Факт может описывать объект (Елена, человек, женщина, 165 см, голубые глаза), связь (Елена – мать Светланы), состояние (родственники) и вообще все, что мы только можем зафиксировать;

– *правил* – преобразования фактов, т. е. их создания, модификации или удаления. Например (если Елена – Мать Светланы, Полина – Мать Елены, то добавить факт «Полина бабушка Елены»). Правила могут носить довольно сложный характер, но общий принцип таков;

– *машины логического вывода* – это реализованный алгоритм использования правил и фактов, т. е. собственно построения выводов.

Поскольку возможности языка Python с специализированными библиотеками позволяют реализовать данную задачу, то она может быть в приоритете по отношению к просто проектированию базы данных.

Для работы будем использовать комплект свободно распространяемого программного обеспечения, состоящий:

– *из интерпретатора языка программирования Python* с некоторым набором расширений – IPython;

– *библиотек алгоритмов* (в том числе интеллектуальных), реализующих всю работу с данными – получение их из разных источников, а также их преобразование и обработку;

– *web-оболочки*, которая позволит нам создавать документы и с описанием, и с программным кодом, и с результатами его исполнения.

Большую часть необходимых средств (интерпретатор языка, основные библиотеки, оболочку для работы) скачаем в виде единого комплекта распространения ПО (дистрибутива) Anaconda. Если не хочется скачивать большой комплект сразу целиком (больше 500 Мбайт), можно скачивать и устанавливать его частями. Недостающие библиотеки установим дополнительно – они также доступны в сети.

Задание 1. Сладкое угощение

Для демонстрации работы разрабатываемой экспертной системы мы используем загруженную библиотеку experta. Импортируем все файлы из данной библиотеки:

```
from experta import *
```

Первая экспертная система позволит определить, подходит ли для нашего сладкого чаепития та еда, которую принесли ребята. То, что подойдет, будет иметь следующие признаки: сладкое, можно разделить на несколько человек.

Для начала создадим класс под машину вывода (KnowledgeEngine), которая сможет выдать ответ на вопрос «Подойдет ли принесенная еда для сладкого чаепития?» только тогда, когда узнает определенные факты.

```
class Food(KnowledgeEngine):
@DefFacts()
def _initial_action(self):
yieldFact(action="know")
```

Изначально в нашей системе нет фактов о принесенном угощении, поэтому мы создаем правило, которое запросит пользователя их ввести. Для этого нужно создать функцию, которая запрашивает у пользователя информацию о таких характеристиках, как сладкое или пресное и можно ли разделить угощение на несколько человек. Почему так важен факт о возможном разделении? Согласитесь, что мало кому понравится, если только один человек будет есть вкуснейшее пирожное. Поэтому такое угощение подойдет только в том случае, если хотя бы несколько человек смогут им полакомиться. Теперь наша

экспертная система выясняет два характеризующих факта, которые пользователь вводит с клавиатуры, отвечая на соответствующие вопросы, и переобозначает их для дальнейшей работы.

```
    @Rule(Fact(action='know'),NOT(Fact(sweet=W())),
NOT(Fact(countable=W())))
    defwhois(self):
    self.declare(Fact(sweet=input("Это сладкое или
пресное? "), countable=input("Его можно разделить на
несколько человек? ")))
```

Теперь нам необходимо получать обратную связь, т. е. наша система должна давать ответ о том, подходит ли данное угощение или нет. Если оно окажется и сладким, и поддающимся делению, то оно полностью подходит нам. Если же оно сладкое, но предназначено лишь для одного человека, то стоит докупить это угощение для большего количества едоков. И в случае если оно пресное, то оно нам просто не подходит, так как наше чаепитие именно сладкое, и факт возможности разделить его на несколько человек уже не является важным. Для получения этого ответа система считывает переменные sweet и countable, где хранятся наши ответы, а затем отправляет их в вывод.

```
    @Rule(Fact(action='know'),Fact(sweet="sweet"<<W()),
Fact(countable="countable"<<                    W()))
    def thinking(self, sweet, countable):
    print("Оно %s и его %s разделить на несколько человек."
% (sweet, countable))
    if((sweet=="сладкое") & (countable=="можно")):
    print("Подходит для чаепития!")
    elif((sweet=="сладкое") & (countable=="нельзя")):
    print("Если купить побольше этой сладости, то она
подойдет для чаепития!")
    else:
    print("Это точно не подойдет для сладкого чаепития.")
    self.sweet = sweet
    self.countable = countable
```

Наша экспертная система готова к использованию. Для ее запуска мы объявляем машину вывода, сбрасываем ее состояние, а затем запускаем на выполнение.

```
engine = Food()
engine.reset()
engine.run()
```

Таким образом мы получили следующий код:

```
from experta import *
class Food(KnowledgeEngine):
    @DefFacts()
    def _initial_action(self):
       yield Fact(action="know")

    @Rule(Fact(action='know'),NOT(Fact(sweet=W())),
NOT(Fact(countable=W())))
    defwhois(self):
    self.declare(Fact(sweet=input("Это сладкое или
пресное? "), countable=input("Его можно разделить на
несколько человек? ")))

    @Rule(Fact(action='know'),Fact(sweet="sweet"<<  W()),
Fact(countable="countable"<< W()))
    def thinking(self, sweet, countable):
    print("Оно %s и его %s разделить на несколько человек."
% (sweet, countable))
    if((sweet=="сладкое")& (countable=="можно")):
    print("Подходит для чаепития!")
    elif((sweet=="сладкое") & (countable=="нельзя")):
    print("Если купить побольше этой сладости, то она
подойдет для чаепития!")
    else:
    print("Это точно не подойдет для сладкого чаепития.")
    self.sweet = sweet
    self.countable = countable
```

```
engine = Food()
engine.reset()
engine.run()
```

Работа данной экспертной системы выглядит следующим образом:

```
1)  Это сладкое или пресное?  пресное
Его можно разделить на несколько человек?  можно
Оно пресное и его можно разделить на несколько человек.
Это точно не подойдет для сладкого чаепития.
2)  Это сладкое или пресное?  сладкое
Его можно разделить на несколько человек?  нельзя
Оно сладкое и его нельзя разделить на несколько человек.
Если купить побольше этой сладости, то она подойдет для чаепития!
3)  Это сладкое или пресное?  сладкое
Его можно разделить на несколько человек?  можно
Оно сладкое и его можно разделить на несколько человек.
Подходит для чаепития!
```

Задание 2. Виртуальный доктор

Во второй практической работе будем разрабатывать экспертную систему, позволяющую воспроизвести примерную беседу в кабинете у врача. В современном мире существуют аналогичные системы, которые на основе данных о самочувствии пациента позволяют поставить предварительный диагноз. Конечно постановка диагноза – очень сложный процесс, поскольку требуется проверить и учесть множество факторов, но для понимания принципа работы в целом мы упростим оценку состояния организма. Если задание окажется слишком легким для программирующих обучающихся, усложните его добавлением предварительного диагноза в более точной медицинской формулировке.

Шаг первый: подключим загруженную нами ранее библиотеку experta (при необходимости повторите ее загрузку):

```
from experta import *
```

Для создания базы знаний необходимо определить, каким образом будет характеризоваться самочувствие пациента. Он может сказать беспокоит ли его какой-либо симптом, но только врач сможет на основании этих данных поставить диагноз.

Каждый симптом будет иметь следующие характеристики: название, «тяжесть», запрос на констатацию и дальнейшее исследование симптома. Опишем самочувствие с потомка класса Fact. Таким образом, мы опишем характеристику каждого изучаемого нами симптома и это пригодится в дальнейшем при написании правил для работы экспертной системы.

```
def ask (question):
return (True if input(question+'[да/нет]') == 'нет' else False)
class Health (Fact):
symptom = Field(str,default='')
severity = Field(str,default='неизвестно')
treatment = Field(str,default='неизвестно')
```

Реализация диалога, позволяющего провести процедуру оценки здоровья, происходит с помощью определенной нами функции. Процесс: пациенту задается вопрос, на который он честно отвечает да или нет. На основе ответа уже происходит дальнейшая работа.

Некоторые симптомы представляют с собой целую группу, и это тоже необходимо учесть. То есть если «главный» симптом группы отсутствует, то второстепенные уже не рассматриваются. Главные симптомы обозначим – major, второстепенные – minor. Создаем класс, отражающий связи между ними:

```
class Relation(Fact):
main = Field(str,default='')
minor = Field(str,default='')
```

Переходим к описанию самой базы знаний. Для этого пропишем интересующие нас симптомы, их связи и начальные знания об их наличии:

```
class HealthDiagnostic(KnowledgeEngine):
@DefFacts()
def _initial_action(self):
yield Health(symptom='Самочувствие',severity='беспокоит')
```

```
            yield Health(symptom='Расстройство ЖКТ',
severity='неизвестно',treatment = 'Испытываете боли в области
живота?')
            yield Health(symptom='Проблемы с дыханием',
severity='неизвестно',treatment = 'Дыхание затрудненное?')
            yield Health(symptom='Боль в горле',
severity='неизвестно',treatment = 'Испытываете боли в горле?')
            yield Health(symptom='Головная боль',
severity='неизвестно', treatment = 'Испытываете головную боль?')

    yield Health(symptom='Тошнота',severity='неизвестно',
treatment = 'Вас тошнит?')
            yield Health(symptom='Насморк',severity='неизвестно',
treatment = 'Нос заложен?')
            yield Health(symptom='Кашель',severity='неизвестно',
treatment = 'Кашель беспокоит?')
            yield
Health(symptom='Головокружение',severity='неизвестно', treatment =
'Испытываете головокружение?')
    yield Health(symptom='Рвота',severity='неизвестно', treatment
= 'Была ли рвота?')
            yield Health(symptom='Обморок',severity='неизвестно',
treatment = 'Приступы обморока были?')

    yield Relation(main='Самочувствие',minor='Головная боль')
            yield
Relation(main='Самочувствие',minor='Расстройство ЖКТ')
            yield Relation(main='Самочувствие',minor='Проблемы с
дыханием')
            yield Relation(main='Самочувствие',minor='Боль в
горле')

    yield Relation(main='Расстройство ЖКТ',minor='Тошнота')
            yield Relation(main='Проблемы с
дыханием',minor='Насморк')
            yield Relation(main='Боль в горле',minor='Кашель')
            yield Relation(main='Головная
боль',minor='Головокружение')

            yield Relation(main='Тошнота',minor='Рвота')
            yield Relation(main='Головокружение',minor='Обморок')
```

К интересующим нас симптомам отнесем головную боль, расстройство ЖКТ, проблемы с дыханием, боль в горле, тошноту, насморк, кашель, головокружение, рвоту, обмороки. При этом, при головной боли возможно головокружение, а при нем возможен обморок. При расстройстве ЖКТ возможна тошнота, а также рвота. Если человек испытывает проблемы с дыхательной системой, то это может сопровождаться насморком. Аналогично с болью в горле и кашлем.

Для проверки симптомов будем задавать соответствующие вопросы:

Симптом	Вопрос
Расстройство ЖКТ	Испытываете боли в области живота?
Проблемы с дыханием	Дыхание затрудненное?
Боль в горле	Испытываете боли в горле?
Головная боль	Испытываете головную боль?
Тошнота	Вас тошнит?
Насморк	Нос заложен?
Кашель	Кашель беспокоит?
Головокружение	Испытываете головокружение?
Рвота	Была ли рвота?
Обморок	Приступы обморока были?

При исследовании симптомы могут иметь следующие статусы:

Статус	Значение
Не беспокоит	Вопрос задан. Ответ – нет, т. е. исследуемый симптом у пациента отсутствует
Беспокоит	Вопрос задан. Ответ – да, т. е. исследуемый симптом у пациента есть
Проверяем	Симптом необходимо исследовать
Неизвестно	Нет данных

Шаг второй: прописать правила, на основе которых будет проходить анализ самочувствия пациента. Одно из важных достоинств нашей системы заключается в том, что в дальнейшем мы сможем добавить в нее любые новые симптомы, задать их связи с уже существующими. Никаких изменений в остальной код вносить не потребуется, так как наши правила актуальны всегда.

Первое правило заключается в следующем: если симптом еще не проверен (стоит по умолчанию), то необходимо задать соответствующий вопрос пациенту, исходя из ответа дать симптому характеристику: беспокоит или не беспокоит. Если беспокоит, то выяснить насчет второстепенных симптомов для данного. В общем случае оно будет срабатывать при исследовании всех симптомов, которые мы выделим как важные для постановки диагноза в будущем.

Важно! Каждый симптом, подходящий под условие в левой части, будет связан с переменной checked с помощью конструкции AS.checked<<, и эту переменную мы передаем функции, которая является правой частью, вместе с ссылкой на механизм вывода.

```
    @Rule(AS.checked << Health(severity='проверяем'),salience=0 )
        def checksymptom(self,checked):
    s = ('не беспокоит' if ask (checked['symptom']+' '
+checked['treatment']) else 'беспокоит' )
            self.modify( checked, severity = s )
            if s == 'беспокоит':
    print(checked['symptom'],' - необходимо исследовать более
подробно')
```

Второе правило позволяет нам не делать лишнюю работу. При написании классов мы говорили, что симптомы бывают главные и второстепенные. Если главный симптом отсутствует, то связанные с ним второстепенные не требуют проверки. Именно это нам и необходимо отметить в характеристике симптома. Например, если у пациента не болит живот, то проверять симптом тошноты не имеет смысла.

```
    @Rule(Health(symptom = MATCH.devname,severity='не
беспокоит'), Relation(main = MATCH.devname , minor =
MATCH.minor),AS.minorpart << Health(symptom = MATCH.minor,
severity= 'неизвестно'), salience=10)
        def markrelation(self, minorpart):
            self.modify(minorpart, severity = 'не беспокоит' )
            print ('Полагаю,',minorpart['symptom'],' не
беспокоит.')
```

Следующее правило определяет, когда нам нужно исследовать второстепенные симптомы: главный симптом – беспокоит, а про второстепенные пока нет информации, так как о них пациента еще не спрашивали. Так происходит отбор симптомов для исследования из оставшихся.

```
    @Rule(Health(symptom = MATCH.devname,severity='беспокоит'),
Relation(main = MATCH.devname , minor = MATCH.minor),NOT(
Health(symptom = MATCH.minor, severity= 'плохое')),AS.minorpart <<
Health(symptom = MATCH.minor, severity= 'неизвестно'), salience=5)
        def markrelation(self,minorpart):
            self.modify(minorpart, severity = 'проверяем' )
            print ('Проверяемый симптом:',minorpart['symptom'])
```

Теперь необходимо определить симптом, который имеет не ярко выраженную картину протекания, вследствие чего необходимо дополнительное исследование. То есть он сам беспокоит пациента, а его подчиненные симптомы нет. Например, головная боль есть, но нет головокружения, но ведь отсутствие последнего не отменяет первого симптома.

```
    @Rule(AS.badsymptom << Health(symptom = MATCH.devname,
severity='беспокоит'),NOT(Relation(main = MATCH.devname, minor =
MATCH.minor), Health(symptom=MATCH.minor,severity=MATCH.severity),
TEST (lambda severity : severity in ('беспокоит',
'неизвестно','проверяем'))), salience=1)
        def isbadsymptom(self,badsymptom):
            print ('Вас беспокоит:',
badsymptom['symptom'],'Необходимо дополнительное исследование')
```

Проверка симптомов рано или поздно закончится, поэтому необходимо правило, определяющее, что исследовать больше нечего:

```
    @Rule(NOT(Health(severity=MATCH.severity),TEST(lambda
severity : severity in ('неизвестно','проверяем'))), salience=10)
        def allchecked(self,minorpart):
            print ('Симптомы проверены')
```

Наша экспертная система готова к использованию. Для ее запуска мы создаем объект – пациент, который проходит диагностику здоровья, переводим его в начальное состояние, а затем отправляем программу на выполнение.

```
engine = HealthDiagnostic()
engine.reset()
engine.run()
```

Таким образом мы получили следующий код:

```
from experta import *
def ask (question):
```

```
        return (True if input(question+'[да/нет]') == 'нет' else
False)
    class Health (Fact):
        symptom = Field(str,default='')
        severity = Field(str,default='неизвестно')
        treatment = Field(str,default='неизвестно')
    class Relation(Fact):
        main = Field(str,default='')
        minor = Field(str,default='')
    class HealthDiagnostic(KnowledgeEngine):
        @DefFacts()
        def _initial_action(self):
            yield
Health(symptom='Самочувствие',severity='беспокоит')
            yield Health(symptom='Расстройство ЖКТ',
severity='неизвестно',treatment = 'Испытываете боли в области
живота?')
            yield Health(symptom='Проблемы с дыханием',
severity='неизвестно',treatment = 'Дыхание затрудненное?')
            yield Health(symptom='Боль в горле',
severity='неизвестно',treatment = 'Испытываете боли в горле?')
            yield Health(symptom='Головная боль',
severity='неизвестно', treatment = 'Испытываете головную боль?')

            yield Health(symptom='Тошнота',severity='неизвестно',
treatment = 'Вас тошнит?')
            yield Health(symptom='Насморк',severity='неизвестно',
treatment = 'Нос заложен?')
            yield Health(symptom='Кашель',severity='неизвестно',
treatment = 'Кашель беспокоит?')
            yield
Health(symptom='Головокружение',severity='неизвестно', treatment =
'Испытываете головокружение?')

            yield Health(symptom='Рвота',severity='неизвестно',
treatment = 'Была ли рвота?')
            yield Health(symptom='Обморок',severity='неизвестно',
treatment = 'Приступы обморока были?')

            yield Relation(main='Самочувствие',minor='Головная
боль')
```

```
            yield
Relation(main='Самочувствие',minor='Расстройство ЖКТ')
            yield Relation(main='Самочувствие',minor='Проблемы с
дыханием')
            yield Relation(main='Самочувствие',minor='Боль в
горле')

            yield Relation(main='Расстройство
ЖКТ',minor='Тошнота')
            yield Relation(main='Проблемы с
дыханием',minor='Насморк')
            yield Relation(main='Боль в горле',minor='Кашель')
            yield Relation(main='Головная
боль',minor='Головокружение')

            yield Relation(main='Тошнота',minor='Рвота')
            yield Relation(main='Головокружение',minor='Обморок')

        @Rule(AS.checked << Health(severity='проверяем'),
salience=0 )
        def checksymptom(self,checked):
            s = ('не беспокоит' if ask (checked['symptom']+' '
+checked['treatment']) else 'беспокоит' )
            self.modify( checked, severity = s )
            if s == 'беспокоит':
                print(checked['symptom'],' - необходимо
исследовать более подробно')
        @Rule(Health(symptom = MATCH.devname,severity='не
беспокоит'), Relation(main = MATCH.devname , minor =
MATCH.minor),AS.minorpart << Health(symptom = MATCH.minor,
severity= 'неизвестно'), salience=10)
        def markrelation(self, minorpart):
            self.modify( minorpart, severity = 'не беспокоит')
            print('Полагаю,',minorpart['symptom'],' тоже не
беспокоит.')
        @Rule(Health(symptom = MATCH.devname,severity='не
беспокоит'), Relation(main = MATCH.devname , minor =
MATCH.minor),AS.minorpart << Health(symptom = MATCH.minor,
severity= 'неизвестно'), salience=10)
        def markrelation(self, minorpart):
            self.modify(minorpart, severity = 'не беспокоит' )
```

```
            print ('Полагаю,',minorpart['symptom'],' не
беспокоит.')

        @Rule(Health(symptom =
MATCH.devname,severity='беспокоит'), Relation(main = MATCH.devname
, minor = MATCH.minor),NOT( Health(symptom = MATCH.minor,
severity= 'плохое')),AS.minorpart << Health(symptom = MATCH.minor,
severity= 'неизвестно'), salience=5)
        def markrelation(self,minorpart):
            self.modify(minorpart,severity = 'проверяем' )
            print ('Проверяемый симптом:',minorpart['symptom'])
        @Rule(AS.badsymptom << Health(symptom = MATCH.devname,
severity='беспокоит'),NOT(Relation(main = MATCH.devname, minor =
MATCH.minor),Health(symptom=MATCH.minor,severity=MATCH.severity),
TEST (lambda severity : severity in ('беспокоит','неизвестно',
'проверяем'))), salience=1)
        def isbadsymptom(self,badsymptom):
            print ('Вас беспокоит:',
badsymptom['symptom'],'Необходимо дополнительное исследование')
        @Rule(NOT(Health(severity=MATCH.severity),TEST(lambda
severity : severity in ('неизвестно','проверяем'))), salience=10)
        def allchecked(self,minorpart):
            print ('Симптомы проверены')
    engine = HealthDiagnostic()
    engine.reset()
    engine.run()
```

Работа данной экспертной системы выглядит следующим образом:

```
Проверяемый симптом: Боль в горле
Проверяемый симптом: Проблемы с дыханием
Проверяемый симптом: Расстройство ЖКТ
Проверяемый симптом: Головная боль
Головная боль Испытываете головную боль?[да/нет] да
Головная боль  - необходимо исследовать более подробно
Проверяемый симптом: Головокружение
Головокружение Испытываете головокружение?[да/нет] да
Головокружение - необходимо исследовать более
подробно
Проверяемый симптом: Обморок
```

Обморок Приступы обморока были?[да/нет] да

Обморок - необходимо исследовать более подробно

Вас серьезно беспокоит: Обморок

Расстройство ЖКТ Испытываете боли в области живота?[да/нет] да

Расстройство ЖКТ - необходимо исследовать более подробно

Проверяемый симптом: Тошнота

Тошнота Вас тошнит?[да/нет] да

Тошнота - необходимо исследовать более подробно

Проверяемый симптом: Рвота

Рвота Была ли рвота?[да/нет] нет

Проблемы с дыханием Дыхание затрудненное?[да/нет] да

Проблемы с дыханием - необходимо исследовать более подробно

Проверяемый симптом: Насморк

Насморк Нос заложен?[да/нет] да

Насморк - необходимо исследовать более подробно

Вас серьезно беспокоит: Насморк

Боль в горле Испытываете боли в горле?[да/нет] да

Боль в горле - необходимо исследовать более подробно

Проверяемый симптом: Кашель

Кашель Кашель беспокоит?[да/нет] нет

ПРИЛОЖЕНИЯ

Приложение 1. Кейсы по основам криптографии

Кейс 1. Летнее путешествие

Цель: научиться использовать шифры перестановки.

Содержание: описание проблемной ситуации, теоретическая информация по теме «Шифры перестановки», вспомогательные задания по шифрам перестановки.

Требования к решению: для реализации задания требуются материалы в печатном или электронном виде, можно воспользоваться наличием компьютеров/ноутбуков. Для выполнения основного задания можно использовать электронную таблицу.

Ответ должен содержать зашифрованное или расшифрованное слово по условию задания. Решения приводить не нужно.

Предполагаемое время выполнения кейса: 30 минут в зависимости от уровня подготовленности и активности обучающихся.

Вспомогательные упражнения нужны для освоения алгоритмов шифров перестановок, их выполнение поможет обучающимся решить основное задание кейса. За выполнение всего кейса (основное и вспомогательные задания) начисляется 5 баллов в соответствие с используемой системой оценивания в школе.

1. Описание ситуации кейса и задание для выполнения

Ситуация: семья Алисы отправлялась в путешествие, название курорта родители Алисы оставили в секрете, чтобы сделать ей сюрприз. Чтобы Алиса случайно не услышала название, родители решили его зашифровать.

Задание: для того, чтобы Алиса не догадалась о названии курорта, родители зашифровали его с помощью шифра перестановки.

1	2	3	4	5	6
3	6	1	5	2	4

Они зашифровали данное название 18 раз. Зашифрованное название курорта выглядит следующим образом: МЬЛААТ

Узнайте название курорта.

2. Теоретическая информация по теме «Шифры перестановки». Шифры перестановки и их примеры

Алгоритм данного вида шифрования подразумевает под собой изменение порядка следования элементов исходного открытого текста, при этом не изменяя их самих.

Попробуем зашифровать слово ТАЙНА с помощью простого шифра перестановки. Пронумеруем каждую букву этого слова.

Т	А	Й	Н	А
1	2	3	4	5

Условимся, что буква под номером 1 перейдет на место 4, под номером 2 – на место 3, 3→1, 4→5, 5→2.

Получаем:

Т	А	Й	Н	А
1	2	3	4	5

Осуществим переход к новым местам букв (через «’» обозначается номер буквы в новом слове).

4’	3’	1’	5’	2’

Поставим буквы на новые места в порядке возрастания.

1’	2’	3’	4’	5’
Й	А	А	Т	Н

ЙААТН – зашифрованное с помощью шифра перестановки слово ТАЙНА.

Запишем условие нашей перестановки в более удобном виде, чем простое перечисление. Пусть наше условие будет таблицей из двух строк, где в верхней строке мы запишем порядковые номера букв слова ТАЙНА, а во второй – номер места, на которое буква под определенным номером должна перейти в шифротексте в соответствии с условием. (Буква под номером 1 перейдет на место 4, под номером 2 – на место 3, 3→1, 4→5, 5→2.)

Получим:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 3 & 1 & 5 & 2 \end{pmatrix}.$$

В общем виде записать шифр перестановки для открытого сообщения длины n будет выглядеть таким образом:

$$\begin{pmatrix} 1 & 2 & \cdots & n \\ i_1 & i_2 & \cdots & i_n \end{pmatrix},$$

где i_1 – номер места шифротекста, на которое попадает первая буква при заданном преобразовании, i_2 – номер места шифротекста, на которая попадает вторая буква и т. д. В верхнем ряду стоят порядковые номера букв исходного сообщения, в нижнем – числа от 1 до n в произвольном порядке.

Дешифрование при знании таблицы шифра перестановки осуществляется очень просто. Попробуем расшифровать сообщение ОАКШЛ с помощью перестановки, описанной выше.

Мы знаем, что буква, оказавшаяся на первом месте в шифртексте, стояла в открытом сообщении на третьем месте, буква на втором месте стояла на пятом и т. д. Запишем обратное преобразование:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 5 & 2 & 1 & 4 \end{pmatrix},$$

где в верхнем ряду укажем номера мест букв в шифротексте, а в нижнем – в открытом сообщении.

Получим слово ШКОЛА.

Можно заметить, что шифры простой перестановки неудобны при достаточно большом n.

3. Вспомогательные задачи по теме «Шифры перестановки»

1. Зашифруйте слово МАСШТАБ с помощью данных перестановок

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 6 & 4 & 3 & 7 & 1 & 5 & 2 \end{pmatrix}$$

2. Дешифруйте сообщение РЕЧЖТЕ, зная *шифрующую* перестановку

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 5 & 1 & 6 & 4 & 2 \end{pmatrix}$$

3. Расшифруйте данные анаграммы ЖЕЕДВНИИ. Найдите и запишите шифрующую перестановку, по которой они были преобразованы.

Кейс 2. Секретный ингредиент

Цель: научиться использовать шифры перестановки.

Содержание: описание проблемной ситуации, теоретическая информация по теме «Шифры перестановки», вспомогательные задания по шифрам перестановки, требования к решению.

Требования к решению: для реализации задания требуются материалы в печатном или электронном виде, можно воспользоваться наличием компьютеров/ноутбуков. Для выполнения основного задания можно использовать электронную таблицу.

Ответ должен содержать зашифрованное или расшифрованное слово по условию задания. Решения приводить не нужно.

Предполагаемое время выполнения кейса: 30 минут в зависимости от уровня подготовленности и активности обучающихся.

Вспомогательные упражнения нужны для освоения алгоритмов шифров перестановок, их выполнение поможет обучающимся решить основное задание кейса. За выполнение всего кейса (основное и вспомогательные задания) начисляется 5 баллов, в соответствии с используемой в школе системой оценивания.

1. Описание ситуации кейса и задание для выполнения

Ситуация: Маша и Оля участвуют в кулинарном конкурсе в одной команде. Каждая команда должна изготовить по 10 пирожков с яблоками (для каждого члена жюри). Приготовление блюд всех команд производится в одном помещении. Маша и Оля договорились изготовить по 5 пирожков каждая. У Маши есть рецепт с добавлением секретного ингредиента, который добавляет пирожкам изысканный вкус и запах. Чтобы об ингредиенте никто не узнал, Маша и Оля решили засекретить его название.

Задание: чтобы никто не догадался об ингредиенте, Маша зашифровала его название с помощью шифра перестановки:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 4 & 5 & 7 & 6 & 2 & 8 & 3 & 1 \end{pmatrix}$$

Она зашифровала данное название один раз, а затем еще 82 раза. Зашифрованное название ингредиента выглядит следующим образом: ДАОМАНРК

Помогите Оле выяснить название ингредиента.

2. Теоретическая информация по теме «Шифры перестановки». Шифры перестановки и их примеры

Алгоритм данного вида шифрования подразумевает под собой изменение порядка следования элементов исходного открытого текста, при этом, не изменяя их самих.

Попробуем зашифровать слово ТАЙНА с помощью простого шифра перестановки. Пронумеруем каждую букву этого слова.

Т	А	Й	Н	А
1	2	3	4	5

Условимся, что буква под номером 1 перейдет на место 4, под номером 2 – на место 3, 3→1, 4→5, 5→2.

Получаем:

Т	А	Й	Н	А
1	2	3	4	5

Осуществим переход к новым местам букв (через «’» обозначается номер буквы в новом слове).

4’	3’	1’	5’	2’

Поставим буквы на новые места в порядке возрастания.

1’	2’	3’	4’	5’
Й	А	А	Т	Н

ЙААТН – зашифрованное с помощью шифра перестановки слово ТАЙНА.

Запишем условие нашей перестановки в более удобном виде, чем простое перечисление. Пусть наше условие будет таблицей с двумя строками, где в верхней строке мы запишем порядковые номера букв слова ТАЙНА, а во второй – номер места, на которые буква под определенным номером должна перейти в шифротексте в соответствии с условием. (Буква под номером 1 перейдет на место 4, под номером 2 →3, 3→1, 4→5, 5→2.) Получим:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 3 & 1 & 5 & 2 \end{pmatrix}.$$

В общем виде записать шифр перестановки для открытого сообщения длины n будет выглядеть таким образом:

$$\begin{pmatrix} 1 & 2 & \cdots & n \\ i_1 & i_2 & \cdots & i_n \end{pmatrix},$$

где i_1 – номер места шифртекста, на которое попадает первая буква при заданном преобразовании, i_2 – номер места шифртекста, на которая попадает вторая буква и т. д. В верхнем ряду стоят порядковые номера букв исходного сообщения, в нижнем – числа от 1 до n в произвольном порядке.

Дешифрование при знании таблицы шифра перестановки осуществляется очень просто. Попробуем расшифровать сообщение ОАКШЛ с помощью перестановки, описанной выше.

Мы знаем, что буква, оказавшаяся на первом месте в шифротексте, стояла в открытом сообщении на третьем месте, буква, стоящая на втором месте, – стояла на пятом и т. д. Запишем обратное преобразование:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 5 & 2 & 1 & 4 \end{pmatrix},$$

где в верхнем ряду укажем номера мест букв в шифротексте, а в нижнем – в открытом сообщении.

Получим слово ШКОЛА.

Можно заметить, что шифры простой перестановки неудобны при достаточно большом n.

3. Вспомогательные задачи по теме «Шифры перестановки»

1. Зашифруйте слово ЗАДУМКА с помощью данных перестановок.

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 6 & 4 & 3 & 7 & 1 & 5 & 2 \end{pmatrix}$$

2. Дешифруйте сообщение НЫБНАА, зная *шифрующую* перестановку.

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 5 & 1 & 6 & 4 & 2 \end{pmatrix}$$

3. Расшифруйте данные анаграммы ЛТЗООО.

Ответы к кейсам

Кейс 1. Основное задание – МАЛЬТА: 1) АШСБМТА; 2) РЕЧЖТЕ; 3) ДВИЖЕНИЕ.

Кейс 2. Основное задание – КАРДАМОН: 1) КУДАЗМА; 2) БАНАНЫ; 3) ЗОЛОТО.

Приложение 2. Кейсы по социальной инженерии

Использование информационных систем во всех сферах современной жизни помимо преимуществ, повлекли за собой появление большого ряда специфических проблем. Одной из таких проблем является необходимость обеспечения эффективной защиты информации, которая обусловлена ростом правонарушений, связанных с кражами и неправомерным доступом к данным посредством нарушения этических норм. Множество способов, психологических и социальных приемов, методов и технологий, применяемых компьютерными злоумышленниками, которые позволяют получить конфиденциальную информацию, называется *социальная инженерия.*

Социальная инженерия занимается тем, что с помощью воздействия на личность, происходит нарушение этических норм в информационной сфере. Это приводит к тому, что нарушается информационное право и, как следствие, возникает угроза информационной безопасности. Возникают проблемы в защите интересов личности и организаций. После обсуждения основных понятий по теме с опорой на нормативные правовые документы целесообразно разобрать проблемные ситуации и способы их разрешения, которые наиболее актуальны в текущий период времени, также используя правовую базу. Именно для этих целей был разработан электронный ресурс в виде кейсов, который можно использовать на уроках информатики, он расположен на сайте: https://case.kocheshkoff.ru/

Учитель может использовать кейсы двумя способами: как электронный ресурс на уроке или печатные материалы для обсуждения. Для второго способа приводятся сценарии кейсов. Для создания предлагаемых интерактивных кейсов использовалось программное обеспечение iSpring Suite, зарегистрированное в едином реестре российских программ для электронных вычислительных машин и баз данных. Был использован формат диалогового тренажера, который позволяет предусмотреть различные исходы событий благодаря созданию нескольких ветвей диалога в зависимости от выбранного варианта ответа. Кроме создания диалоговых тренажеров данный конструктор можно использовать и при создании тестов, опросов, курсов, видеолекций на основе PowerPoint. Программа iSpring Suite содержит встроенную библиотеку контента, из которой можно выбрать персонажа для диалогового тренажера и сцену. Также в этой программе можно загрузить озвучку для реплик героев.

Такой формат работы со старшеклассниками позволит реализовать предметные результаты Рабочей программы среднего общего образования по информатике.

Было разработано пять кейсов в формате диалогового тренажера:

Кейс 1. «Разочарование для шопоголика».

Кейс 2. «Продолжай в том же духе!»

Кейс 3. «Фейковый трейдинг».

Кейс 4. «Такой обманчивый 900».

Кейс 5. «Компьютерный мастер».

Кейс 1. Разочарование для шопоголика

Данный кейс направлен на формирование понятий «фишинговый сайт», «фишинговая ссылка», а также на формирование умения выявлять их с учетом изложенных в кейсе признаков.

При работе с диалоговым тренажером можно предложить обучающимся самостоятельно сформулировать определения, рассматриваемые в кейсе, признаки фишингового сайта и записать их в тетрадь. Также предложить свои способы выявления фишинга. При обсуждении предложенной ситуации можно рассмотреть другие способы получения от мошенника фишинговой ссылки и способы защиты, обсудить с обучающимися попадали ли они в схожие ситуации.

1. Предисловие. Представьте себе, что вы – успешный юрист, дающий консультации в сфере информационной безопасности, и прямо сейчас вам поступил очередной звонок.

2. Девушка. Здравствуйте! Прошу, пожалуйста, помогите мне! Я совершенно не знаю, что мне делать!

3. Юрист. Здравствуйте! Давайте попробуем разобраться с вашей проблемой. Расскажите подробнее, что произошло.

4. Девушка. Понимаете, дело в том, что я давно хотела обновить свой гардероб. Сейчас из-за больших трат в связи с подготовкой к новогодним праздникам я отложила этот вопрос, но я не могла упустить возможность закупиться новыми вещами, ведь не поверите, мне на электронную почту пришло письмо от моего любимого сайта «Ламода» о том, что сейчас действует акция: «При покупке трех вещей скидка 25%», и я, конечно же, не могла упустить такой шанс. Я долго выбирала и в итоге заказала себе пару кофточек и платье.

Когда я все оплатила, с моей карты сняли больше требуемой суммы. Значительно больше! Я в недоумении! Кроме того, мне не пришло никакого подтверждения заказа, как это было всегда! Подскажите, что мне делать?!

5. Юрист. Какую сумму списали с вашей карты?

6. Юрист. Вы сказали, что письмо пришло на электронную почту. Проверьте пожалуйста адрес отправителя, от кого пришло письмо, и продиктуйте его.

7. Юрист. Вы сказали, что это ваш любимый магазин. Часто ли вы делаете там покупки?

8. Девушка. С карты списали 20 тысяч, хотя заказ при этом был примерно на 12 тысяч.

9. Девушка. sale@larnoda.ru Х-мм, кажется, в адресе сайта ошибка – написано «Ларнода», а не «Ламода», но в первый раз я этого не заметила.

10. Девушка. Да, я пользуюсь этим магазином уже больше двух лет. Последний раз я заказывала вещи три месяца назад.

11. Юрист. Возможно, это ошибка сайта либо банковской системы. Попробуйте на сайте найти раздел «Контакты» и свяжитесь с представителем магазина для уточнения деталей произошедшего.

12. Юрист. Проверьте, пожалуйста, внимательно, при переходе по ссылке из письма, какой адрес сайта отображается в адресной строке?

13. Юрист. Я думаю, что вам и ранее приходили письма с рассылкой от этого магазина. Сравните, пожалуйста, адреса отправителей последнего письма и полученных ранее, например несколько месяцев назад.

14. Девушка. Вы знаете, я не могу найти на сайте раздел с контактами, хотя уже все просмотрела, но я точно помню, что раньше он был.

15. Девушка. Так, я вижу адрес lannoda.ru. Как такое могло произойти?

16. Девушка. Да, я нашла письмо полугодовой давности, адреса действительно различаются. В последнем письме адрес sale@larnoda.ru, а в раннем письме адрес sale@lamoda.ru.

17. Юрист. Скорее всего, вы попали на фишинговую рассылку и произвели оплату на фишинговом сайте. Это значит, что данные вашей карты попали в руки злоумышленников. В самые кратчайшие сроки позвоните в банк и заблокируйте карту, с которой была произведена оплата.

18. Юрист. К сожалению, вы не единственный человек, кто столкнулся с такой проблемой. Фишинговые рассылки и сайты сейчас очень распространены. Этим методом социальной инженерии активно пользуется большое количество мошенников.

19. Юрист. Вы можете обратиться к нотариусу за нотариальным обеспечением доказательств существования этого сайта в Интернете. Также не лишним будет самостоятельно сделать скриншоты страниц этого сайта, включая страницу контактов, если она имеется. Если мошенник известен, то можно подать исковое заявление о взыскании неосновательного обогащения в зависимости от суммы в мировой или районный суд (ГК РФ ст. 11.02).

20. Юрист. Впредь будьте внимательны: при получении подобных рассылок по СМС, на электронную почту, социальные сети, мессенджеры обращайте внимание на адрес отправителя, его имя, прикрепленные к письму ссылки и документы, грамматические, орфографические и дизайнерские ошибки на сайтах, наличие страницы контактов, отсутствие пользовательских соглашений, проверьте, что подключение к сайту защищено – используется протокол соединения https:// (указывается в начале адресной строки). Все эти признаки могут свидетельствовать о том, что сайт является фишинговым, и создан мошенниками для получения ваших личных данных, в том числе платежных (данных банковских карт и счетов).

21. Девушка. Спасибо большое за подробную консультацию! Теперь я буду внимательнее и сейчас же заблокирую карту и обращусь к нотариусу. До свидания!

22. Юрист. До свидания!

Кейс 2. Продолжай в том же духе!

Данный кейс направлен на формирование понятия «кибербуллинг», а также на формирование умения вести себя в схожей ситуации. Несмотря на то что обучающиеся должны знакомиться с этим термином еще в основной школе, в старшей школе стоит вновь возвратиться к этому понятию. Это связано с тем, что школьники должны знать и уметь правильно себя вести при столкновении с кибербуллером, которых сейчас в Интернете огромное количество. Кроме того, если ситуация не ограничивается разовыми оскорблениями, а перерастает в угрозы, преследование, то в старшей школе точно стоит говорить о правовой составляющей вопроса.

При работе с диалоговым тренажером следует обсудить само понятие «кибербуллинг»: что оно в себя включает, выработать правильную модель поведения при столкновении с обидчиком. Можно осуществить самостоятельный поиск информации о мерах наказания в нормативно-правовых актах, регулирующих различные аспекты понятия «кибербуллинг». Также стоит обсудить с обучающимися их личный опыт – сталкивались ли они с кибербуллингом в свой адрес или адрес своих знакомых.

1. Предисловие. Вам пишет друг, который живет в другом городе, и вы с ним регулярно общаетесь в Интернете.

2. Друг. Привет. Ты же знаешь, что я давно интересуюсь историей XX в., знаю много непопулярных и интересных фактов. Недавно я решил создать свой YouTube-канал и поделиться этой информацией с такими же увлекающимися историей людьми. Я уже выложил несколько видео на канал. Кстати, попозже скину ссылку – подпишись и посмотри, надеюсь, тебе понравится. В целом дела идут хорошо – у меня уже около 1000 подписчиков, я думаю, что за месяц это отличный результат! Но, не все идет так гладко, как хотелось бы…

3. Вы. Привет! Отлично, я давно тебе говорил, что тебе нужно поделиться своими находками с другими! Круто, что ты решил этим всерьез заняться. 1000 подписчиков – отличное начало, думаю дальше будет еще больше. Ты сказал, что все не так гладко. Что случилось?

4. Друг. В последнее время я заметил, что под каждым видео начали появляться комментарии от одного человека – он постоянно оскорбляет меня. Есть люди, которые пишут конструктивную критику, я им отвечаю, и в ходе дискуссии рождается истина. Но этот человек пишет абсолютно необоснованную ерунду.

5. Вы. А что именно он пишет?

6. Друг. Последний его комментарий был таким: «Голос – жуть, слушать невозможно! Все отписывайтесь, если не хотите быть как он!»

7. Вы. А что еще он тебе писал?

8. Вы. А ты не пробовал его заблокировать?

9. Вы. А что ты ему на это ответил?

10. Друг. Да много всего. Писал, что я зануда и вообще, чтобы больше ничего никогда не снимал.

11. Друг. Пробовал, он создал еще один аккаунт.

12. Друг. Пока я ничего ему не отвечал, даже не хочется вступать с ним в спор.

13. Вы. Действительно, очень глупые и необоснованные комментарии. Просто заблокируй его.

14. Вы. Я считаю, что тебе нужно ответить ему. Напиши, что раз ему не нравится – пусть просто не смотрит твои видео.

15. Вы. Это похоже на кибербуллинг. Правильно, не нужно реагировать на эту агрессию в твою сторону, это лишь заставит его писать подобные комментарии все чаще и чаще.

16. Вы. Какой настойчивый человек, это напоминает кибербуллинг. Думаю, тебе не нужно отвечать на его комментарии, просто продолжай блокировать все его новые аккаунты, и со временем ему это просто надоест, и он оставит тебя в покое.

17. Друг. Думаю, ты прав. Так и поступлю.

18. Друг. Не думаю, что это правильно. Это похоже на кибербуллинг, и если я буду отвечать на каждый такой его комментарий, то это лишь заставит писать его все чаще и чаще.

19. Вы. Кроме того, я посмотрел твои видео, это действительно очень интересно! Продолжай в том же духе. Людям это нравится, ведь на 1000 подписчиков, действительно заинтересованных твоим контентом, нашелся всего один такой завистник – ты молодец!

20. Вы. Да, ты прав, я не подумал об этом. Думаю, тебе и вправду не нужно отвечать на его комментарии, просто продолжай блокировать все его новые аккаунты, и со временем ему это просто надоест, и он оставит тебя в покое.

21. Друг. Спасибо тебе огромное за поддержку, а то я уже начал переживать из-за него. Но теперь я понял, что мои видео действительно полезные и интересные и я буду продолжать заниматься своим любимым делом.

Кейс 3. Фейковый трейдинг

Данный кейс направлен на формирование умения обращаться со своими персональными данными.

При работе с диалоговым тренажером можно предложить обучающимся перечислить: какие данные относятся к персональным; какие нормативно-правовые акты регулируют работу с персональными данными и их защиту; какая

ответственность наступает в случае их разглашения третьими лицами. Кроме того, можно обсудить, где используются персональные данные человека и возможно ли их разглашение не по вине их собственника.

1. Предисловие. Вы уже больше месяца играете в онлайн-игру с человеком, с которым познакомились в этой же игре. Во время очередного матча у вас состоялся следующий диалог.

2. Он. Видел, какой я себе инвентарь собрал?

3. Вы. Да, классно. Наверное, дорого стоит?

4. Он. Да, почти 30 тысяч все вместе стоит.

5. Вы. Откуда у тебя такие деньги?

6. Он. У меня есть друг, который уже много лет занимается обменом вещей. Выгодно обменивает несколько дешевых на дорогие. Он мне и помог.

7. Вы. Ого, классно! А он может и мне так сделать?

8. Он. Обычно, он этим за деньги занимается, но я могу у него спросить.

9. Вы. Было бы здорово.

10. Он. Я ему написал, он согласен, но ему будут нужны логин и пароль от твоего аккаунта.

11. Вы. А сколько это займет времени?

12. Вы. Окей, вот логин и пароль – grandmaster, 1is2tsvT@%sgd.

13. Вы. А ты тоже ему давал свой логин и пароль?

14. Он. По-разному, зависит от того, есть ли предложения на обмен или нет.

15. Он. Хорошо, я ему отправил данные, жди.

16. Он. Да, конечно, у него они до сих пор есть. Как только появляются выгодные предложения – он заходит и обменивает вещи.

17. Он (*прошло несколько часов*). Всё, все твои вещи мы вывели и продали, а на аккаунте играли с читами, и теперь он будет заблокирован. Спасибо за доверие, но больше не попадайся на такой развод. **Вы добавлены в черный список и больше не можете отправлять сообщения пользователю**

18. Вы. Знаешь, я так подумал, что это выглядит странно. Я бы мог и сам обменять вещи. Он может рассказать, как он это делает и на какой площадке?

19. Он. Нет, ему это не выгодно. Вдруг ты пойдешь всем остальным рассказывать?

20. Вы. Нет, я никому не буду про это рассказывать, обещаю.

21. Он. Окей, если тебе не нужен нормальный инвентарь, то так и сиди со своим старым. **Вы добавлены в черный список и больше не можете отправлять сообщения пользователю**

22. Вы. Видимо, он пытался развести меня на вещи или я вообще мог бы остаться без своего аккаунта. Хорошо, что я не отправил ему логин и пароль. Надо быть внимательнее и осторожнее при знакомстве с людьми в Интернете, даже если с ними общаешься достаточно долго.

Кейс 4. Такой обманчивый 900

Данный кейс также направлен на формирование умения обращаться со своими персональными данными. Одной из самых популярных целей мошенников являлась и является получение доступа к денежным средствам жертвы путем компрометации секретных данных банковской карты, в частности кода CVV/CVV2. Предложенная ситуация описывает именно эту схему злоумышленников, которая до сих пор является актуальной и широко применяемой.

1. Предисловие. Вам поступает телефонный звонок с короткого номера 900, вы отвечаете на него, понимая, что это номер банка.

2. Вы. Алло, здравствуйте!

3. Банк. Добрый день, меня зовут Наталья, я являюсь сотрудником службы безопасности Сбербанка. Михаил Юрьевич, с вашей карты совершен подозрительный перевод Прокофьевой Елизавете Сергеевне на сумму 4300 рублей. Вы совершали этот перевод?

4. Вы. Нет.

5. Банк. Михаил Юрьевич, у вас карта используется в основном в Москве, вы не входили в Сбербанк-онлайн из Саратова?

6. Вы. Нет.

7. Банк. Михаил Юрьевич, что мне делать с этой операцией: подтверждать или отклонять?

8. Вы. Отклонять.

9. Банк. Для отмены операции нужно вас идентифицировать. Идентифицировать можно по номеру договора, личному коду id или по номеру действующей карты. Как вы будете проходить идентификацию?

10. Вы. По номеру карты.

11. Банк. Михаил Юрьевич, номер карты необходимо по одной цифре назвать автоматизированной системе ввода данных. Сейчас я вас на нее переключу. Готовы?

12. Вы. Да, готов.

13. Робот. Автоматизированная система ввода данных Сбербанк, назовите код после звукового сигнала. **Звуковой сигнал**

14. Вы. **Называете номер своей карты**

15. Вы. **Кладете трубку**

16. Банк. Робот проверил правильность ввода. Сейчас на ваш телефон придет СМС с официального номера 900. Там будет код, который нужно тоже сообщить роботу. Это код отмены операции.

17. Вы. Да, код пришел.

18. Банк. Вы готовы его произнести? Перевожу на робота.

19. Вы. **Произносите код**

20. Банк. Спасибо, операция отменена! **Повесили трубку**

21. Вы. **На телефон приходит СМС-уведомление о списании с карты 10 000 рублей. Вы решаете позвонить в банк**

22. Сотрудник. Добрый день, меня зовут Никита, чем я могу вам помочь?

23. Вы. Добрый день, мне сейчас звонили из банка, сказали, что произведен подозрительный перевод, для отмены попросили назвать номер карты и код из СМС. После этого пришло СМС-уведомление, что с карты все равно списали 10 тысяч рублей.

24. Сотрудник. Очень сожалею, но вам звонили мошенники. Я не могу отменить перевод денежных средств, так как он уже совершен. В данной ситуации вы можете обратиться с заявлением в полицию. Чтобы таких ситуаций больше не происходило, могу предложить вам установить суточный лимит на денежные переводы. Он позволяет в течение дня переводить не более установленной суммы, отсчет суток начинается с первого перевода. Для перевода сверх установленного лимита вам будет необходимо обратиться лично в банк.

25. Вы. Хорошо, спасибо большое.

26. Сотрудник. Всего доброго, до свидания!

27. Вы. **Звоните в банк для уточнения информации**

28. Вы. **Продолжаете заниматься своими делами**

29. Сотрудник. Добрый день, меня зовут Никита, чем я могу вам помочь?

30. Вы. Добрый день, мне сейчас звонили из банка, с номера 900, сказали, что произведен подозрительный перевод, для отмены попросили назвать номер карты и код из СМС. Я положил трубку и хочу узнать, все ли в порядке с моей картой?

31. Сотрудник. Одну минуту, проверяю информацию. Да, с вашей картой все в порядке, вам звонили мошенники. Вы правильно сделали, что положили трубку. Если подобные ситуации повторятся, никогда никому не сообщайте данные своей карты и коды из СМС-уведомлений, с помощью этих данных мошенники могут списать средства с вашей карты. Кроме того, могу предложить вам установить суточный лимит на денежные переводы. Он позволяет в течение дня переводить не более установленной суммы, отсчет суток начинается с первого перевода. Для перевода сверх установленного лимита вам будет необходимо обратиться лично в банк.

32. Вывод. Вам звонили мошенники. Вы правильно сделали, что положили трубку. Если подобные ситуации повторятся, никогда никому не сообщайте данные своей карты и коды из СМС-уведомлений, с помощью этих данных мошенники могут списать средства с вашей карты. Кроме того, вы можете установить суточный лимит на денежные переводы. Он позволяет в течение дня переводить не более установленной суммы, отсчет суток начинается с первого перевода. Для перевода сверх установленного лимита вам будет необходимо обратиться лично в банк. Если же вы сообщили данные карты мошенникам, в ближайшее время позвоните сотрудникам банка для блокировки карты и обратитесь в полицию.

Кейс 5. Компьютерный мастер

Данный кейс направлен на формирование понятия обратной социальной инженерии. В старшей школе следует подробно разобрать вопросы установки программного обеспечения, которое замедляет работу компьютера, его скрытие от пользователя, принцип работы, а также обсудить правовую сторону вопроса некачественного оказания услуг мастером и последствия, которые за это наступают.

1. Предисловие. Несколько лет назад вы помогли другу собрать новый компьютер, но через некоторое время вы переехали в другой город. Связь

с другом вы поддерживаете в социальных сетях. Однажды вы получили от него следующее сообщение.

2. Друг. Привет. Не хотел тебя беспокоить по этому вопросу, но с моим компьютером происходит что-то странное. Некоторое время назад он начал тормозить, и так как я не очень в этом всем разбираюсь, я вызвал компьютерного мастера, чтобы он посмотрел, что происходит. Его номер я нашел на объявлении в своем подъезде.

3. Друг. Первое время все было хорошо. Мастер сказал, что он почистил компьютер от ненужных системных файлов, обновил драйвера. Но он предупредил, что через некоторое время могут снова начаться проблемы, потому что установлено всего 4 ГБ оперативной памяти и этого мало. Сказал, что может обновить и добавить еще, если понадобится.

4. Вы. Привет! Да, помню, когда собирал, 4 ГБ еще было достаточно для работы и для игр. И что сейчас в итоге получилось?

5. Друг. Как и сказал мастер, компьютер стал снова тормозить, я ему позвонил, он приехал и сказал, что добавил оперативной памяти до 8 ГБ. Этого будет достаточно? Потому что особого прироста быстродействия я не вижу.

6. Вы. Нужно посмотреть, какую конкретно он поставил оперативную память, на какой частоте она работает и работает ли в двухканальном режиме. Зайди в настройки системы и отправь мне скриншот.

7. Друг. **Присылает скриншот**

8. Вы. Мда… У тебя здесь так и стоит до сих пор 4 ГБ оперативной памяти, причем видимо той же самой, которую я ставил при сборке. Скорее всего, мастер поставил программу-замедлитель, которая с каждым днем все больше и больше тормозила компьютер, чтобы потом взять еще с тебя денег якобы за установку дополнительной оперативной памяти.

9. Друг. И что теперь делать?

10. Вы. У тебя остался его номер?

11. Вы. Он оставил тебе акт выполненных работ?

12. Друг. Да, номер остался. Позвонить ему?

13. Друг. Да, он оставил мне какой-то акт.

14. Вы. Да, позвони ему, попробуй узнать, что он конкретно делал в последний раз.

15. Вы. Посмотри, есть ли в акте выполненных работ строка про замену оперативной памяти.

16. Друг. Он заблокировал мой номер, не могу ему дозвониться.

17. Друг. Да, здесь есть строка про установку 8 ГБ оперативной памяти.

18. Вы. Скорее всего, тебе попался мошенник. Я читал, что это одна из техник обратной социальной инженерии, которую применяют недобросовестные компьютерные мастера. Они заставляют жертву снова и снова обращаться к ним за помощью, при этом сами закладывают проблемы и неисправности в твой компьютер.

19. Друг. И что мне теперь делать?

20. Вы. Я бы посоветовал тебе обратиться в суд, если сумма, которую ты отдал за выполнение работ достаточно большая для тебя. По закону «О защите прав потребителей» можно потребовать полного возмещения убытков либо устранения выявленных недостатков.

21. Друг. Эх, ладно… Спасибо за помощь. Пойду и попробую разобраться с этим. В следующий раз лучше проконсультируюсь с тобой.

22. Вы. Впредь будь внимательнее, обращайся в проверенный сервис, а не к частному мастеру. Жаль, что так вышло. Если что – пиши, всегда рад помочь.

Приложение 3. Практикум по трехмерному моделированию и прототипированию в среде T-FLEX CAD

В тематическом разделе «Информационные технологии» в 11 классе по теме «3D-моделирование» предусматриваются практические работы и выделено 8 часов. При наличии в школе трехмерных принтеров можно заниматься проектными работами по созданию прототипов изделий. Эти проекты могут быть реализованы в виде учебных стартапов. Предлагаем рассмотреть три направления развития темы, реализованные в виде проблемных ситуаций для обсуждения и выбора тематики проектной работы.

Ситуация 1. В школьном буфете сломалась ручная соковыжималка. Старшеклассники решили сделать ее самостоятельно на школьном трехмерном принтере. Работа делиться на две части: разработка модели в специализированной среде T-FLEX CAD и печать прототипа изделия на трехмерном принтере.

Целями данной работы являются: получение первоначальных навыков работы в профессиональной среде трехмерного моделирования и прототипирования с использованием сложного оборудования – трехмерного принтера. Предлагается практикум, который позволит сформировать необходимые компетенции. Его можно использовать в классах технологического профиля или предпрофессиональных инженерных классах.

Ситуация 2. Во время очередной эпидемии гриппа необходимо постоянно носить маски или респираторы. Респиратор имеет стандартизированные размеры и его можно напечатать на трехмерном принтере. При этом можно включить творческую составляющую, разработать внешний дизайн изделия.

Целями данной работы являются развитие навыков трехмерного моделирования в профессиональной среде трехмерного моделирования и прототипирования с использованием сложного оборудования – трехмерного принтера. Творческая составляющая деятельности обучающихся дает возможность реализовать результат как продуктовый результат проектной деятельности.

Ситуация 3. Обилие разнообразных чехлов для сотовых телефонов подтолкнуло к идее изготовления таких чехлов на трехмерном принтере, организации творческого конкурса и возможности проработать идею стартапа по данной тематике.

Целями данной работы являются возможность продолжить развитие проектной идеи, организовать команду для дальнейшего его усовершенствования, привлечь инвесторов, защитить обновленный проект, реализовать его в виде бизнес-идеи.

Подготовка к выполнению практикума

Установка программы. Перед началом работы скачаем программу. Получить установочные файлы T-FLEX CAD можно, перейдя на официальный сайт разработчика: https://www.tflexcad.ru/download/t-flex-cad-free/files.php

Прежде чем начать установку необходимо проверить соответствие вашего компьютера системным требованиям (рис. 12).

Минимальные	
Операционная система:	Windows 7 x64 (с Пакетом обновлений 1)
Процессор:	Intel или AMD с поддержкой SSE3
Объем оперативной памяти:	2 Гб
Объем свободного дискового пространства:	3 Гб
Видеокарта:	видеокарта с поддержкой OpenGL 3.3 и выше
Рекомендуемые	
Операционная система:	Windows 8.1x64, 10 x64
Процессор:	Core i5 или выше
Жёсткий диск:	SSD накопитель
Объем оперативной памяти:	16 Гб и больше
Видеокарта:	высокопроизводительная видеокарта NVIDIA или AMD с памятью 1Гб и выше, а также поддержкой OpenGL 4.2 и выше

Рис. 12. Системные требования для установки T-FLEX CAD

Если компьютер соответствует хотя бы минимальным требования, можно начать установку. Необходимо загрузить архив с инсталляцией образовательной версии T-FLEX CAD (пакетный установщик) на свой компьютер, распакуйте его во временную папку, запустите файл **Setup.exe** и следуйте указаниям программы установки. Обязательно после установки установите Компоненты поддержки и Библиотеки стандартных элементов, для этого нужно запустить файл **Setup.exe** из каталога «Компоненты поддержки T-FLEX» и следовать указаниям программы установки.

Когда установка буде завершена, программа автоматически откроется. При запуске учебной версии T-FLEX CAD всегда появляется окно с активацией программы. Для дальнейшей работы нажимаем кнопку «Продолжить» (рис. 13).

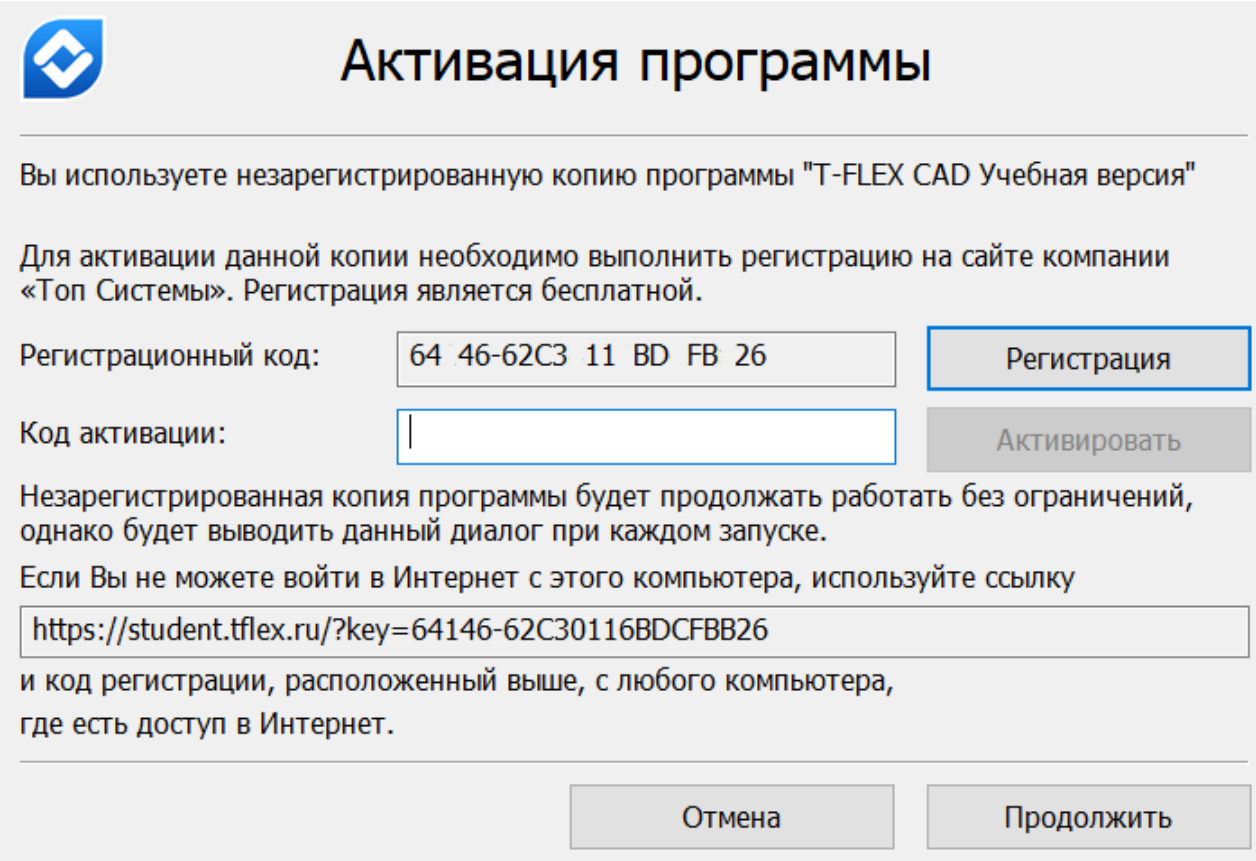


Рис. 13. Активация учебной версии T-FLEX CAD

После проделанных шагов появится страница приветствия, в центральной части которой будут располагаться прототипы для создания чертежей, деталей и других документов (рис. 14). На панели слева будут отображаться уже созданные проекты (при первом запуске она, естественно, будет пуста).

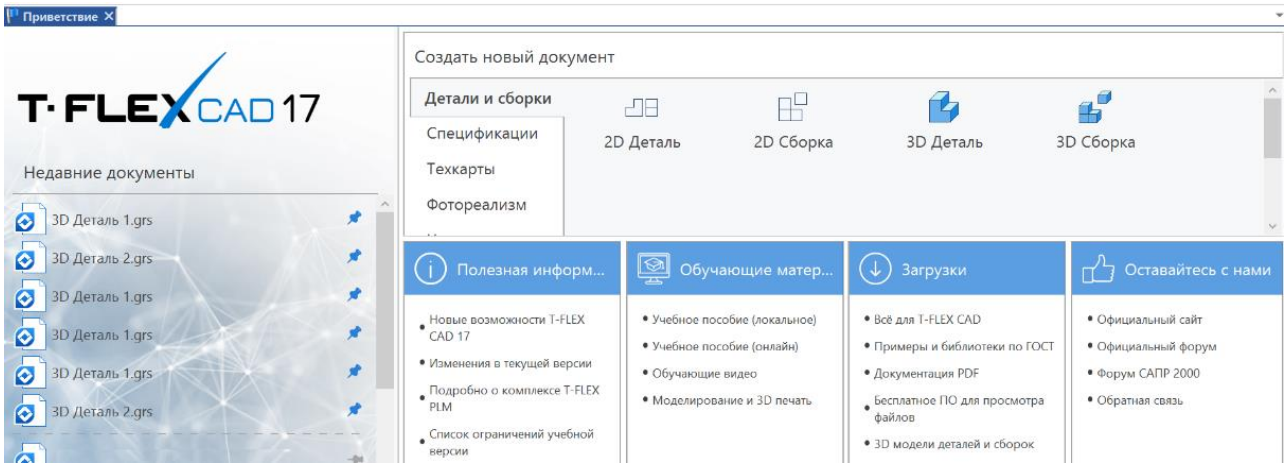


Рис. 14. Страница приветствия

Теперь у нас есть все инструменты, необходимые для того, чтобы создавать собственные работы.

Практическая работа «Ручная соковыжималка»

Цель работы: создать модель кухонного прибора для выжимания сока из плодов цитрусовых (рис. 15).

Рис. 15. Общий вид готовой модели соковыжималки

Шаг 1. Создание конуса цитрус-пресса

На странице приветствия перейдем на вкладку **Детали и сборки** и левой кнопкой мыши кликнем на иконку 3D-деталь**.**

1.1. Начнем создание нового скетча (эскиза, чертежа) с помощью команды **Чертить** из группы **Построения**. Активируем команду.

1.2. Выберем плоскость «Вид слева» (рис. 16), так как мы будем изображать вид сбоку, а затем «крутить» объект вокруг пространственной оси, совершая полный оборот (360°).

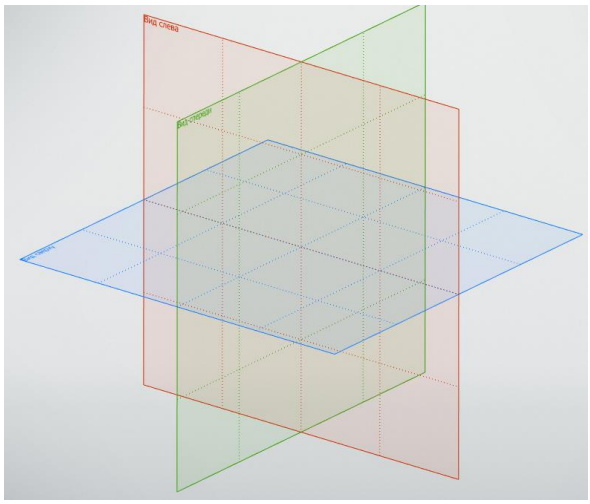


Рис. 16. Выбор плоскости для построения эскиза

1.3. Активировав команду **Отрезок** из группы **Эскиз,** изобразим перпендикулярные прямые с размерами, приведенными на рисунке 6. Соединим концы отрезков, чтобы получить замкнутый контур, используя команду **Сплайн** по точкам из группы **Эскиз** (рис. 7). Выберем команду **Закончить задание**

сплайна (пиктограмма маленькой зеленной галочки), затем нажмем на кнопку **ESC** на клавиатуре. Получился прямоугольный треугольник (с двумя катетами и гипотенузой).

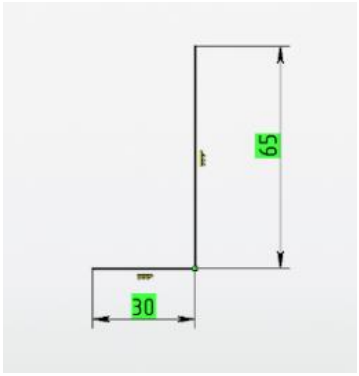


Рис. 17. Построение перпендикулярных прямых

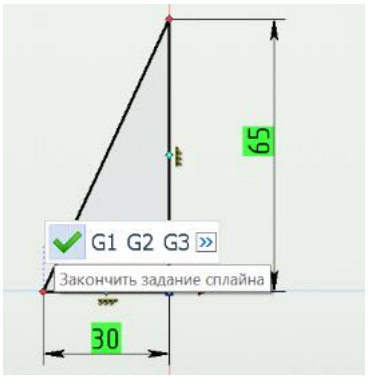


Рис. 18. Создание замкнутого контура

1.4. Придадим гипотенузе треугольника «кривизну», задав угол поворота конечных точек сплайна. Левой кнопкой мыши нажмем на созданную гипотенузу треугольника. В левой части от рабочей области появится окно **Параметры команды**, которую мы использовали на предыдущем шаге. На вкладке **Параметры сплайна** левой кнопкой мыши нажмем на первую точку, выделенную на рисунке 19 голубым цветом, и на вкладке **Параметры точки** зададим угол 90°. После задания угла для точки в **Параметрах сплайна** для первой точки появиться символ G1, обозначающий создание граничных условий.

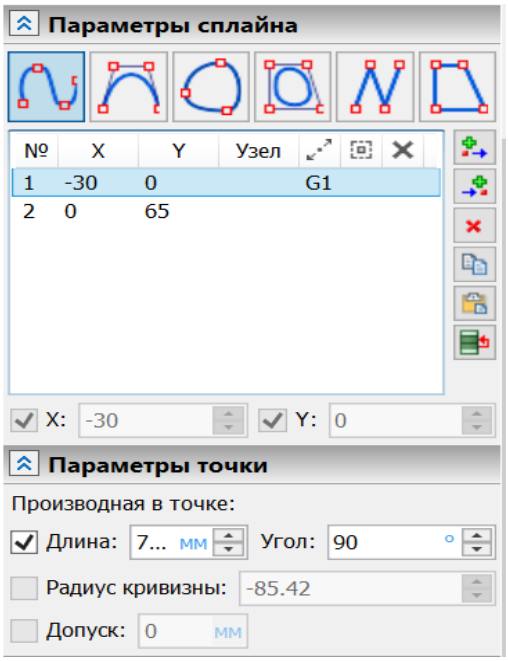


Рис. 19. Задание угла поворота первой точки

Щелкаем левой кнопкой мыши на вторую точку в окне **Параметры сплайна**, выделенную на рисунке 20 голубым цветом, и в **Параметрах точки** задаем угол 45°. Выберем команду **Завершить,** чтобы завершить черчение на рабочей плоскости (располагается на ленте инструментов на вкладке *Главная*)**.**

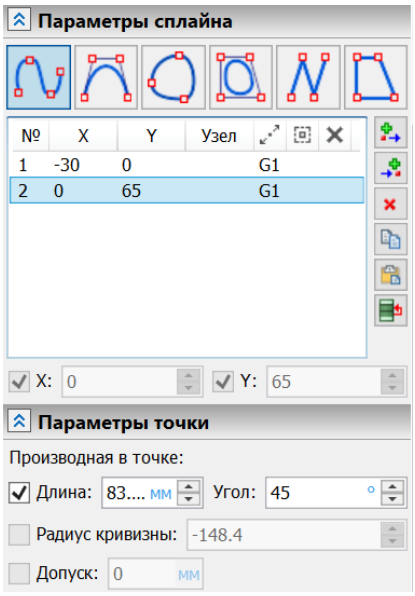


Рис. 20. Задание угла поворота второй точки

1.5. Далее выберем операцию **Вращение** из группы **Операции** и выделим контур полученного профиля, а затем его вертикальную линию в качестве оси вращения. Во вкладке **Основные параметры** операции зададим угол поворота – 360° (рис. 21.).

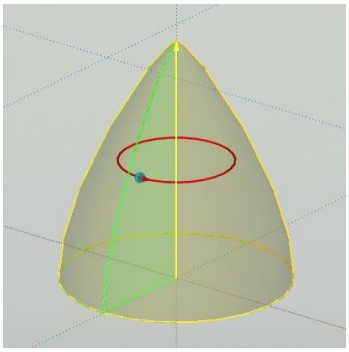


Рис. 21. Вращение профиля

1.6. Для создания основания соковыжималки выберем рабочую плоскость «Вид спереди». Активировав команду **Отрезок** из группы **Эскиз**, от центра конуса отложим отрезки так же, как показано на рисунке 22. Размеры отрезков на рисунке выделены зеленным цветом. После создания отрезков завершим команду.

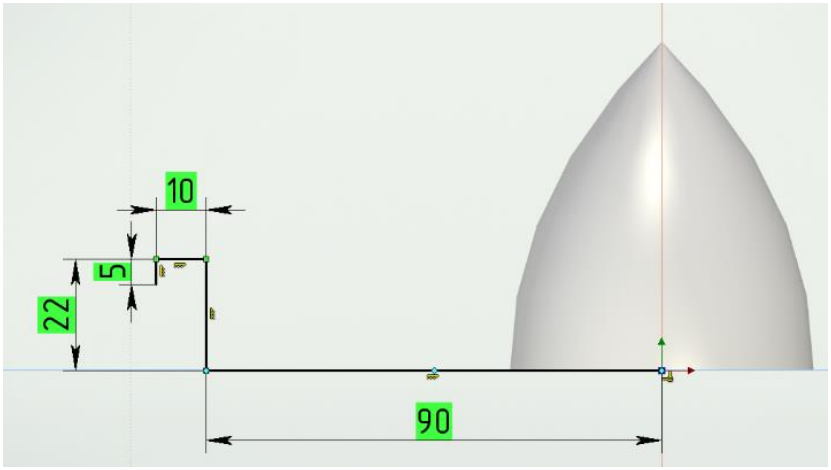


Рис. 22. Эскиз основания соковыжималки

1.7. Закруглим углы, активировав команду **Скругление** из группы **Эскиз**. Суть команды **Скругление** заключается в выборе двух отрезков и определении параметра скругления.

Для скругления угла между стенкой и дном основания соковыжималки, задается радиус, равный 15 мм. Параметр скругления верхнего правого угла – 5 мм, а левого угла – 4 мм (рис. 23, 24).

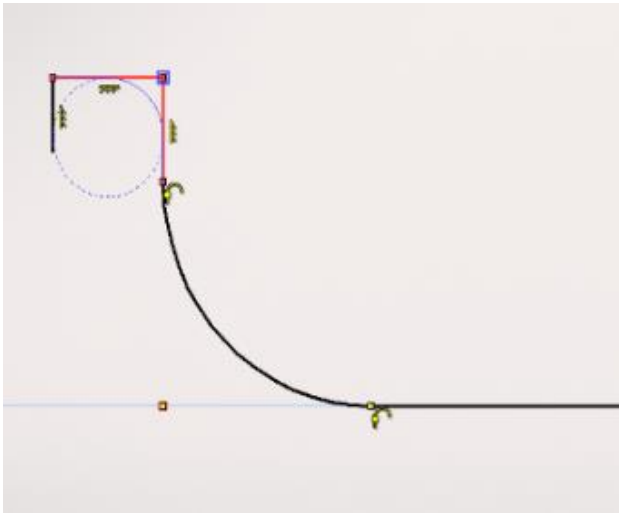


Рис. 23. Скругление нижних углов

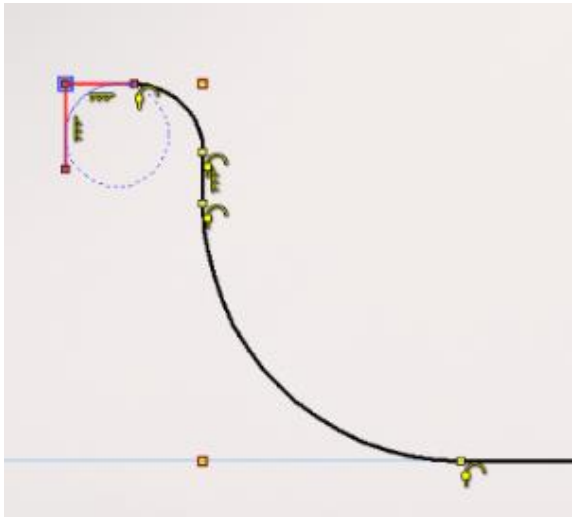


Рис. 24. Скругление верхних углов

1.8. Выберем получившийся профиль и применим к нему операцию **Вращение** из группы **Операции**. Сначала левой кнопкой мыши нажмем на созданный профиль, а затем щелкнем на конус, выбрав его в качестве объекта, вокруг которого будет «вращаться» профиль. В окне **Параметры** операции вращения необходимо указать, что создается тонкостенный элемент. Для этого необходимо задать параметры, указанные на рисунке 25.

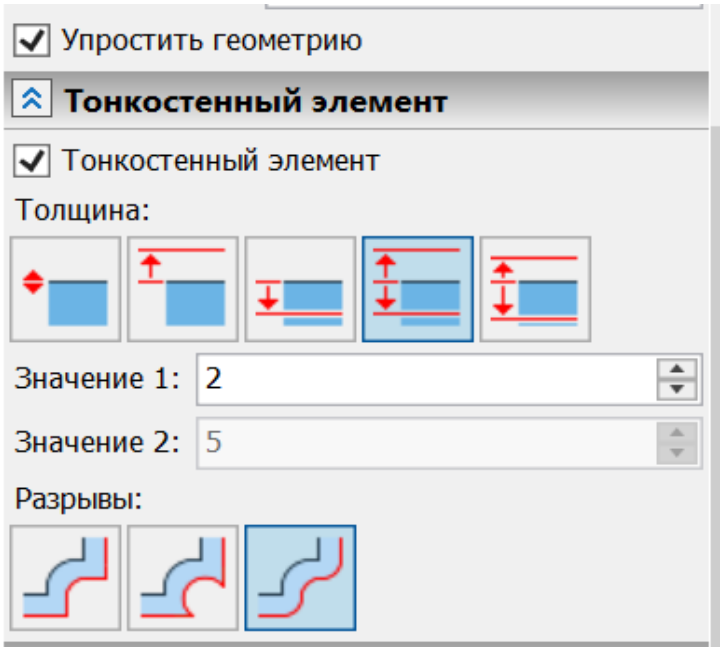


Рис. 25. Выбор параметров операции вращения профиля

1.9. Готово! Осталось добавить отверстия на нижней поверхности конуса и на самом конусе.

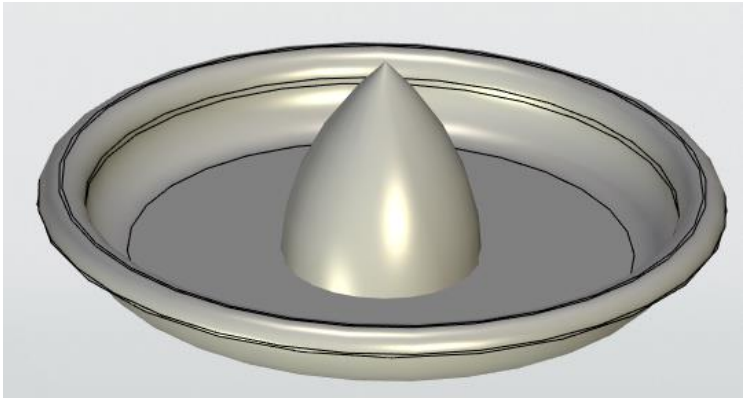


Рис. 26. Основное тело соковыжималки

Шаг 2. Создание отверстий и ребристых граней

2.1. Для создания вырезов нам понадобится какое-нибудь другое тело. Пусть в нашем случае это будет цилиндр. Откроем группу **Специальные** и выберем операцию **Примитив** из группы **Специальные** (рис. 27).

2.2. Цилиндр создается автоматически в исходной системе координат. В окне **Параметры** отредактируем размеры созданного цилиндра, изменив значения диаметра на 8 мм, а высоту на 100 мм.

2.3. С помощью манипуляторов **Перемещение**, расположенных в нижней части фигуры, переместим цилиндр к конусу таким образом, чтобы поверхности

тел соприкасались. Для этого из центра необходимо переместить Цилиндр вправо на 35 мм, затем повернуть его вокруг оси *X* на 22° и по оси *Y* поднять на 7 мм, как показано на рисунке 28.

При перемещении манипулятора значение изменяемого размера отображается в окне **Параметры операции** , на вкладке **Преобразования**.

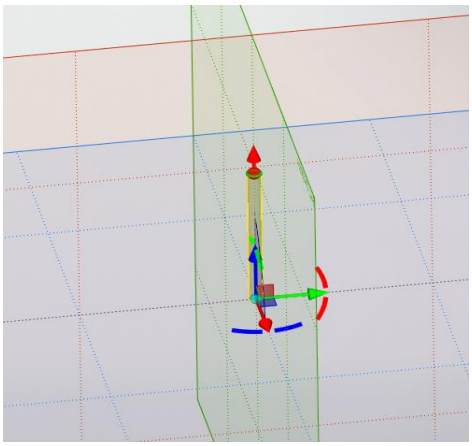


Рис. 27. Создание цилиндра

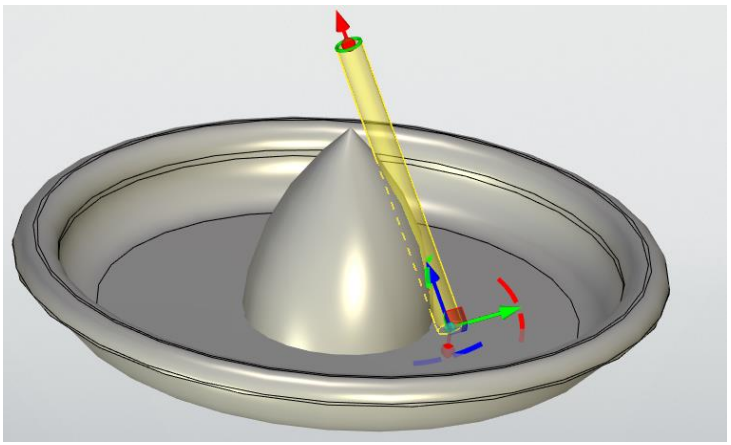


Рис. 28. Расположение цилиндра

2.4. Далее активируем операцию **Булева** из группы **Операции** и последовательно щелкнем левой кнопкой мыши сначала по конусу, потом по цилиндру. При этом в окне **Параметры оперции** выберем режим **Вычитание**. Зеленым цветом подсвечивается элемент, из которого удаляют тело, а желтым – удаляемое тело, т. е. сам цилиндр. (рис. 29).

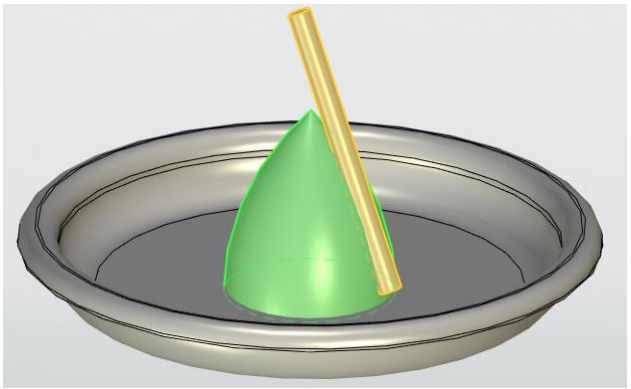


Рис. 29. Вычитание из конуса цилиндра

2.5. В итоге мы получили элегантный вырез (рис. 30). Если хочется изменить форму выреза, то можно выбрать другой примитив, проделав пп. 2.1–2.4, или можно вернуться по шкале действий к моменту расположения цилиндра и «подвинуть» его в другое место. Изменения коснутся также и дальнейших шагов.

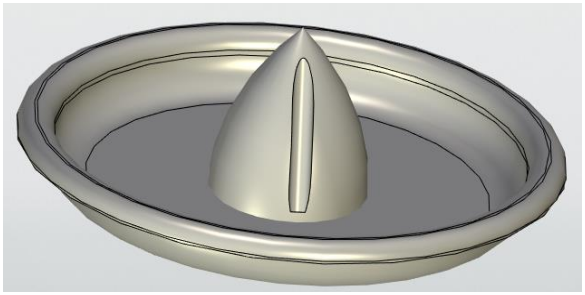


Рис. 30. Результат выреза

2.6. Остается «размножить» отверстия по поверхности конуса. Для этого активируем операцию **Круговой массив** из группы **Операции**. В окне **Параметры**, на вкладке **Основные параметры,** укажем свойство «массив граней». Щелкнем левой кнопкой мыши по вырезу.

В окне **Параметры операции** укажем количество копий – 8 шт., а в качестве оси вращения выберем конус (рис. 31). Программа автоматически рассчитает шаг между создаваемыми копиями выреза и угол вращения, как показано на рисунке 31. Выберем команду **Закончить ввод** и увидим получившийся результат.

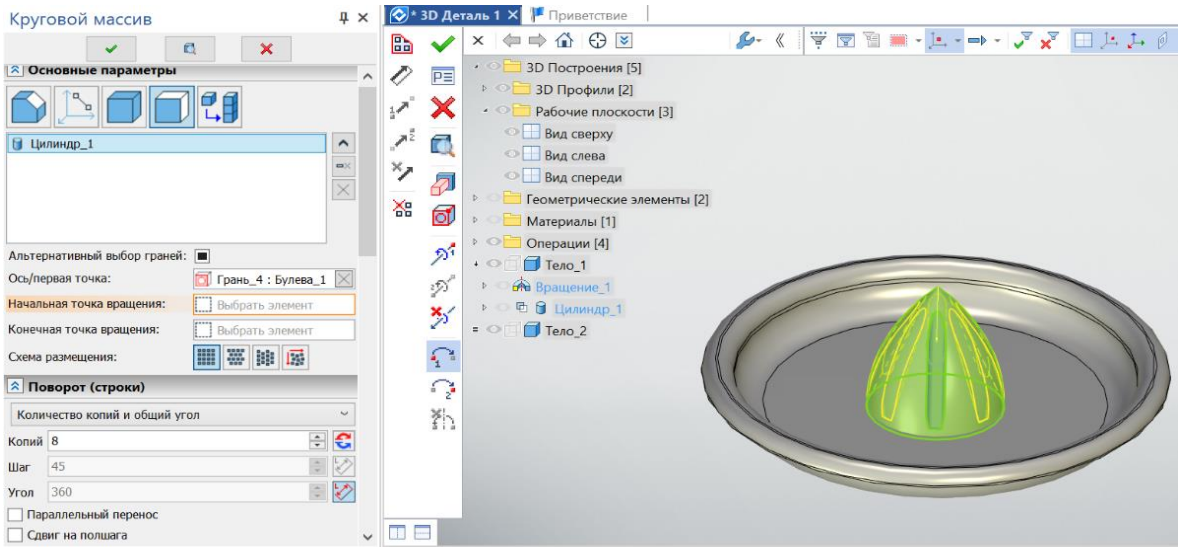


Рис. 31. Массив граней

2.7. Скруглим ребра, чтобы сок от фруктов не скапливался в углах конусной поверхности. Для этого выберем операцию **Сглаживание ребер** из группы **Операции**, левой кнопкой мыши щелкнем по нижним ребрам отверстий; они выделены зеленым цветом на рисунке 32. В окне **Параметры операции** укажем радиус – 2 мм, выберем вид операции «скругление», активируем параметры: удалить элементы, продолжить по касательной и выбор граней ребер и вершин. Выберем команду **Закончить ввод**.

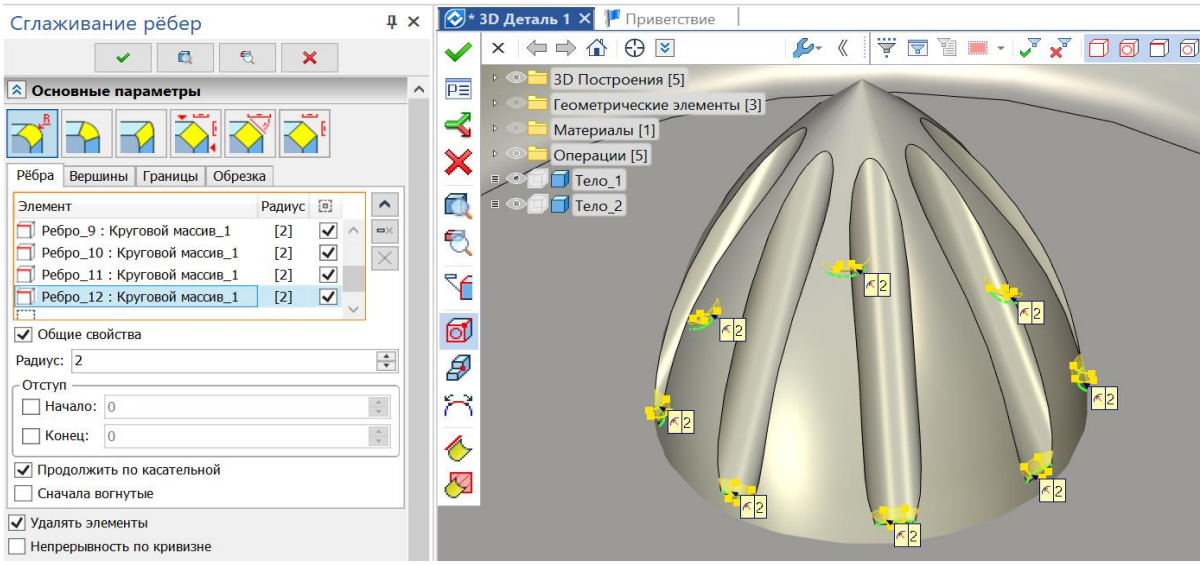


Рис. 32. Сглаживание ребер

2.8. Готово!

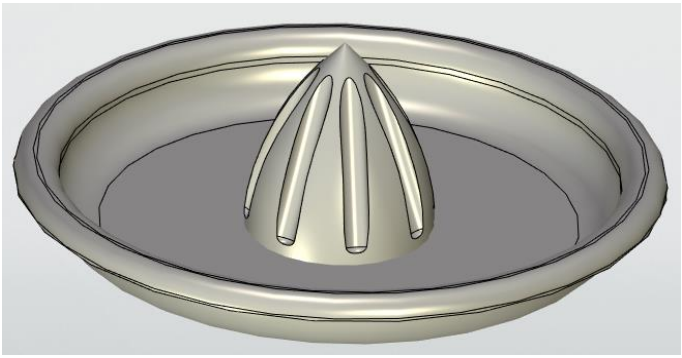


Рис. 33. Готовый конус цитрус-пресса

2.9. Выберем внутреннюю поверхность основания конуса в качестве рабочей поверхности для будущего эскиза (рис. 34). Для этого активируем команду **Чертить** из группы **Построения** и левой кнопкой мыши щелкнем по нужной поверхности.

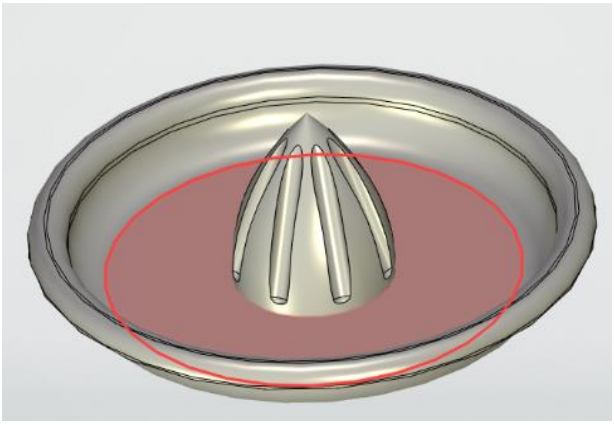


Рис. 34. Выбор рабочей поверхности

2.10. Создадим две перпендикулярные прямые с узлом в точке пересечения в начале координат. Для этого раскроем дополнительные параметры команды **Прямая** из группы **Построения** и выберем команду **Прямые с узлом (0, 0)**. Программа автоматически создаст две прямые (рис. 35).

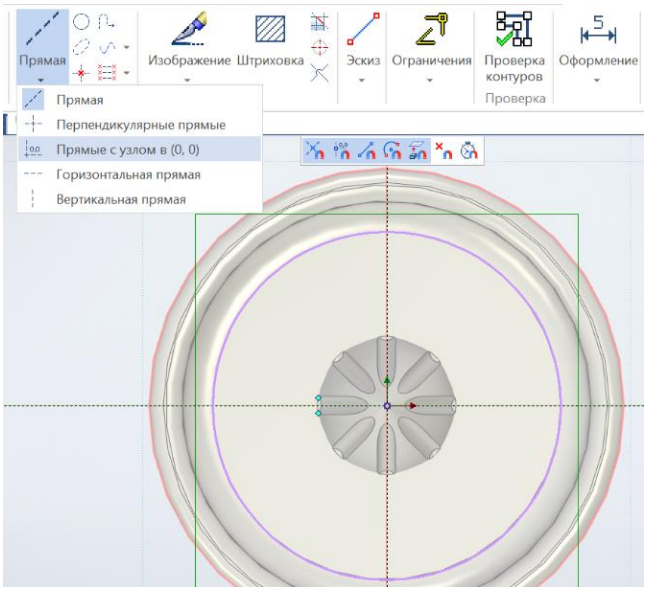


Рис. 35. Прямые, пересекающиеся в начале координат

2.11. Затем активируем команду **Паз** из группы **Эскиз** и создадим фигуру с радиусом 2,5 мм, длинной 30 мм и шириной 5 мм. Размеры фигуры необходимо задать в окне **Параметры паза**. Важно, чтобы эскиз паза находился сверху от горизонтальной прямой, как на рисунке 36.

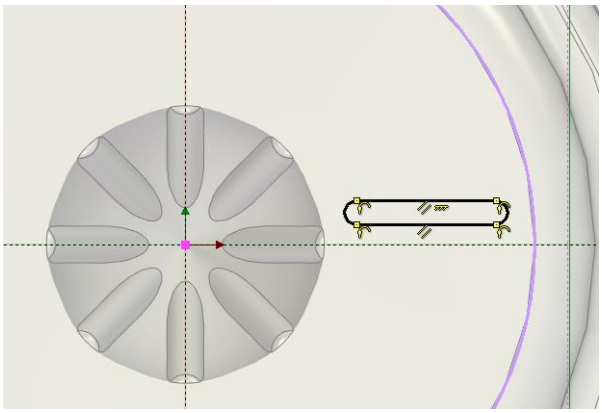


Рис. 36. Создание паза

2.12. С помощью команды **Размер** из группы **Оформление** зададим расстояние от центра до паза. Для этого активируем команду **Размер**, левой кнопкой мыши щелкнем на вертикальную прямую, а затем на середину левого края паза (рис. 37).

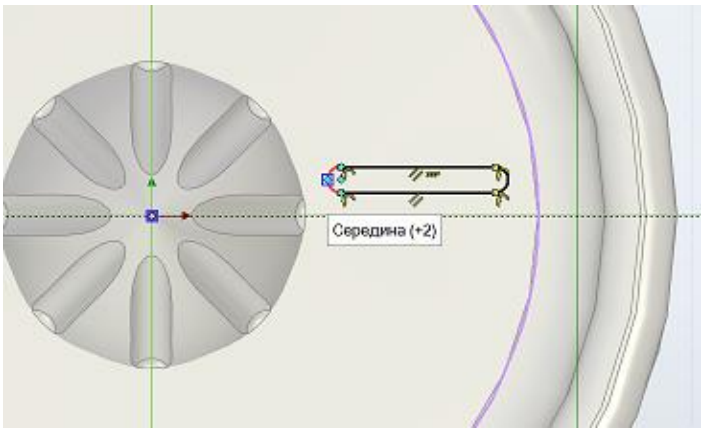


Рис. 37. Выбор элементов эскиза

Автоматически рассчитается расстояние между элементами, значение которого можно изменить. Нажмем левой кнопкой мыши на поверхность соковыжималки, а затем в появившемся окне укажем значение 38 мм (рис. 38). Программа автоматически пересчитала расстояние между вертикальной прямой и пазом, сдвинув элементы друг от друга на заданное расстояние. Нажмем на зеленую галочку, а затем выберем команду **Завершить**.

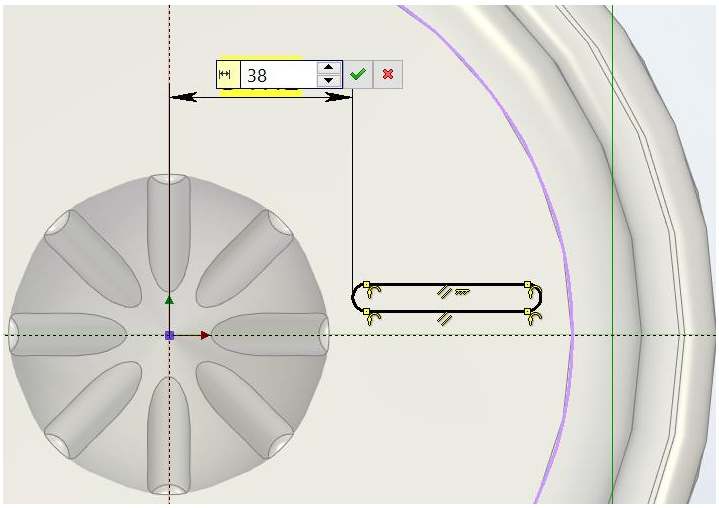


Рис. 38. Выбор элементов эскиза

2.13. С помощью операции **Выталкивание** из группы **Операции** создадим отверстие. Активируем операцию **Выталкивание**, затем левой кнопкой мыши щелкнем по созданному профилю паза. В окне параметров операции укажем направление «Симметрично по общей длине», а также длину выталкивания – 10 мм. Включим режим булевой операции в активной вкладке параметров операции, также активируем опцию **Выбрать исходное тело для булевой операции**. Нажмем левой кнопкой мыши на нижнюю поверхность соковыжималки (рис. 39). Далее нажмем на команду **Закончить ввод**.

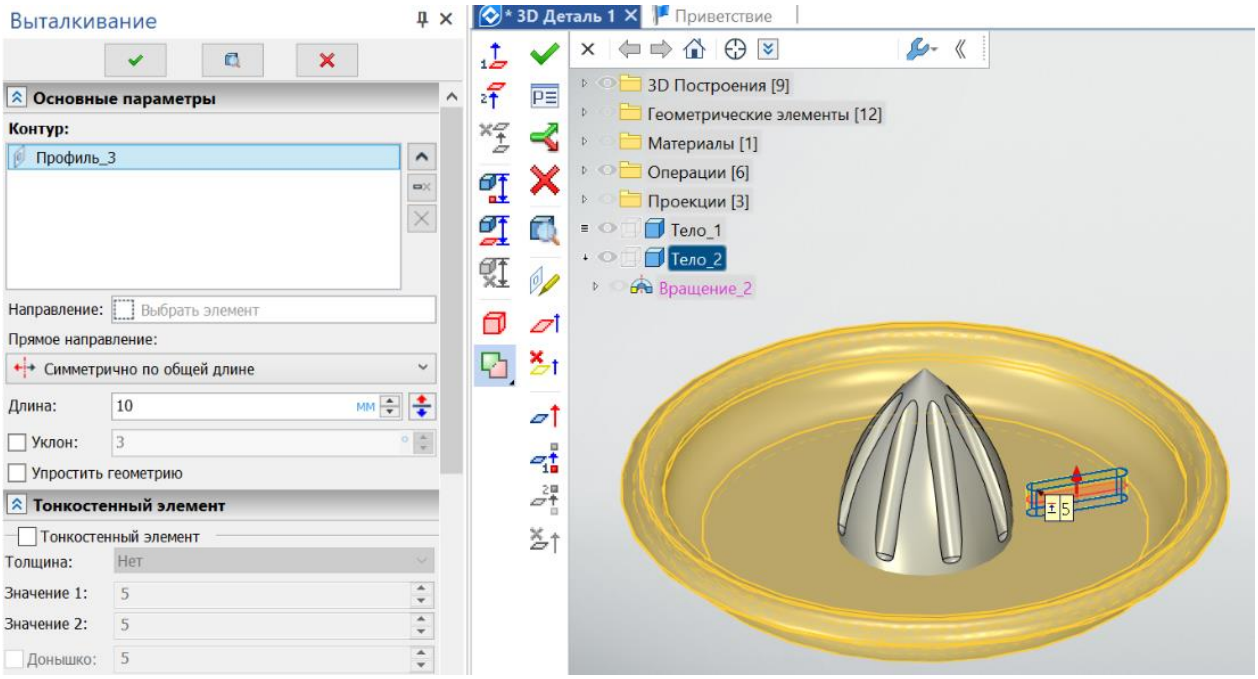


Рис. 39. Создание отверстия

2.14. Мы получили отверстие в форме паза (рис. 40). Создадим копии отверстия с помощью операции **Круговой массив граней**. Стоит учитывать, что при создании массива граней никогда не создается новое 3D-тело – только трансформируется уже существующее.

Рис. 40. Отверстие в форме паза

2.15. Активируем операцию **Круговой массив** из группы **Операции**. Левой кнопкой мыши щелкнем на внутреннюю стенку паза, выделенную красным цветом на рисунке 41. Затем левой кнопкой мыши щелкнем на конус, вокруг которого будут создаваться копии отверстия. В окне **Параметры операции** должен быть указан режим «Массив граней», «Стандартная» схема размещения, а также количество копий – 16 шт. Автоматически программа покажет предварительный результат операции (см. рис. 41).

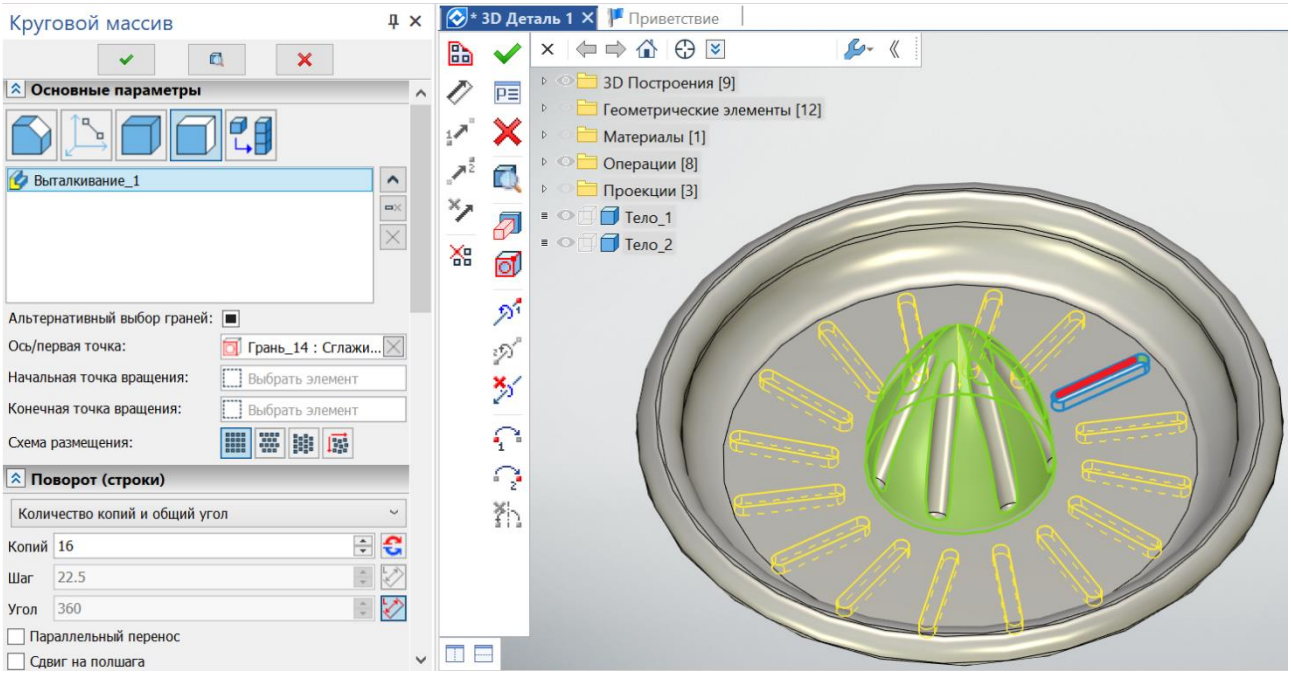


Рис. 41. Предварительный результат создания копий паза

2.16. Выберем команду **Закончить ввод**. Ручная соковыжималка для цитрусовых фруктов готова (рис. 42). Осталось применить к ней материал, из которого прототип изделия будет изготавливаться, а также сделать фотореалистичное изображение.

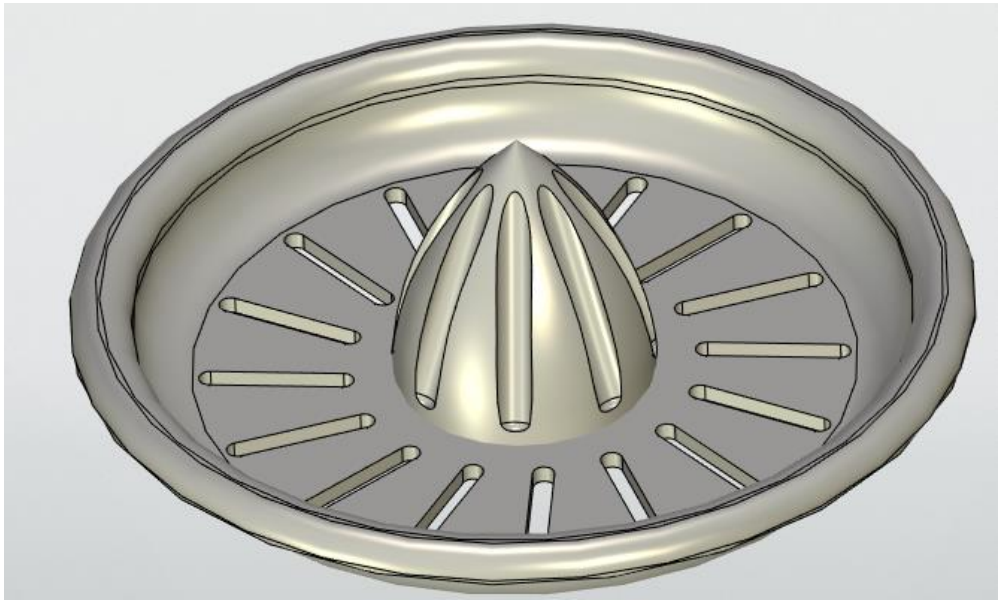


Рис. 42. Модель соковыжималки для цитрусовых

Шаг 3. Применение материалов для создания фотореалистичного изображения

3.1. Переключимся в режим **Фотореалистичного вида** (рис. 43).

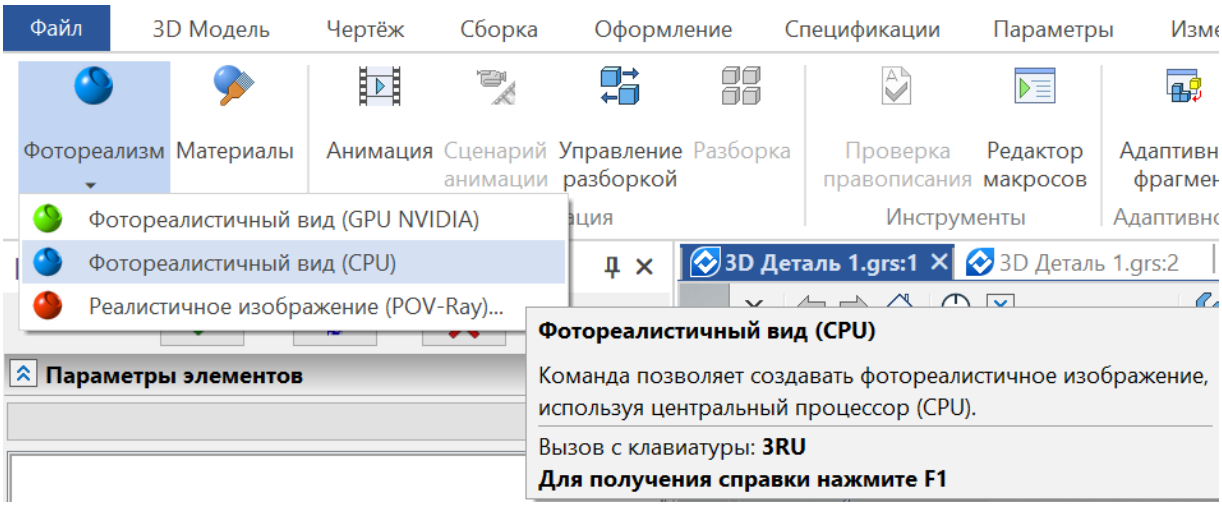


Рис. 43. Режим **Фотореалистичный вид**

3.2. Рассмотрим основные кнопки на панели инструментов в данном режиме (рис. 44).

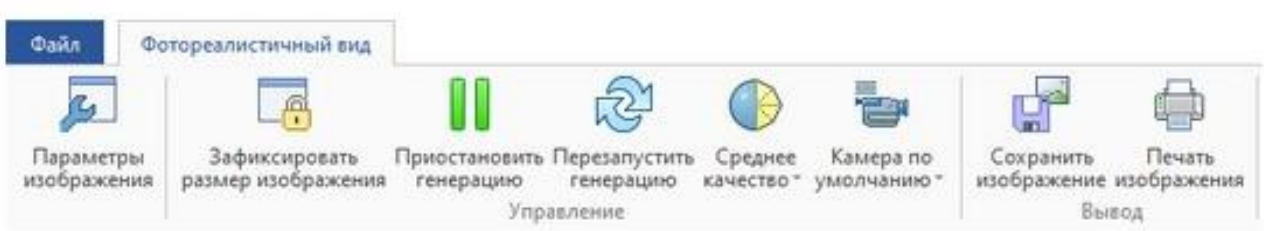


Рис. 44. Панель инструментов в режиме **Фотореалистичный вид**

1. **Параметры изображения** позволяют задавать параметры генерации изображения. Более подробное описание опции дано ниже.

2. Кнопка **Зафиксировать размер изображения** позволяет зафиксировать направление взгляда и масштаб изображения. Вращение модели становится невозможным.

3. Функция **Приостановить генерацию** позволяет временно прекратить генерацию изображения. При этом высвобождаются затрачиваемые на этот процесс ресурсы компьютера, в связи с чем повышается производительность.

4. Функция **Перезапустить генерацию** запускает генерацию фотореалистичного изображения заново, при этом происходит сброс текущих результатов.

5. Кнопкой **Выбор качества генерируемого изображения** в выпадающем списке можно выбрать одно из четырех значений качества изображения.

Низкое и среднее качество применяются для черновых вариантов изображений. При выборе такого качества система автоматически высчитывает минимальное количество итерации, необходимое для получения изображений с определенным уровнем «шумов».

Для получения наиболее реалистичных изображений нужно выбрать высокое или максимальное качество. При максимальном качестве количество итераций не ограничено.

Важную роль в создании реалистичных изображений играет настройка Качество изображения. Ее можно изменить с помощью выпадающего списка в окне **Параметры документа** на закладке 3D (см. выше). Чем выше качество, тем выше плотность сетки. Для получения наиболее реалистичных изображений рекомендуется устанавливать качество не ниже, чем «Повышенное». Данный параметр особенно важен при наличии в модели скругленных поверхностей.

6. Функция **Выбор текущей активной камеры** позволяет выбрать одну из присутствующих в 3D-сцене камер. Изображение будет создаваться в соответствии с положением выбранной камеры.

7. Кнопка **Сохранить изображение** позволяет экспортировать полученное изображение в файлы растровых форматов *.bmp, *.jpg, *gif, *tiff, *tif, *.png, *.tga. Для файла можно задать имя и указать, где он будет храниться.

8. Кнопка **Печать изображения** позволяет вывести получаемое изображение на печать.

3.3. Выберем операцию **Наложение материалов** из группы **Расширенные**. Если данная операция отсутствует на панели инструментов, то ее можно найти, используя инструмент **Поиск команд,** который изображен в виде лупы. Появиться поисковая строка, в которую необходимо ввести название инструмента, операции или команды.

После вызова операции **Наложение материалов** в автоменю становятся доступны опции, позволяющие выбирать грани тела. Автоматически включается опция **Выбрать грань.**

3.4. Выберем все грани тела. Для этого зажмем правую кнопку мыши и переместим курсор мыши справа налево. Необходимо, чтобы все элементы 3D-модели подсветились зеленным цветом. В окне параметров наложения материала появятся название выбранных нами граней (рис. 45).

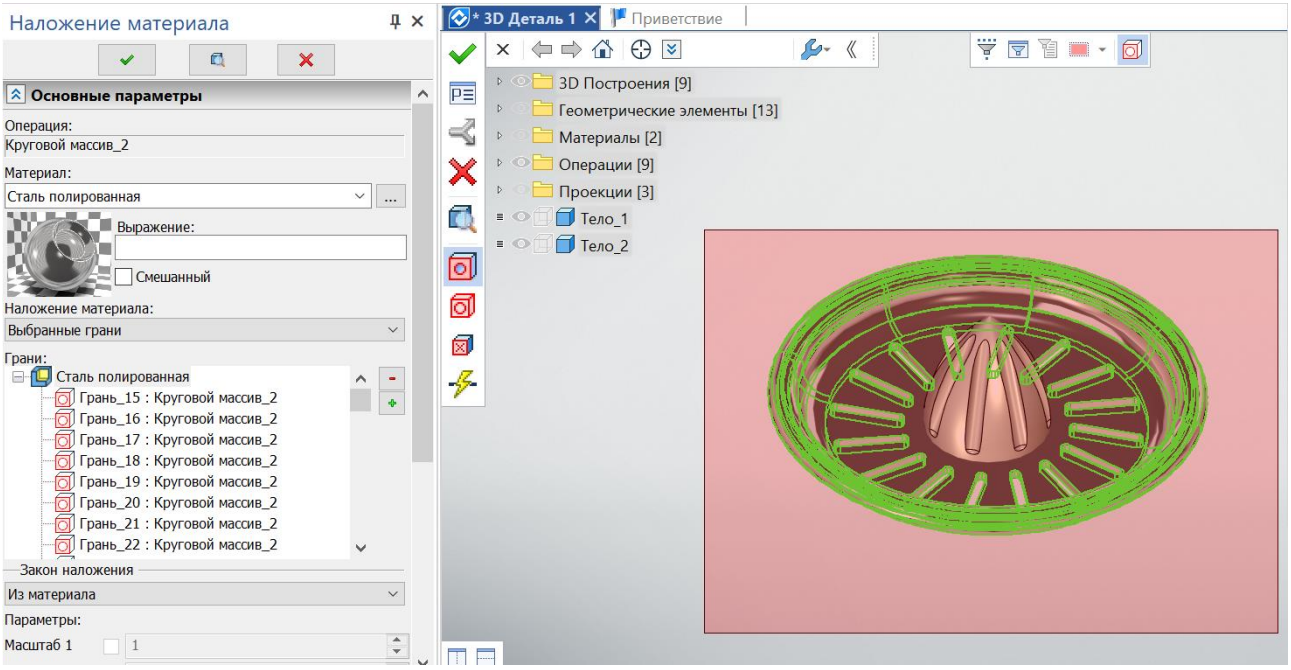


Рис. 45. Выбор граней тела

3.5. Откроем список материалов из вкладки основных параметров, найдем папку «Пластик» → Оранжевый пластик (матовый) (рис. 46).

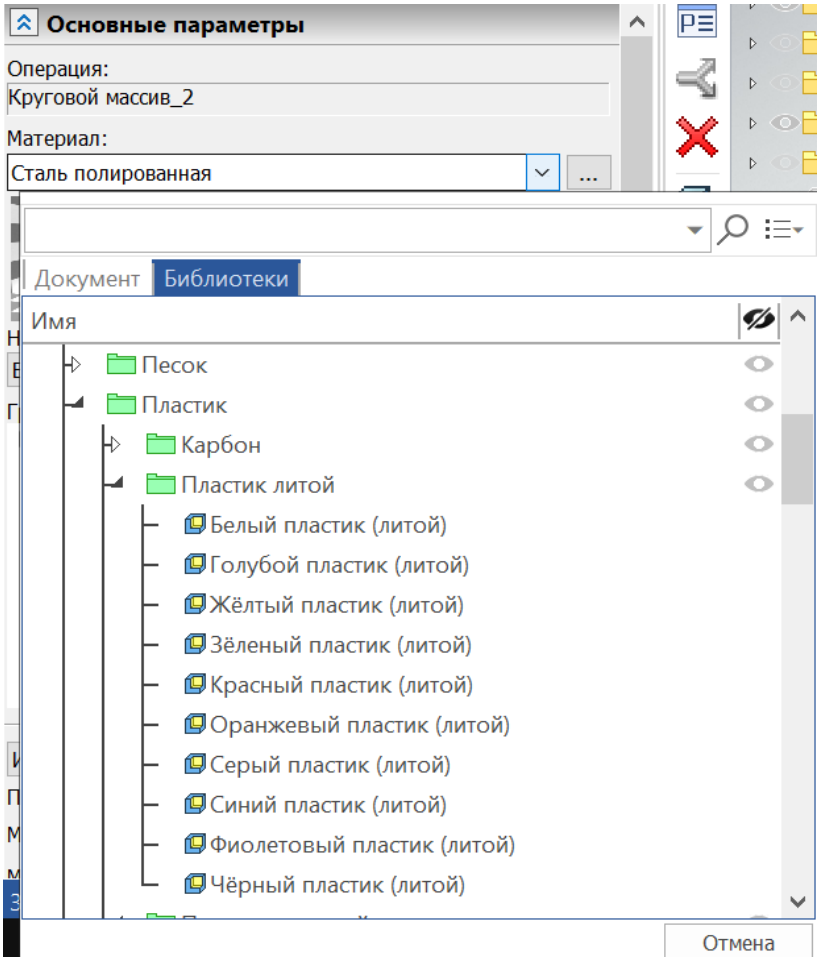


Рис. 46. Выбор материала

3.6. Дважды щелкнем левой кнопкой мыши на материал. Он «применился» к поверхности соковыжималки. Выберем команду **Закончить ввод**.

Повторим наложение материала для конуса, учитывая, что отверстия у конуса необходимо сделать желтым цветом, как на рисунке 47.

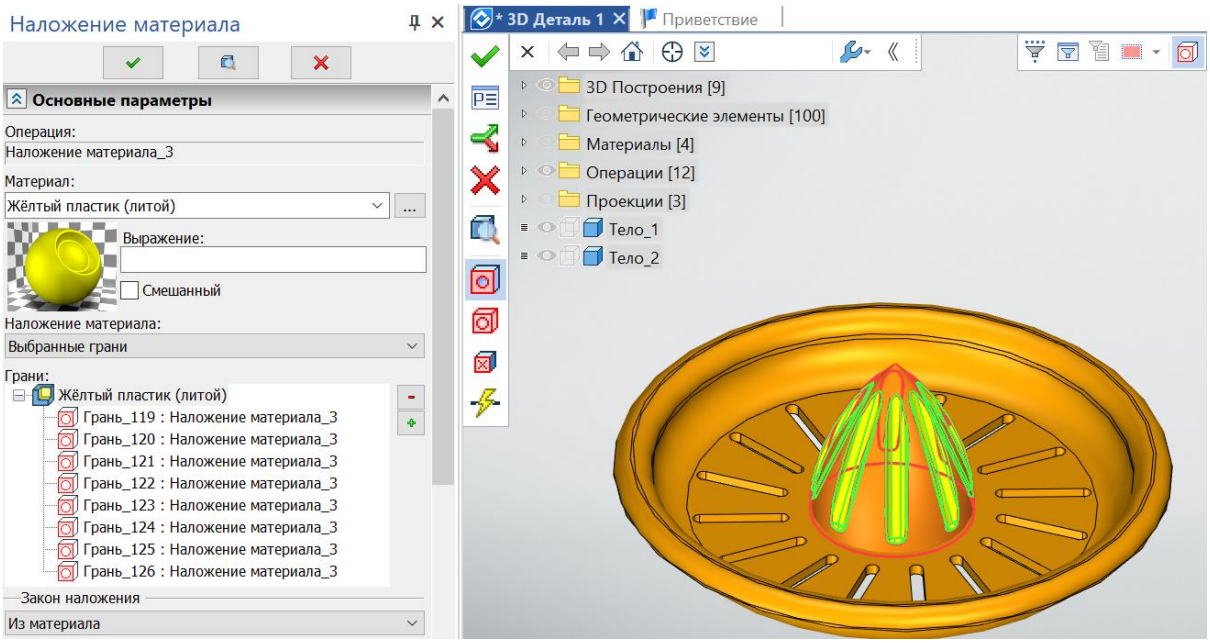


Рис. 47. Применение материала

3.7. А теперь «соберем» фотореалистичное изображение. Активируем команду **Фотореализм** из группы **Оформление**, находящейся на вкладке **Инструменты**.

3.8. T-FLEX постепенно строит изображение.

Качество и скорость создаваемого изображения во многом зависит от количества итераций. Итерация – вычисление цвета пикселей изображения. Количество итераций зависит от размера изображения, плотности сетки и количества объектов. Количество итераций отображается в нижней части экрана (рис. 48).

Рис. 48. Шкала итераций

Внимание! В зависимость от мощности компьютера, сложности модели и установленного качества изображения процесс генерирования изображения может занимать от нескольких минут до нескольких часов. Стоит учитывать, что при изменениях положения и/или угла поворота камеры построение изображения начнется заново.

3.9. По окончании построения мы получим финальное изображение (рис. 49).

Рис. 49. Финальное изображение

3.10. Теперь мы можем зафиксировать размер изображения, а также выбрать цвет фона.

Оптимальные параметры для создания фотореалистичного изображения установлены по умолчанию.

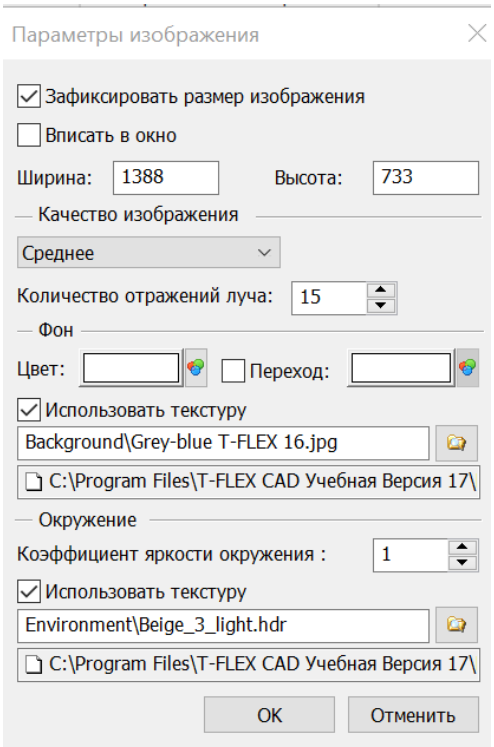


Рис. 50. Параметры рисунка

3.11. Нажмем кнопку **Сохранить изображение**. Выбираем папку, куда будет сохранено изображение, задаем имя файлу «Соковыжималка», выбираем тип файла и нажимаем «Сохранить».

Наше фотореалистичное изображение соковыжималки сохранено.

В конце практической работы выдается задание для самостоятельного выполнения, которые различаются по уровню сложности.

Задания для самостоятельного выполнения:

1. Добавьте боковые ручки к соковыжималке.

2. Примените другой материал к модели соковыжималки.

3. Создайте собственную емкость под соковыжималку для цитрусовых фруктов. В конце создайте фотореалистичное изображение своей модели.

После успешного выполнения заданий у каждого обучающегося появится собственный проект. При наличии в школе трехмерных принтеров можно напечатать прототип изделия.

Подготовка модели к печати на 3D-принтере

Напомним, что мы будем исходить из того, что уже произведены все настройки, как самого устройства, так и программного обеспечения, согласно инструкции, прилагаемой к 3D-принтеру.

Шаг 1. Сохранение 3D-модели в формате STL

1.1. Откроем созданную нами модель соковыжималки

1.2. На панели быстрого доступа найдем команду Экспорт и воспользуемся ей. Так же данную команду можно активировать сочетанием клавиш Ctrl и W.

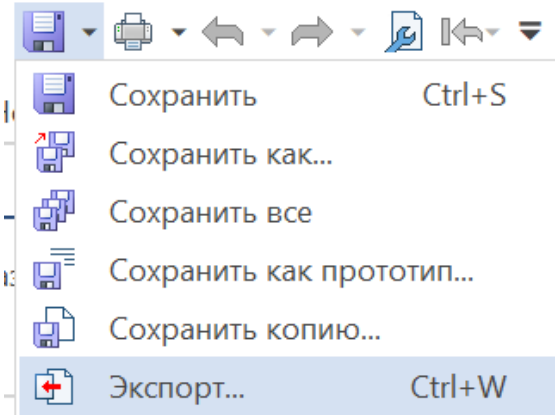


Рис. 51. Команда экспорт

После вызова команды, в системном окне отобразятся соответствующие параметры для экспорта форматов. Выбираем формат STL.

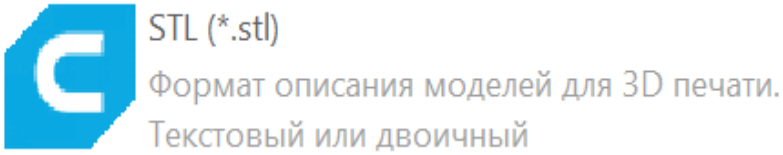


Рис. 52. Выбор формата экспортирования

1.3. Далее необходимо выбрать путь, где будет храниться файл, задаем имя файлу и нажимаем кнопку **Сохранить**.

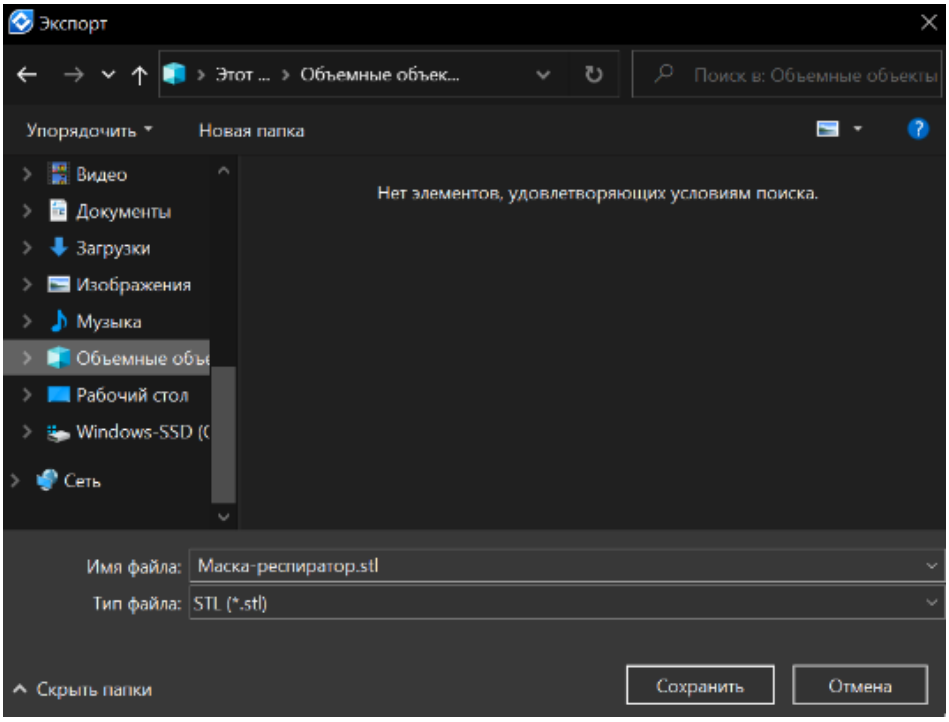


Рис. 53. Сохранение файла

Наша модель сохранена! Теперь подготовим ее к печати с помощью Ultimaker Cura.

Шаг 2. Подготовка задания на печать в Ultimaker Cura

2.1. Запустим Ultimaker Cura (рис. 54). Напомним еще раз, мы исходим из того, что принтер уже подготовлен к печати, подключен к компьютеру и в него заправлен пластик, а также загружена и настроена сама программа.

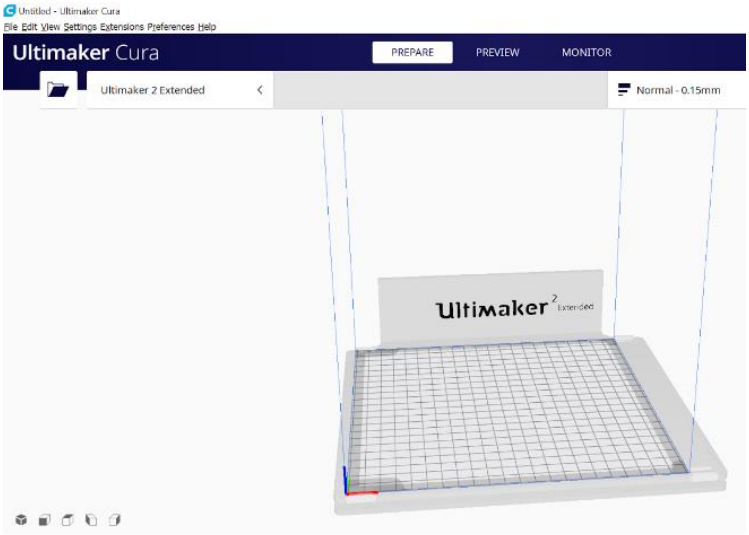


Рис. 54. Программа Ultimaker Cura

2.2. Для загрузки 3D-модели нажмем на пиктограмму папки  , расположенную в левом верхнем углу.

2.3. В открывшемся окне находим сохраненную модель и загружаем ее. В окне печати видим, что маска загружается на платформу, нижней гранью «прилипая» к платформе. Именно такое расположение 3D-модели позволит минимизировать печать поддержек.

2.4. Раскрываем вкладку **Настройка 3D-печати**, которая располагается в верхнем правом углу. Здесь задаются основные параметры печати.

Рис. 55. Вкладка «Настройка 3D-печати»

Настройки печати (рис. 55) зависят от материала, из которого будет печататься маска. В нашем случае мы будем подбирать параметры печати для пластика FLEX.

Изделия из FLEX-пластика можно стерилизовать, механически обрабатывать, а также не бояться их деформировать.

Важно! При 3D-печати пластиком FLEX следует соблюдать следующие меры безопасности: при печати следует использовать перчатки для защиты от термических ожогов, и респираторы; печатать в хорошо проветриваемом помещении или использовать 3D-принтер закрытого типа, с отводом газов из печатной камеры.

Стоит учитывать, что оптимальные настройки печати подбираются индивидуально, в зависимости от торговой марки материала и рекомендаций производителя 3D-принтера. В данной работе мы выставляем усредненные параметры печати.

2.5. Нам необходимо выставить нагрев платформы перед началом печати до температуры 110°C. Толщина слоя должна быть на 20% меньше диаметра сопла. Скорость печати минимальна 10–15 мм/с, так как в нагретом состоянии пластик очень вязок и при высокой скорости печати может забить

экструдер. Также необходимо выставить плотность заполнения. Сделаем ее равной 15%. Обычно это значение варьируется в пределах 5–20% в зависимости от требуемой прочности модели. Чем больше заполнение, тем более прочной будет модель, но тем больше время печати и количество используемого пластика.

Узор заполнения выберем Concentric (концентрический тип заполнения), такой тип обеспечивает достаточную прочность боковых стенок.

В процессе печати FLEX-пластиком требуется включать обдув до 15%. Так как у модели есть выступы, минимальный угол которых – 50°, то нам необходимо поставить галочку на включение поддержек при печати.

Настройка адгезии стола – необходимый пункт, так как предотвращает одну из распространенных проблем – отлипания пластика от стола для печати.

Существует три варианта для улучшения контакта – Skirt, Brim или Raft. Мы выбираем Skirt. Это настройка, которую лучше использовать всегда. Skirt помогает прочистить сопло во время начала 3D-печати.

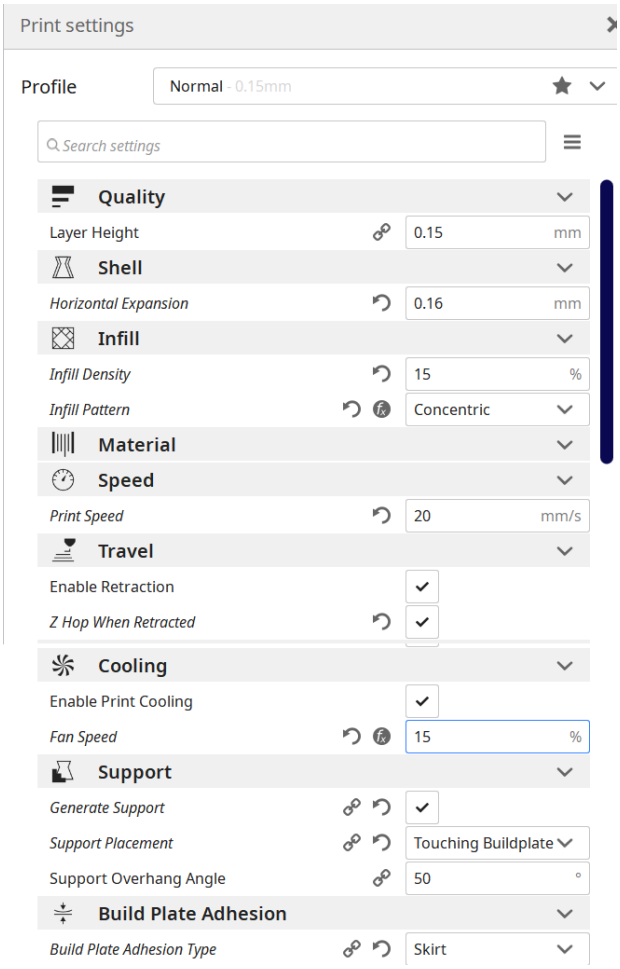


Рис. 56. Параметры 3D-печати

2.5. Далее нажимаем на кнопку **Slice** и ожидаем окончания процесса подготовки модели. После чего мы можем увидеть предварительные данные о статистике печати (рис. 57).

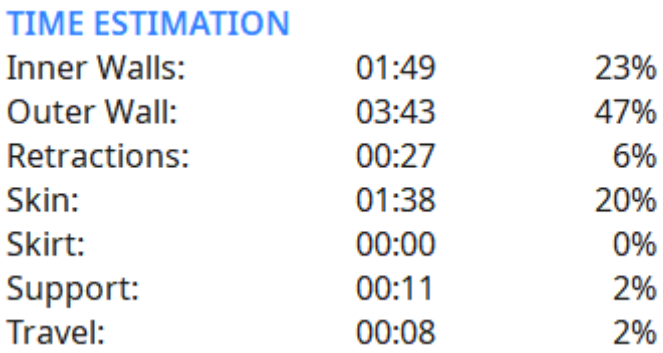


Рис. 57. Время печати

Если результат нас устраивает, то мы готовы к печати!

ЛИТЕРАТУРА

1. *Босова Л.Л.* Современные тенденции развития школьной информатики в России и за рубежом / Л.Л. Босова // Информатика и образование. – 2019. – № 1 (300). – С. 22–32.

2. *Калинин И.А.* Информатика. 10 класс: углубленный уровень: учебник / И.А. Калинин, Н.Н. Самылкина. – М.: Бином; Лаборатория знаний, 2020. – 256 с. – (ФГОС).

3. *Калинин И.А.* Информатика: углубленный уровень: учебник для 11 класса / И.А. Калинин, Н.Н. Самылкина.– Москва: Бином; Лаборатория знаний, 2020. – 211 с. – (ФГОС).

4. *Калинин И.А.* Информатика: углубленный уровень: задачник-практикум для 10–11 классов / И.А. Калинин, Н.Н. Самылкина, П.В. Бочаров. – М.: Бином; Лаборатория знаний, 2014. – 247 с. – (ФГОС). – URL: https://files.lbz.ru/authors/informatika/8/kal-zp10-11.pdf (дата обращения: 23.06.2023).

5. *Калинин И.А.* Искусственный интеллект: 10–11 классы: учеб. пособие / И.А. Калинин, Н.Н. Самылкина, А.А. Салахова. – М.: Просвещение, 2023. – 144 с.– (Профильная школа). – URL: https://shop.prosv.ru/iskusstvennyj-intellekt--10-11-klassy21811 (дата обращения: 23.06.2023).

6. Самылкина Н.Н. Информатика: 8–11 классы: практикум / Н.Н. Самылкина, И.А. Калинин, А.А. Салахова, В.В. Тарапата. – М.: Просвещение, 2023. – 157 с.

7. Паспорт национальной программы «Цифровая экономика Российской Федерации». [Электронный ресурс]. – URL: http://static.government.ru/media/files/urKHm0gTPPnzJlaKw3M5cNLo6gczMkPF.pdf (дата обращения: 23.06.2023).

8. *Поляков К.Ю.* Программирование. Python. C++. Ч. 1: учеб. пособие. – М.: Бином; Лаборатория знаний. – 2022. – URL: https://catalog.prosv.ru/item/48891 (дата обращения: 23.06.2023).

9. *Поляков К.Ю.* Программирование. Python. C++. Ч. 2: учеб. пособие. – М.: Бином; Лаборатория знаний. – 2022. – URL: https://catalog.prosv.ru/item/48892 (дата обращения: 23.06.2023).

10. *Поляков К.Ю.* Программирование. Python. C++. Ч. 3: учеб. пособие. – М.: Бином; Лаборатория знаний. – 2022. – URL: https://catalog.prosv.ru/item/48893 (дата обращения: 23.06.2023).

11. *Поляков К.Ю.* Программирование. Python. C++. Часть 4: учебное пособие. – М.: Бином. Лаборатория знаний. – 2022. – URL: https:// catalog.prosv.ru/item/48894 (дата обращения: 23.06.2023).

12. Федеральный закон от 29.12.2012 г. № 273-ФЗ «Об образовании в Российской Федерации» (последняя редакция). – URL: https://www. consultant.ru/document/cons_doc_LAW_140174/ (дата обращения: 23.06.2023).

13. *Чихольд Я.* Облик книги. Избранные статьи о книжном оформлении и типографике / Ян Чихольд; пер. с нем. Е. Шкловской-Корди. – 5-е изд. – М.: Изд-во Студии Артемия Лебедева, 2009. – 228 с.

14. *Щерба А.В.* Программирование на Python: первые шаги. – М.: Лаборатория знаний, 2022. – URL: https://pilotlz.ru/books/276/11128/ (дата обращения: 23.06.2023).

15. Приказ Министерства просвещения Российской Федерации от 12.08.2022 № 732 «О внесении изменений в федеральный государственный образовательный стандарт среднего общего образования, утвержденный приказом Министерства образования и науки Российской Федерации от 17 мая 2012 г. № 413» (Зарегистрирован Минюстом России 12.09.2022 № 70034).

16. Приказ Министерства просвещения Российской Федерации от 18.05.2023 № 371 «Об утверждении федеральной образовательной программы среднего общего образования» (Зарегистрирован Минюстом России 12.07.2023 № 74228).

Научное издание

Н.Н. Самылкина

**ИНФОРМАТИКА (УГЛУБЛЕННЫЙ УРОВЕНЬ).
РЕАЛИЗАЦИЯ ТРЕБОВАНИЙ ФГОС
СРЕДНЕГО ОБЩЕГО ОБРАЗОВАНИЯ**

Методическое пособие для учителя

101000, г. Москва, ул. Жуковского, д. 16
ФГБНУ «Институт стратегии развития образования»
Тел. +7(495)621–33–74
info@instrao.ru
https://instrao.ru

Подготовлено к изданию 08.08.2023.
Формат 60×90 1/8.
Усл. печ. л. 13,7.

ISBN 978-5-6049296-3-6