Государственное автономное профессиональное образовательное учреждение

Чувашской Республики

«Чебоксарский экономико-технологический колледж»

Министерства образования и молодежной политики Чувашской Республики

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**ДЛЯ ПРАКТИЧЕСКИХ ЗАНЯТИЙ**

**УПУУ.12 ИНФОРМАТИКА**

специальность

среднего профессионального образования

**43.02.15 Поварское и кондитерское дело**

Разработчик:

Петрова О.И., преподаватель

Чебоксары 2022

СОДЕРЖАНИЕ

[Введение 3](#_Toc480543228)

[Тематика практических занятий по учебной дисциплине 4](#_Toc480543229)

[Методические рекомендации к практическим занятиям 8](#_Toc480543230)

Литература 122

ВВЕДЕНИЕ

Рабочей программой дисциплины Информатика предусмотрены различные виды учебной работы: теоретическое обучение в объеме 8 часов и практические занятия в объеме 88 часов. Практические работы студентов – важнейшая составная часть занятий по информатике, необходимая для полного усвоения программы курса.

Целью практических занятий является изучение, закрепление и углубление знаний, полученных студентами на занятиях, подготовке к текущим занятиям, промежуточным формам контроля знаний.

Методические указания разработаны в соответствии с программой учебной дисциплины Информатика, на основе требований Федерального государственного образовательного стандарта (ФГОС) СПО..

Учебная дисциплина Информатика является профильной дисциплиной общеобразовательного цикла.

Методические указания имеют определенную структуру.

В первом разделе представлена тематика практических занятий и время, отведенное на их выполнение.

Во втором разделе представлены указания к практическим занятиям.

ТЕМАТИКА ПРАКТИЧЕСКИХ ЗАНЯТИЙ ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

|  |  |
| --- | --- |
| Наименование тем | Количество часов |
| **Тема 1.1.**  **Тексты и кодирование. Передача данных** |  |
| **Практическое занятие № 1. Тексты и кодирование. Передача данных**  Передача данных. Источник, приемник, канал связи, сигнал, кодирующее и декодирующее устройства.  *Пропускная способность и помехозащищенность канала связи. Кодирование сообщений в современных средствах передачи данных.* Искажение информации при передаче по каналам связи.Коды с возможностью обнаружения и исправления ошибок. *Способы защиты информации, передаваемой по каналам связи. Криптография (алгоритмы шифрования). Стеганография.* | 2 |
| **Тема 1.2.**  **Дискретизация** |  |
| **Практическое занятие № 2. Дискретизация**  Измерения и дискретизация. Частота и разрядность измерений. Универсальность дискретного представления информации. Дискретное представление звуковых данных. Многоканальная запись. Размер файла, полученного в результате записи звука. Дискретное представление статической и динамической графической информации.  *Сжатие данных при хранении графической и звуковой информации*. | 2 |
| **Тема 1.3. Системы счисления.** |  |
| **Практическое занятие № 3. Системы счисления.**  *Краткая и развернутая форма записи смешанных чисел в позиционных системах счисления. Перевод смешанного числа в позиционную систему счисления с заданным основанием.*  *Представление целых и вещественных чисел в памяти компьютера.* *Компьютерная арифметика.* | 2 |
| **Тема 1.4.**  **Элементы комбинаторики, теории множеств и математической логики** |  |
| **Практическое занятие № 4. Элементы комбинаторики, теории множеств и математической логики**.  Операции «импликация», «эквиваленция». Логические функции.  Законы алгебры логики. Эквивалентные преобразования логических выражений. Логические уравнения. Построение логического выражения с данной таблицей истинности.Дизъюнктивная нормальная форма. *Конъюнктивная нормальная форма.* Логические элементы компьютеров. Построение схем из базовых логических элементов. Дискретные игры двух игроков с полной информацией. Выигрышные стратегии. | 2 |
| **Тема 1.5.**  **Дискретные объекты** |  |
| **Практическое занятие № 5. Дискретные объекты.**  Решение алгоритмических задач, связанных с анализом графов (примеры: построения оптимального пути между вершинами ориентированного ациклического графа; определения количества различных путей между вершинами).  Обход узлов дерева в глубину. *Упорядоченные деревья (деревья, в которых упорядочены ребра, выходящие из одного узла).* Использование деревьев при решении алгоритмических задач (примеры: анализ работы рекурсивных алгоритмов, разбор арифметических и логических выражений). Бинарное дерево. *Использование деревьев при хранении данных.* Использование графов, деревьев, списков при описании объектов и процессов окружающего мира. | 2 |
| **Раздел 2. Алгоритмы и элементы программирования** |  |
| **Тема 2.1.**  **Алгоритмы и структуры данных** |  |
| **Практическое занятие № 6. Алгоритмы и структуры данных**  Алгоритмы исследования элементарных функций, в частности – точного и приближенного решения квадратного уравнения с целыми и вещественными коэффициентами, определения экстремумов квадратичной функции на отрезке. Алгоритмы анализа и преобразования записей чисел в позиционной системе счисления. Алгоритмы, связанные с делимостью целых чисел. Алгоритм Евклида для определения НОД двух натуральных чисел. | 2 |
| **Практическое занятие № 7. Алгоритмы и структуры данных**  Алгоритмы линейной (однопроходной) обработки последовательности чисел без использования дополнительной памяти, зависящей от длины последовательности (вычисление максимума, суммы; линейный поиск и т.п.). Обработка элементов последовательности, удовлетворяющих определенному условию (вычисление суммы заданных элементов, их максимума и т.п.). Алгоритмы обработки массивов. Примеры: перестановка элементов данного одномерного массива в обратном порядке; циклический сдвиг элементов массива; заполнение двумерного числового массива по заданным правилам; поиск элемента в двумерном массиве; вычисление максимума и суммы элементов двумерного массива. *Вставка и удаление элементов в массиве.* | 2 |
| **Практическое занятие № 8. Алгоритмы и структуры данных**  Рекурсивные алгоритмы, в частности: нахождение натуральной и целой степени заданного ненулевого вещественного числа; вычисление факториалов; вычисление n-го элемента рекуррентной последовательности (например, последовательности Фибоначчи). Построение и анализ дерева рекурсивных вызовов. Возможность записи рекурсивных алгоритмов без явного использования рекурсии. | 2 |
| **Практическое занятие № 9. Алгоритмы и структуры данных**  Сортировка одномерных массивов. Квадратичные алгоритмы сортировки (пример: сортировка пузырьком). Слияние двух отсортированных массивов в один без использования сортировки. Алгоритмы анализа отсортированных массивов. Рекурсивная реализация сортировки массива на основе слияния двух его отсортированных фрагментов.  Алгоритмы анализа символьных строк, в том числе: подсчет количества появлений символа в строке; разбиение строки на слова по пробельным символам; поиск подстроки внутри данной строки; замена найденной подстроки на другую строку. | 2 |
| **Практическое занятие № 10. Алгоритмы и структуры данных**  Построение графика функции, заданной формулой, программой или таблицей значений*.* Алгоритмы приближенного решения уравнений на данном отрезке, например, методом деления отрезка пополам. Алгоритмы приближенного вычисления длин и площадей, в том числе: приближенное вычисление длины плоской кривой путем аппроксимации ее ломаной; приближенный подсчет методом трапеций площади под графиком функции, заданной формулой, программой или таблицей значений. *Приближенное вычисление площади фигуры методом Монте-Карло. Построение траекторий, заданных разностными схемами. Решение задач оптимизации*. *Алгоритмы вычислительной геометрии. Вероятностные алгоритмы.* Сохранение и использование промежуточных результатов. Метод динамического программирования. Представление о структурах данных.Примеры: списки, словари, деревья, очереди. *Хэш-таблицы.* | 2 |
| **Тема 2.2.**  **Языки программирования** |  |
| **Практическое занятие № 11. Языки программирования**  Подпрограммы (процедуры, функции). Параметры подпрограмм. Рекурсивные процедуры и функции. Логические переменные. Символьные и строковые переменные. Операции над строками. Двумерные массивы (матрицы). *Многомерные массивы.* Средства работы с данными во внешней памяти. Файлы. | 2 |
| **Практическое занятие № 12. Языки программирования**  Подробное знакомство с одним из универсальных процедурных языков программирования. Запись алгоритмических конструкций и структур данных в выбранном языке программирования. Обзор процедурных языков программирования. *Представление о синтаксисе и семантике языка программирования.*  *Понятие о непроцедурных языках программирования и парадигмах программирования. Изучение второго языка программирования.* | 2 |
| **Тема 2.3.**  **Разработка программ** |  |
| **Практическое занятие № 13. Разработка программ**  Этапы решения задач на компьютере. Структурное программирование. Проверка условия выполнения цикла до начала выполнения тела цикла и после выполнения тела цикла: постусловие и предусловие цикла. Инвариант цикла.  Методы проектирования программ «сверху вниз» и «снизу вверх». Разработка программ, использующих подпрограммы. Библиотеки подпрограмм и их использование. | 2 |
| **Практическое занятие № 14. Разработка программ**  Интегрированная среда разработки программы на выбранном языке программирования. Пользовательский интерфейс интегрированной среды разработки программ. Понятие об объектно-ориентированном программировании. Объекты и классы. *Инкапсуляция, наследование, полиморфизм*. Среды быстрой разработки программ. Графическое проектирование интерфейса пользователя. Использование модулей (компонентов) при разработке программ. | 2 |
| **Практическое занятие № 15. Элементы теории алгоритмов**  Формализация понятия алгоритма. Машина Тьюринга – пример абстрактной универсальной вычислительной модели. Тезис Чёрча–Тьюринга. *Другие универсальные вычислительные модели* (*пример:* *машина Поста). Универсальный алгоритм. Вычислимые и невычислимые функции. Проблема остановки и ее неразрешимость.*  *Абстрактные универсальные порождающие модели (пример: грамматики).* | 2 |
| **Практическое занятие № 16. Элементы теории алгоритмов**  Сложность вычисления: количество выполненных операций, размер используемой памяти; их зависимость от размера исходных данных. Сложность алгоритма сортировки слиянием (MergeSort). Примеры задач анализа алгоритмов: определение входных данных, при которых алгоритм дает указанный результат; определение результата алгоритма без его полного пошагового выполнения. *Доказательство правильности программ.* | 2 |
| **Тема 2.5.**  **Математическое моделирование** |  |
| **Практическое занятие № 17. Математическое моделирование**  Практическая работа с компьютерной моделью по выбранной теме. Проведение вычислительного эксперимента. Анализ достоверности (правдоподобия) результатов компьютерного эксперимента. Представление результатов моделирования в виде, удобном для восприятия человеком. Графическое представление данных (схемы, таблицы, графики). Построение математических моделей для решения практических задач. Имитационное моделирование. *Моделирование систем массового обслуживания.* | 2 |
| **Практическое занятие № 18. Математическое моделирование**  *Использование дискретизации и численных методов в математическом моделировании непрерывных процессов.*  *Использование сред имитационного моделирования (виртуальных лабораторий) для проведения компьютерного эксперимента в учебной деятельности. Компьютерный (виртуальный) и материальный прототипы изделия. Использование учебных систем автоматизированного проектирования.* | 2 |
| **Раздел 3. Информационно-коммуникационные технологии и их использование для анализа данных** |  |
| **Тема 3.1.**  **Аппаратное и программное обеспечение компьютера** |  |
| **Практическое занятие № 19. Аппаратное и программное обеспечение компьютера**  Аппаратное обеспечение компьютеров. Персональный компьютер. Многопроцессорные системы. *Суперкомпьютеры*. *Распределенные вычислительные системы и обработка больших данных.* Мобильные цифровые устройства и их роль в коммуникациях. *Встроенные компьютеры. Микроконтроллеры. Роботизированные производства.*  Соответствие конфигурации компьютера решаемым задачам. Тенденции развития аппаратного обеспечения компьютеров. Программное обеспечение (ПО) компьютеров и компьютерных систем. Классификация программного обеспечения. Многообразие операционных систем, их функции. Программное обеспечение мобильных устройств. *Модель информационной системы «клиент–сервер». Распределенные модели построения информационных систем. Использование облачных технологий обработки данных в крупных информационных системах.* Инсталляция и деинсталляция программного обеспечения. *Системное администрирование.* Тенденции развития компьютеров. *Квантовые вычисления.* Техника безопасности и правила работы на компьютере. Гигиена, эргономика, ресурсосбережение, технологические требования при эксплуатации компьютерного рабочего места. *Проектирование автоматизированного рабочего места в соответствии с целями его использования.* *Применение специализированных программ для обеспечения стабильной работы средств ИКТ. Технология проведения профилактических работ над средствами ИКТ: диагностика неисправностей.* | 2 |
| **Тема 3.2. Подготовка текстов и демонстрационных материалов** |  |
| **Практическое занятие № 20. Подготовка текстов и демонстрационных материалов**  Технологии создания текстовых документов. | 2 |
| **Практическое занятие № 21. Подготовка текстов и демонстрационных материалов**  Технологии создания текстовых документов. | 2 |
| **Практическое занятие № 22. Подготовка текстов и демонстрационных материалов**  Вставка графических объектов, таблиц. Средства создания и редактирования математических текстов. | 2 |
| **Практическое занятие № 23. Подготовка текстов и демонстрационных материалов**  Использование готовых шаблонов и создание собственных. | 2 |
| **Практическое занятие № 24. Подготовка текстов и демонстрационных материалов**  Средства поиска и замены. Системы проверки орфографии и грамматики. Нумерация страниц. Разработка гипертекстового документа: определение структуры документа, автоматическое формирование списка иллюстраций, сносок и цитат, списка используемой литературы и таблиц. Библиографическое описание документов. Коллективная работа с документами. Рецензирование текста. | 2 |
| **Практическое занятие № 25. Подготовка текстов и демонстрационных материалов**  Технические средства ввода текста. Распознавание текста. *Распознавание устной речи.* *Компьютерная верстка текста. Настольно-издательские системы.* | 2 |
| **Тема 3.3.**  **Работа с аудиовизуальными данными** |  |
| **Практическое занятие № 26. Работа с аудиовизуальными данными**  Технические средства ввода графических изображений. Кадрирование изображений. Цветовые модели. Коррекция изображений. Работа с многослойными изображениями. Работа с векторными графическими объектами. Группировка и трансформация объектов. Технологии ввода и обработки звуковой и видеоинформации. *Технологии цифрового моделирования и проектирования новых изделий. Системы автоматизированного проектирования. Разработка простейших чертежей деталей и узлов с использованием примитивов системы автоматизированного проектирования. Аддитивные технологии (3D-печать).* | 2 |
| **Тема 3.4.**  **Электронные (динамические) таблицы** |  |
| **Практическое занятие № 27. Электронные (динамические) таблицы**  Технология обработки числовой информации. Ввод и редактирование данных. Автозаполнение. Форматирование ячеек. Стандартные функции. | 2 |
| **Практическое занятие № 28. Электронные (динамические) таблицы**  Виды ссылок в формулах. | 2 |
| **Практическое занятие № 29. Электронные (динамические) таблицы**  Фильтрация и сортировка данных в диапазоне или таблице. Коллективная работа с данными. *Подключение к внешним данным и их импорт.* | 2 |
| **Практическое занятие № 30. Электронные (динамические) таблицы**  Решение вычислительных задач из различных предметных областей. | 2 |
| **Практическое занятие № 31. Электронные (динамические) таблицы**  Компьютерные средства представления и анализа данных. Визуализация данных. | 2 |
| **Тема 3.5.**  **Базы данных** |  |
| **Практическое занятие № 32. Базы данных**  Понятие и назначение базы данных (далее – БД). Классификация БД. Системы управления БД (СУБД). | 2 |
| **Практическое занятие № 33. Базы данных**  Таблицы. Запись и поле. Ключевое поле. Типы данных. | 2 |
| **Практическое занятие № 34. Базы данных**  Запрос. Типы запросов. Запросы с параметрами. | 2 |
| **Практическое занятие № 35. Базы данных**  Сортировка. Фильтрация. Вычисляемые поля. | 2 |
| **Практическое занятие № 36. Базы данных**  *Формы. Отчеты.* | 2 |
| **Практическое занятие № 37. Базы данных**  Многотабличные БД. Связи между таблицами. *Нормализация*. | 2 |
| **Практическое занятие № 38. Базы данных**  Многотабличные БД. Связи между таблицами. *Нормализация*. | 2 |
| **Тема 3.6.**  **Подготовка и выполнение исследовательского проекта** |  |
| **Практическое занятие № 39. Подготовка и выполнение исследовательского проекта**  Технология выполнения исследовательского проекта: постановка задачи, выбор методов исследования, составление проекта и плана работ, подготовка исходных данных, проведение исследования, формулировка выводов, подготовка отчета. Верификация (проверка надежности и согласованности) исходных данных и валидация (проверка достоверности) результатов исследования. Статистическая обработка данных. Обработка результатов эксперимента. |  |
| **Тема 3.7.**  ***Системы искусственного интеллекта и машинное обучение*** |  |
| **Практическое занятие № 40. *Системы искусственного интеллекта и машинное обучение***  *Машинное обучение – решение задач распознавания, классификации и предсказания. Искусственный интеллект. Анализ данных с применением методов машинного обучения. Экспертные и рекомендательные системы.*  *Большие данные в природе и технике* *(геномные данные, результаты физических экспериментов, интернет-данные, в частности данные социальных сетей). Технологии их обработки и хранения.* | 2 |
| **Раздел 4. Работа в информационном пространстве** |  |
| **Тема 4.1. Компьютерные сети** |  |
| **Практическое занятие № 41. Компьютерные сети**  Принципы построения компьютерных сетей. *Аппаратные компоненты компьютерных сетей. Проводные и беспроводные телекоммуникационные каналы.* Сетевые протоколы. Принципы межсетевого взаимодействия. Сетевые операционные системы. *Задачи системного администрирования компьютеров и компьютерных сетей.*  Интернет. Адресация в сети Интернет (IP-адреса, маски подсети). Система доменных имен.  Технология WWW. Браузеры. Веб-сайт. Страница. Взаимодействие веб-страницы с сервером. Язык HTML. Динамические страницы. Разработка веб-сайтов. Язык HTML, каскадные таблицы стилей (CSS). *Динамический HTML. Размещение веб-сайтов. Использование сценариев на языке Javascript. Формы. Понятие о серверных языках программирования.* Сетевое хранение данных. Облачные сервисы. | 2 |
| **Тема 4.2.**  **Деятельность в сети Интернет** |  |
| **Практическое занятие № 42**. **Деятельность в сети Интернет**  Расширенный поиск информации в сети Интернет. Использование языков построения запросов.  Другие виды деятельности в сети Интернет. Сервисы Интернета. Геолокационные сервисы реального времени (локация мобильных телефонов, определение загруженности автомагистралей и т.п.); интернет-торговля; бронирование билетов и гостиниц и т.п. Облачные версии прикладных программных систем. Новые возможности и перспективы развития Интернета: мобильность, облачные технологии, виртуализация, социальные сервисы, доступность. *Технологии «Интернета вещей». Развитие технологий распределенных вычислений.* | 2 |
| **Тема 4.3. Социальная информатика** |  |
| **Практическое занятие № 43**. **Социальная информатика**  Социальные сети – организация коллективного взаимодействия и обмена данными. Проблема подлинности полученной информации. *Государственные электронные сервисы и услуги.* Мобильные приложения. Открытые образовательные ресурсы. Информационная культура. Информационные пространства коллективного взаимодействия. Сетевой этикет: правила поведения в киберпространстве. *Стандартизация и стандарты в сфере информатики и ИКТ докомпьютерной эры (запись чисел, алфавитов национальных языков, библиотечного и издательского дела и др.) и компьютерной эры (языки программирования).* | 2 |
| **Тема 4.4. Информационная безопасность** |  |
| **Практическое занятие № 44**. **Информационная безопасность**  Средства защиты информации в автоматизированных информационных системах (АИС), компьютерных сетях и компьютерах. Общие проблемы защиты информации и информационной безопасности АИС. Компьютерные вирусы и вредоносные программы. Использование антивирусных средств. Электронная подпись, сертифицированные сайты и документы. Правовые нормы использования компьютерных программ и работы в Интернете. Законодательство РФ в области программного обеспечения. Техногенные и экономические угрозы, связанные с использованием ИКТ. Правовое обеспечение информационной безопасности. | 2 |
| Всего | 88 |

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ К ПРАКТИЧЕСКИМ ЗАНЯТИЯМ

**Практическое занятие № 1. Тексты и кодирование. Передача данных**

Передача данных. Источник, приемник, канал связи, сигнал, кодирующее и декодирующее устройства.

*Пропускная способность и помехозащищенность канала связи. Кодирование сообщений в современных средствах передачи данных.* Искажение информации при передаче по каналам связи.Коды с возможностью обнаружения и исправления ошибок. *Способы защиты информации, передаваемой по каналам связи. Криптография (алгоритмы шифрования). Стеганография.*

**Цель работы:** Получить навыки кодирования и декодирования информации

**Требования к оформлению работы:**

Название практического занятия;

Цель;

Самостоятельная работа;

Ответы на контрольные вопросы;

Вывод о проделанной работе.

**Ход работы:**

1. Изучить теорию по теме [«Теория кодирования»](http://museum-pat.narod.ru/date/teo_cd.html)
2. [Примеры выполнения заданий](http://museum-pat.narod.ru/date/primer_cd.html)
3. [Выполнить самостоятельную работу](http://museum-pat.narod.ru/date/sam_cd.html)
4. Ответить на контрольные вопросы:
5. Что изучает теория кодирования?
6. Объясните термины: кодирование и декодирование.
7. Что такое код?
8. В чем заключается алфавитное кодирование?
9. Как кодируются тексты?
10. Как кодируются изображения?
11. Как кодируются звук и видео?

**Теория кодирования** – это раздел теории информации, изучающий способы отождествления сообщений с отражающими их сигналами. Задачей теории кодирования является согласование источника информации с каналом связи.

Объектом кодирования служит как дискретная, так и непрерывная информация, которая поступает к потребителю через источник информации. Понятие кодирование означает преобразование информации в форму, удобную для передачи по определенному каналу связи.

Обратная операция – **декодирование** – заключается в восстановлении принятого сообщения из закодированного вида в общепринятый, доступный для потребителя.

В теории кодирования существует ряд направлений:

* статическое или эффективное кодирование;
* помехоустойчивое кодирование;
* корректирующие коды;
* циклические коды;
* арифметические коды.

С появлением управляющих систем, в частности ЭВМ, роль кодирования существенно возросла и изменилась, так как без кодирования невозможна передача информации. В последнее время в связи с развитием телекоммуникационных систем и широким использованием вычислительной техники для обработки и хранения информации возникла новая область знаний – информационная безопасность.

**Кодированием** называют универсальный способ отображения информации при ее хранении, обработке и передаче в виде системы соответствий между сигналами и элементами сообщений, при помощи которых эти элементы можно зафиксировать.

**Код**– это правило однозначного преобразования сообщения из одной символической формы представления сообщения в другую, обычно без каких-либо потерь информации.

Если все кодовые слова имеют одинаковую длину, то код называется равномерным, или блочным.

Под абстрактным алфавитом будем понимать упорядоченное дискретное множество символов.

**Алфавитное кодирование.** Алфавитное, т.е. побуквенное, кодирование можно задать таблицей кодов. Фактически кодом преобразования является некоторая подстановка.

Тогда http://museum-pat.narod.ru/date/design/formula/f13.gif, где http://museum-pat.narod.ru/date/design/formula/f14.gif алфавиту А, http://museum-pat.narod.ru/date/design/formula/f15.gif множеству слов, составленных в алфавите В. Множество кодов букв называется множеством элементарных кодов. Алфавитное кодирование можно использовать для любого множества сообщений.

Компьютерная обработка данных основана на применении двоичного кода. Этот универсальный способ кодирования годится для любых данных, независимо от их происхождения и содержания.

**Кодирование текста**

**Тексты** – это последовательности символов, входящих некоторый алфавит. Кодирование текста сводится к двоичному кодированию алфавита, на основе которого он построен. Чаще всего применяется байтовое кодирование алфавита. В этом случае максимальная мощность алфавита составляет 256 символов. Такой алфавит может содержать два набора буквенных символов (например, русский и латинский), цифры, знаки препинания и математические знаки, пробел и небольшое число дополнительных символов. Примером такого алфавита является код **ASCII**.

Однако, ограниченный набор из 256 кодов символов сегодня уже не удовлетворяет возросшие потребности международного общения. Все большее распространение получает универсальная система 16-разрядного кодирования символов **UNICODE**.

Мощность алфавита в системе кодирования UNICODE составляет 216=65 536 разных кодов, из которых 63 484 кода соответствуют символам большинства алфавитов, а оставшиеся 2048 кодов разделены пополам и образуют таблицу размером 1024 столбцов х 1024 строк. В этой таблице более миллиона ячеек, в которых можно разместить еще более миллиона различных символов. Это символы «мертвых» языков, а также символы, не имеющие лексического содержания, указатели, знаки и т.п. Для записи этих дополнительных символов необходима пара 16-разрядных слов (16 разрядов для номера строки и 16 разрядов для номера столбца).

Таким образом, система UNICODE является универсальной системой кодирования всех символов национальных письменных систем и обладает возможностью существенного расширения.

**Кодирование изображений**

Рисунки, картинки, фотографии кодируются **в растровом формате**. В этом виде каждое изображение представляет собой прямоугольную таблицу, состоящую из цветовых точек. Цвет и яркость каждой отдельной точки выражаются в числовой форме, что позволяет использовать двоичный код для представления графических данных.

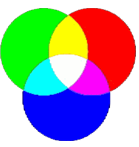
Черно-белые изображения принято представлять в градациях серого цвета, для этого используется модель **GreyScale**. Если яркость точки кодируется одним байтом, можно использовать 256 различных серых тонов. Такая точность согласуется с восприимчивостью человеческого глаза и возможностями полиграфической техники.

При кодировании цветных изображений применяют принцип декомпозиции цвета на составляющие, для этого используют модель **RGB**. Цветное изображение на экране получается путем смешивания трех базовых цветов: красного (Red, R), синего (Blue, B) и зеленого (Green, G).

Каждый пиксель на экране состоит из трех близко расположенных элементов, светящихся этими цветами.

Цветные дисплеи, использующие такой принцип называются RGB -мониторами.

Код цвета пикселя содержит информацию о доле каждого базового цвета.



**Схема цветообразования**

Если все три составляющих имеют одинаковую интенсивность (яркость), то из их сочетаний можно получить 8 различных цветов:

|  |  |  |  |
| --- | --- | --- | --- |
| **R** | **G** | **B** | **цвет** |
| 0 | 0 | 0 | http://museum-pat.narod.ru/date/design/formula/black.gif черный |
| 0 | 0 | 1 | http://museum-pat.narod.ru/date/design/formula/blue.gif синий |
| 0 | 1 | 0 | http://museum-pat.narod.ru/date/design/formula/green.gif зеленый |
| 0 | 1 | 1 | http://museum-pat.narod.ru/date/design/formula/cyan.gif голубой |
| 1 | 0 | 0 | http://museum-pat.narod.ru/date/design/formula/red.gif красный |
| 1 | 0 | 1 | http://museum-pat.narod.ru/date/design/formula/pink.gif розовый |
| 1 | 1 | 0 | http://museum-pat.narod.ru/date/design/formula/brown.gif коричневый |
| 1 | 1 | 1 | http://museum-pat.narod.ru/date/design/formula/white.gif белый |

Формирование цветов при глубине цвета 24 бита:

|  |  |  |  |
| --- | --- | --- | --- |
| цвет | интенсивность R | интенсивность G | интенсивность B |
| черный | 00000000 | 00000000 | 00000000 |
| красный | 11111111 | 00000000 | 00000000 |
| зеленый | 00000000 | 11111111 | 00000000 |
| синий | 00000000 | 00000000 | 11111111 |
| голубой | 00000000 | 11111111 | 11111111 |
| желтый | 11111111 | 11111111 | 00000000 |
| белый | 11111111 | 11111111 | 11111111 |

Чем больше глубина цвета, тем шире диапазон доступных цветов и тем точнее их представление в оцифрованном изображении. Пиксель с битовой глубиной, равной единице, имеет лишь 2 (в первой степени) возможных состояния — два цвета: черный или белый. Пиксель с битовой глубиной в 8 единиц имеет 28 или 256 возможных цветовых значений. Пиксель же с битовой глубиной в 24 единицы имеет 224 степени) или 16,7 миллионов возможных значений. Считается, что 24-битные изображения, содержащие 16,7 миллионов цветов, достаточно точно передают краски окружающего нас мира. Как правило, битовое разрешение задается в диапазоне от 1 до 48 бит/пиксель.

При печати на бумаге используется несколько иная цветовая модель: если монитор испускал свет, оттенок получался в результате сложения цветов, то краски - поглощают свет, цвета вычитаются. Поэтому в качестве основных используют голубую (Cyan, C), пурпурную (Magenta, M) и желтую (Yellow, Y) краски. Кроме того, из-за не идеальности красителей, к ним обычно добавляют четвертую -- черную (black, K). Для хранения информации о каждой краске и в этом случае чаще всего используется 1 байт. Такая система кодирования носит название **CMYK**.

Более грубое представление цвета использует меньшее число разрядов. Например, кодирование цветной графики 16-разрядными числами носит название **High Color**. В этом случае каждому цвету отводят пять разрядов.

**Кодирование звука и видео**

Приемы работы со звуковой информацией пришли в компьютерную технику позже всего. Аналитический метод кодирования, применимый к любым звуковым сигналам основан на аналогово-цифровом преобразовании. Исходный аналоговый сигнал представляют как последовательность цифровых сигналов, записанных в двоичном коде. Разрядность преобразования определяет объем данных, соответствующих отдельному цифровому сигналу. При воспроизведении звука выполняют обратное цифро-аналоговое преобразование.

Этот метод кодирования содержит погрешность, так что воспроизводимый сигнал несколько отличается от оригинала.

Метод кодирования на основе табличного синтеза применим только к музыкальным произведениям. В заранее подготовленных таблицах хранятся образцы (**сэмплы**) звуков различных музыкальных инструментов. Числовые коды определяют инструмент, ноту и продолжительность звучания.

При кодировании видеосигнала требуется записать последовательность изображений (кадров) и звук (звуковая дорожка). Формат видеозаписи позволяет включить оба потока данных в одну цифровую последовательность.

http://museum-pat.narod.ru/date/design/title/primer.png

1. Заменяя каждую букву ее порядковым номером в алфавите, зашифруйте фразу: "Я УМЕЮ КОДИРОВАТЬ ИНФОРМАЦИЮ".   
  
Решение: Для того, чтобы легко закодировать фразу, заменяя каждую букву ее порядковым номером в алфавите необходимо составить таблицу соответствия "Буква - её порядковый номер":

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **№** | **Буква** | **№** | **Буква** | **№** | **Буква** | **№** | **Буква** | **№** | **Буква** |
| 1 | А | 8 | Ж | 15 | Н | 22 | Ф | 29 | Ы |
| 2 | Б | 9 | З | 16 | О | 23 | Х | 30 | Ь |
| 3 | В | 10 | И | 17 | П | 24 | Ц | 31 | Э |
| 4 | Г | 11 | Й | 18 | Р | 25 | Ч | 32 | Ю |
| 5 | Д | 12 | К | 19 | С | 26 | Ш | 33 | Я |
| 6 | Е | 13 | Л | 20 | Т | 27 | Щ | - | расстояние между буквами |
| 7 | Ё | 14 | М | 21 | У | 28 | Ъ | - | пробел |

Теперь можно легко зашифровать фразу. Она будет выглядеть так: "33 21-14-6-32 12-16-5-10-18-16-3-1-20-30 10-15-22-16-18-14-1-24-10-32"

2. Дана кодировочная таблица (первая цифра кода - номер строки, вторая - номер столбца):

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| **0** | А | Б | В | Г | Д | Е | Ж | З | И |
| **1** | Й | К | Л | М | Н | О | П | Р | С |
| **2** | Т | У | Ф | Х | Ц | Ч | Ш | Щ | Ъ |
| **3** | Ы | Ь | Й | Э | Ю | Я | \_ | . | , |
| **4** | : | ; | - | ! | " |  |  |  |  |

С помощью этой кодировочной таблицы зашифруйте фразу: ТЕЛЕГРАММА. Решение: 20 05 12 05 03 17 00 13 13 00

Вариантhttp://museum-pat.narod.ru/date/design/var/num/1.png

1 Дана кодовая таблица азбуки Морзе

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| А | • – | Л | • – • • | Ц | – • – • |
| Б | – • • • | М | – – | Ч | – – – • |
| В | • – – | Н | – • | Ш | – – – – |
| Г | – – • | О | – – – | Щ | – – • – |
| Д | – • • | П | • – – • | Ъ | • – – • – • |
| Е | • | Р | • – • | Ы | – • – – |
| Ж | • • • – | С | • • • | Ь | – • • – |
| З | – – • • | Т | – | Э | • • – • • |
| И | • • | У | • • – | Ю | • • – – |
| Й | • – – – | Ф | • • – • | Я | • – • – |
| К | – • – | Х | • • • • |  |  |

Расшифруйте (декодируйте), что здесь написано (буквы отделены друг от друга пробелами): – – – – – • – • • – – – – • • – • – • – • – –

2 Закодируйте с помощью азбуки Морзе слова: информатика, данные, алгоритм.

3 Закодируйте с помощью азбуки Морзе свое имя и фамилию.   
 Дана кодировочная таблица (первая цифра кода - номер строки, вторая - номер столбца)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| **0** | А | Б | В | Г | Д | Е | Ж | З | И |
| **1** | Й | К | Л | М | Н | О | П | Р | С |
| **2** | Т | У | Ф | Х | Ц | Ч | Ш | Ш | Ъ |
| **3** | Ы | Ь | Э | Ю | Я | \_ | . | , | ? |
| **4** | : | ; | - | ! | " |  |  |  |  |

С помощью этой кодировочной таблицы зашифруйте фразу: Я УМЕЮ РАБОТАТЬ С ИНФОРМАЦИЕЙ! А ТЫ?

 Заменяя каждую букву ее порядковым номером в алфавите, зашифруйте фразу: "Я УМЕЮ КОДИРОВАТЬ ИНФОРМАЦИЮ". Что необходимо предусмотреть, чтобы зашифрованный текст был записан без пропусков?

Вариантhttp://museum-pat.narod.ru/date/design/var/num/2.png

1 Дана кодовая таблица азбуки Морзе

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| А | • – | Л | • – • • | Ц | – • – • |
| Б | – • • • | М | – – | Ч | – – – • |
| В | • – – | Н | – • | Ш | – – – – |
| Г | – – • | О | – – – | Щ | – – • – |
| Д | – • • | П | • – – • | Ъ | • – – • – • |
| Е | • | Р | • – • | Ы | – • – – |
| Ж | • • • – | С | • • • | Ь | – • • – |
| З | – – • • | Т | – | Э | • • – • • |
| И | • • | У | • • – | Ю | • • – – |
| Й | • – – – | Ф | • • – • | Я | • – • – |
| К | – • – | Х | • • • • |  |  |

Расшифруйте (декодируйте), что здесь написано (буквы отделены друг от друга пробелами): - • - • - • • • - - - - -  
2 Закодируйте с помощью азбуки Морзе слова: информатика, данные, алгоритм.   
3 Закодируйте с помощью азбуки Морзе свое имя и фамилию.   
 Дана кодировочная таблица (первая цифра кода - номер строки, вторая - номер столбца)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| **0** | А | Б | В | Г | Д | Е | Ж | З | И |
| **1** | Й | К | Л | М | Н | О | П | Р | С |
| **2** | Т | У | Ф | Х | Ц | Ч | Ш | Ш | Ъ |
| **3** | Ы | Ь | Э | Ю | Я | \_ | . | , | ? |
| **4** | : | ; | - | ! | " |  |  |  |  |

С помощью этой кодировочной таблицы зашифруйте фразу: "Таблицы маршрутизации - электронные базы данных".   
 Заменяя каждую букву ее порядковым номером в алфавите, зашифруйте фразу: "Скрипт - набор команд какого-либо языка". Что необходимо предусмотреть, чтобы зашифрованный текст был записан без пропусков?

Вариантhttp://museum-pat.narod.ru/date/design/var/num/3.png

1Дана кодовая таблица азбуки Морзе

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| А | • – | Л | • – • • | Ц | – • – • |
| Б | – • • • | М | – – | Ч | – – – • |
| В | • – – | Н | – • | Ш | – – – – |
| Г | – – • | О | – – – | Щ | – – • – |
| Д | – • • | П | • – – • | Ъ | • – – • – • |
| Е | • | Р | • – • | Ы | – • – – |
| Ж | • • • – | С | • • • | Ь | – • • – |
| З | – – • • | Т | – | Э | • • – • • |
| И | • • | У | • • – | Ю | • • – – |
| Й | • – – – | Ф | • • – • | Я | • – • – |
| К | – • – | Х | • • • • |  |  |

Расшифруйте (декодируйте), что здесь написано (буквы отделены друг от друга пробелами): • - - • • - • - - - • • -  
2 Закодируйте с помощью азбуки Морзе слова: флаг, имя, кодирование.  
3 Закодируйте с помощью азбуки Морзе свое имя и фамилию.   
 Дана кодировочная таблица (первая цифра кода - номер строки, вторая - номер столбца)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| **0** | А | Б | В | Г | Д | Е | Ж | З | И |
| **1** | Й | К | Л | М | Н | О | П | Р | С |
| **2** | Т | У | Ф | Х | Ц | Ч | Ш | Ш | Ъ |
| **3** | Ы | Ь | Э | Ю | Я | \_ | . | , | ? |
| **4** | : | ; | - | ! | " |  |  |  |  |

С помощью этой кодировочной таблицы зашифруйте фразу: "УЧЕНИЕ ДА ТРУД К СЛАВЕ ВЕДУТ"  
 Заменяя каждую букву ее порядковым номером в алфавите, зашифруйте фразу: "ЮЗЕР - ПОЛЬЗОВАТЕЛЬ КОМПЬЮТЕРА". Что необходимо предусмотреть, чтобы зашифрованный текст был записан без пропусков?

**Практическое занятие № 2. Дискретизация**

Измерения и дискретизация. Частота и разрядность измерений. Универсальность дискретного представления информации. Дискретное представление звуковых данных. Многоканальная запись. Размер файла, полученного в результате записи звука. Дискретное представление статической и динамической графической информации.

*Сжатие данных при хранении графической и звуковой информации*.

Цель:

1. Закрепление понятий информация, информационные процессы.
2. Рассмотрение основных свойств информации.
3. Ознакомление с кодированием информации в В.Т.
4. Решение задач на кодирование информации, на количество информации.

Студент должен знать:

1. Понятие информации, свойства информации, информационные процессы.
2. Способы кодирования информации.
3. Единицы измерения информации.
4. Виды информации.

Студент должен уметь:

1. Кодировать информацию.
2. Решать задачи на нахождение количества информации.

### Теория:

1 Гб = 1024 Мб

1 Мб = 1024 Кб

1 Кб = 1024 байта

1 байт = 8 бит

Практические задания:

1. Переведите 2 Мб в Кб, байты, биты.
2. Переведите 27262976 бит в Мб, Кб, байты

### Теория:

По умолчанию (если в задаче не указано специально) при решении задачи указывается 256-символьный алфавит – таблица ASCII (мощность алфавита = 256 символов). Значит, на один символ (букву, цифру, знак, знак препинания, пробел) приходится 8 бит информации или 1 байт.

Практические задания:

1. Определить количество информации, которое содержится на печатном листе бумаги (двусторонняя печать), если на одной стороне умещается 40 строк по 67 символов в строке.

### Теория:

,

где i – количество информации в бит, N – количество символов, количество событий

Символом является и знак препинания, и цифра, и специальные знаки. И пробел!

1. Какое количество информации будет содержаться на странице печатного текста при использовании 32-х символьного алфавита (на странице 60 строк по 56 символов).

Практические задания:

1. Какое количество символов содержится на странице энциклопедического словаря, если в памяти компьютера эта страница занимает 13 Кб?
2. Используя данные предыдущей задачи и зная, что в одной строке находится 85 знаков, определить количество строк на странице.
3. Сколько бит содержится в слове ИНФОРМАЦИЯ?
4. В каком алфавите одна буква несет в себе больше информации, в русском или латинском? Примечание: в русском языке 33 буквы, в латинском – 26 букв.
5. Какое количество вопросов необходимо задать, чтобы наверняка угадать загаданного обучающегося из вашей группы?
6. Сколько символов содержится в алфавите, при помощи которого написана книга из 20 страниц, на каждой из которых содержится 15 строк по 20 символов и занимает в памяти компьютера вся книга 5,86 Кб.
7. В алфавите некоторого формального языка всего два знака буквы. Каждое слово этого языка состоит обязательно из 7 букв. Какое максимальное число слов возможно записать в этом языке?
8. Сколько цветов содержит рисунок размером 100\*150 пиксел и объемом 29,3 Кб

**Практическое занятие № 3. Системы счисления.**

Краткая и развернутая форма записи смешанных чисел в позиционных системах счисления. Перевод смешанного числа в позиционную систему счисления с заданным основанием.

Представление целых и вещественных чисел в памяти компьютера. Компьютерная арифметика.

Цель:

1. Ознакомится с приемами перевода из одной системы счисления в другую.
2. Овладение приемами работы в двоичной арифметике.

Студент должен знать:

1. Классификацию систем счисления.
2. Алгоритмы перевода чисел из одной системы в другую.

Студент должен уметь:

1. Уметь пользоваться алгоритмами по переводу чисел из одной системы счисления в другую.
2. Уметь производить арифметические действия в двоичной системе счисления.

Примеры выполнения заданий

1. Переведите целые числа из предложенной системы счисления в десятичную

Решение:





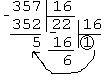
 - неправильная запись числа



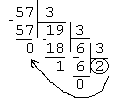


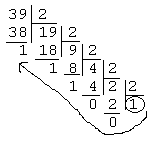
1. Переведите целые числа из десятичной системы счисления в предложенную

Решение:



перевести в 3-сс = 20103

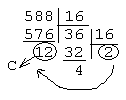




перевести в 16-сс = 16516

перевести в 2-сс = 1001112

перевести в 16-сс = 24С16

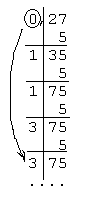


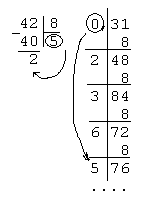
1. Переведите дробные числа из предложенной системы счисления в десятичную

Решение:



1. Переведите дробные числа из десятичной системы счисления в предложенную





Решение:

перевести в 5-сс = 0,11(3)5

перевести в 8-сс = 52,23658

### Теория:

Таблица сложения в двоичной системе счисления:

0 + 0 = 0

1 + 0 = 1

0 + 1 = 1

1 + 1 = 10

Т.к. в двоичной системе счисления в записи чисел используются только 2 цифры – 0 и 1, значит при сложении 1 + 1 в младшем разряде записывается 0, а 1 переходит в старший разряд.

По аналогии с 10-СС: 9 + 1 (цифры десять нет в записи чисел), записывается 0 и 1 в старшем разряде, получается 10.

1. Выполните арифметические операции сложения для следующих двоичных чисел:

101101 101 101

+ 11011 + 01 + 11

1001000 110 1000

1. Выполните арифметические операции вычитания для следующих двоичных чисел:

1011 1001

- 10 - 10

1001 111

Алгоритм процедуры вычитания в двоичной системе:

11101001 - 1001101

(обязательно числа нужно представить целым количеством байт)

11101001 - 01001101

1. Вычитаемое сначала инвертируется: 01001101 превращается в 10110010

2. К нему прибавляется единица: 10110010 + 1 = 10110011 – получается обратный код

3. Выполняется операция сложения 11101001 + 10110011 = 1'10011100

единица из восьмого разряда отбрасывается и остается ответ:

10011100

Теория:

Операция умножения сводится к поочередному сдвигу чисел и нахождению их суммы.

1. Выполните арифметические операции умножения для следующих двоичных чисел:

101 101

\* 11 \* 101

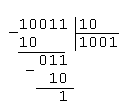
101 101

+ 101 + 000

1111 101

11001

1. Выполните арифметические операции деления для следующих двоичных чисел:



1. По виду числа определить СС:
   1. 241 – может быть записано в СС, начиная с 5-ричной, т.к. в ней используются цифры 0 1 2 3 4
   2. 73 - может быть записано в СС, начиная с 8-ричной, т.к. в ней используются цифры 0 1 2 3 4 5 6 7
2. Какое число ошибочно записано в:
   1. Троичной СС – 79, 212, 531

Решение: в троичной СС для записи чисел используются цифры 0 1 2 , значит цифры 79 и 531 записаны неверно

* 1. Девятиричной СС – 419, 832, 4А

Решение: в девятиричной СС для записи чисел используются цифры 0 1 2 3 4 5 6 7 8, значит цифры 419 и 4А записаны неверно

1. Записать число в виде многочлена:
   1. 143,710
   2. 246,58

Решение:

143,710 = 1\*102+4\*101+3\*100+7\*10-1

246,58 = 2\*82+4\*81+6\*80+5\*8-1

1. Когда 5 \* 3 = 21 ?

Решение:

Перефразируем задание: в какой СС число 15 (ведь 5 \* 3 = 15), записывается как 21?

Попробуем переводить число 15 в различные СС, можно сразу определить в какую.

Задаем вопрос: на что надо разделить число 15, чтобы частное было равно 2.



Значит, в 7-ричной СС число 15 записывается как 21.

1. У меня 100 братьев. Младшему 1000 лет, а старшему 1111 лет. Старший учится в 1001 классе. Может ли такое быть?

Решение:

Такое может быть, если попробовать перевести данные в какую-либо СС. Начнем с двоичной. Представим, что все эти числа записаны в двоичной СС. Переведем их в десятичную.

100 – это число 4

1000 – это число 8

1111 - это число 15

1001 – это число 9

У меня 4 брата. Младшему 8 лет, а старшему 15 лет. Старший учится в 9 классе.

Такое может быть.

1. Расставить знаки арифметических операций так, чтобы были верны следующие равенства в двоичной СС:

1100 ? 11 ? 100 = 100000

Решение:

Перепишем неравенство в десятичной СС и расставим знаки:

12 \* 3 – 4 = 32

Практические задания:

Часть – А

№1. Выполните перевод чисел в соответствующую систему счисления:

3510→?2

10210→?8

5510→?16

10102→?10

248→?10

3316→?10

№2. Выполните расчеты

111012+10102

110012+101102

11001012+110102

111012-1012

1011012-10102

1101012-101012

Часть – В

№1. Выполните перевод чисел в соответствующую систему счисления:

|  |  |  |  |
| --- | --- | --- | --- |
| 2-я СС | 8-я СС | 10-я СС | 16-я СС |
|  | 23 |  |  |
|  |  | 15,5 |  |

№2. Выполните расчеты

111112+10102

110112+1001102

11001012+100112

111012-1112

1011112-10102

1101112-11012

Часть – С

№1. Выполните перевод чисел в соответствующую систему счисления:

|  |  |  |  |
| --- | --- | --- | --- |
| 2-я СС | 8-я СС | 10-я СС | 16-я СС |
| 1101,1 |  |  |  |
|  |  | 10,25 |  |

№2. Выполните расчеты

101112+1111012

100011012+1101102

11001012+110112

11010001012-1111110002

101100102-10100012

11011111002-1001000102

**Практическое занятие № 4. Элементы комбинаторики, теории множеств и математической логики**.

Операции «импликация», «эквиваленция». Логические функции. Законы алгебры логики. Эквивалентные преобразования логических выражений. Логические уравнения. Построение логического выражения с данной таблицей истинности.Дизъюнктивная нормальная форма. *Конъюнктивная нормальная форма.* Логические элементы компьютеров. Построение схем из базовых логических элементов. Дискретные игры двух игроков с полной информацией. Выигрышные стратегии.

Студент должен знать

Арифметические и логические основы работы компьютера.

Практические задания:

1. **Отметьте, какие из следующих предложений являются логическими высказываниями**

|  |  |
| --- | --- |
| **3+5=9** |  |
| **Луна есть спутник Земли.** |  |
| **Сегодня идет дождь.** |  |
| **Если два угла равны между собой, то они вертикальные.** |  |
| **Студент первого курса.** |  |
| **Чебоксары – столица Чувашии.** |  |
| **Все целые числа положительные.** |  |
| **У нее карие глаза.** |  |

1. **Отметьте истинные высказывания**

|  |  |
| --- | --- |
| **23 – простое число.** |  |
| **- 6 > 0** |  |
| **Информатика – это дисциплина.** |  |
| **Технические средства в английском языке обозначаются словом SoftWare.** |  |
| **1 Пбайт = 1024 Тбайт** |  |
| **Информационные технологии – это совокупность методов и устройств, используемых людьми для обработки информации.** |  |
| **Принтер – печатающее устройство.** |  |
| **МЭСМ – отечественная машина первого поколения** |  |

1. **Приведите примеры**

|  |  |
| --- | --- |
| **истинные  высказывания** |  |
|  |
| **ложные высказывания** |  |
|  |

1. **Установите соответствие**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **логическое умножение** |  | **дизъюнкция** |
|  | **логическое сложение** |  | **инверсия** |
|  | **логическое отрицание** |  | **конъюнкция** |

**Ответ: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. **Установите соответствие**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **высказывание (суждение)** |  | цепочка высказываний или утверждений, определенным образом связанных друг с другом |
|  | **умозаключение** |  | суждение, которое требуется доказать или опровергнуть |
|  | **утверждение** |  | некоторое предложение, которое может быть истинно (верно) или ложно |
|  | **рассуждение** |  | логическая операция, в результате которой из одного или нескольких данных суждений получается (выводится) новое суждение |

**Ответ: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. **Вставьте пропущенное слово**
   1. **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ - это часть электронной логической схемы, которая реализует элементарную логическую функцию.**
   2. **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ - это табличное представление логической схемы, в котором перечислены все возможные сочетания значений истинности входных сигналов вместе со значением истинности выходного сигнала для каждого из сочетаний.**
   3. **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** - связывает два простых логических выражения, из которых первое является условием (А), а второе (В)– следствием из этого условия.

**7.Пусть *а* = «это простое число», а *b* = «это четное число». Выразите следующие формулы на обычном языке:**

* + 1. ****
    2. ****
    3. ****
    4. ****
    5. ****
    6. ****

1. **Составьте таблицу истинности и схему логических элементов**

****

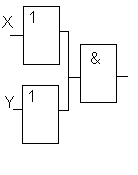
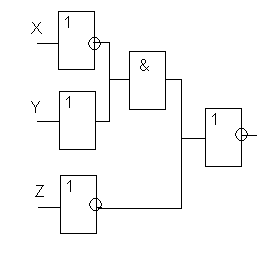
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **a** | **b** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **0** | **0** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **0** | **1** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **1** | **0** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **1** | **1** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

****

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **А** | **В** | **С** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **0** | **0** | **0** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **0** | **0** | **1** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **0** | **1** | **0** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **0** | **1** | **1** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **1** | **0** | **0** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **1** | **0** | **1** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **1** | **1** | **0** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **1** | **1** | **1** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

****

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **А** | **В** | **С** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **0** | **0** | **0** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **0** | **0** | **1** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **0** | **1** | **0** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **0** | **1** | **1** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **1** | **0** | **0** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **1** | **0** | **1** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **1** | **1** | **0** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **1** | **1** | **1** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

1. Составьте по схеме логическое выражение:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Практическое занятие № 5. Дискретные объекты.**

Решение алгоритмических задач, связанных с анализом графов (примеры: построения оптимального пути между вершинами ориентированного ациклического графа; определения количества различных путей между вершинами). Обход узлов дерева в глубину. *Упорядоченные деревья (деревья, в которых упорядочены ребра, выходящие из одного узла).* Использование деревьев при решении алгоритмических задач (примеры: анализ работы рекурсивных алгоритмов, разбор арифметических и логических выражений). Бинарное дерево. *Использование деревьев при хранении данных.* Использование графов, деревьев, списков при описании объектов и процессов окружающего мира.

Цель урока:

1. Получить знания о графах, их видах, свойствах.

Задачи урока:

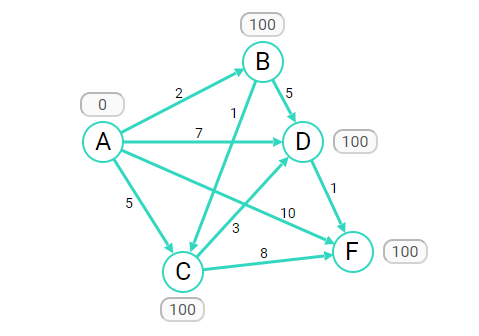
1. Получить навыки преобразования матрицы смежности (табличной формы представления информации) в граф;
2. Сформировать навык построения путей в графе и поиска кратчайшего пути.

*Понятие минимального пути в графе*

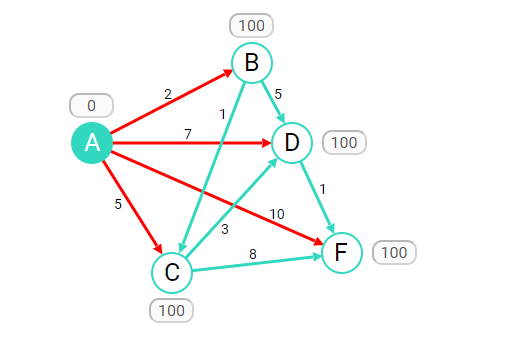
Условие задачи:

Дан следующий взвешенный ориентированный граф. Найти кратчайший путь от вершины A до F.

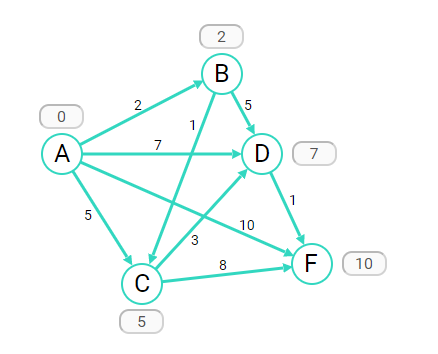
1. Присвоим вершине А метку равную 0, потому как эта вершина — начало. Остальным вершинам присвоим метки равные бесконечно большому числу (пусть это будет 100)



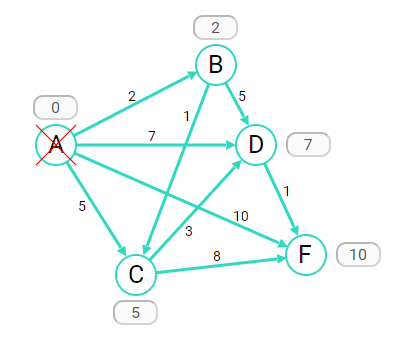
2. Выбери вершину, вес которой является минимальными все ребра выходящие из этой вершины



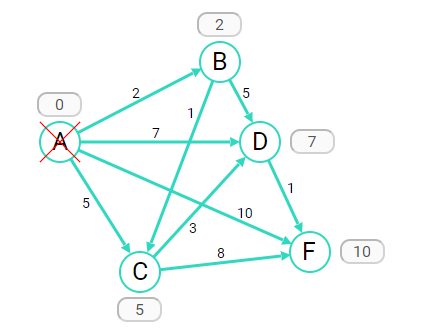
3. Вычисли сумму веса вершины источника и веса ребра для каждой конечной вершины. Если она окажется меньше веса вершины приемника, то изменим вес этой вершины.



4. Вычеркни вершину-источник

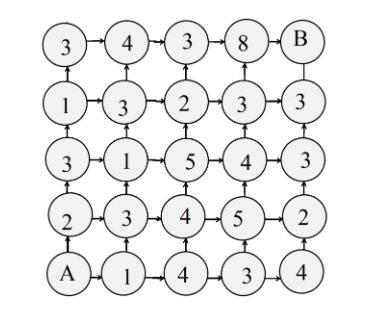


А теперь проделай все пункты 2,3,4 самостоятельно до тех пор, пока не будут вычеркнуты ВСЕ вершины

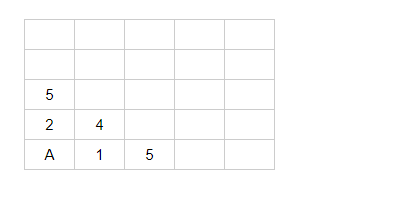


*Поиск кратчайшего пути методом динамического программирования*

Пусть дан некоторый лабиринт, соединяющий комнаты в виде графа. При этом заходя в каждую комнату, нужно заплатить пошлину. Необходимо пройти по нему от точки А до точки В, потратив наименьшее количество денег.

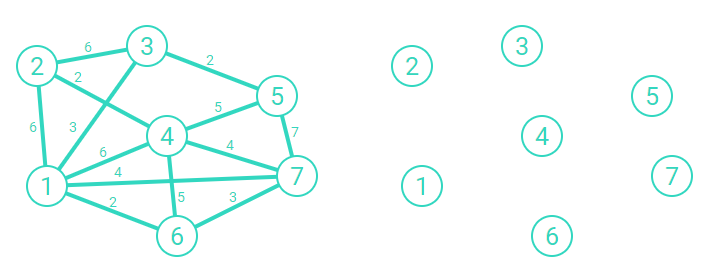


Составим таблицу, в которой каждая ячейка будет соответствовать определенной ячейке. Числа в ячейках будут равны минимальному числу пошлины, которое можно получить, пройдя от начала (A) до соответствующей клетки.



*Построение минимального остовного дерева*

Есть некоторое количество городов, между которыми можно проложить дороги различной стоимости (обозначим весом ребра). Необходимо выбрать набор дорог, соединяющий все города, минимальной суммарной стоимости.



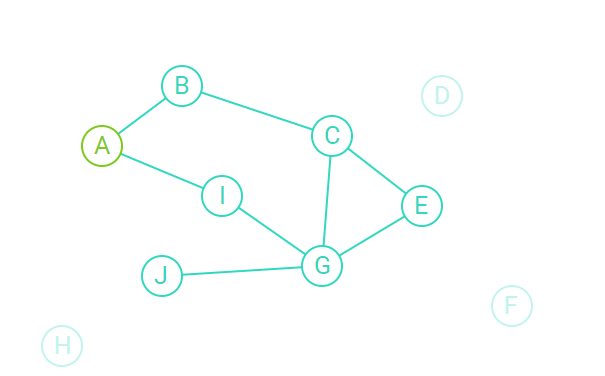
Дороги в стране требуют инспекции. За ремонт дороги отвечает мэр города, в который въезжает эта дорога. Давайте поможем инспектору пройти из города A в город J, посещая все города, не проходя одной дороги несколько раз.

*Алгоритм Краскала*

Переберите ребра от меньшего веса к большему, добавляя в каркас таким образом, чтобы в полученном графе не получилось цикла.

*Построение минимального остовного дерева*

Дороги с стране требуют инспекции. За ремонт дороги отвечает мэр города, в который въезжает эта дороги. Давайте поможем инспектору пройти из города A в город J, посещая все города, не проходя одной дороги несколько раз.



Графы делятся на:

Неориентированные и ориентированные (когда движение по ребру возможно только в одну сторону).

Взвешенные (когда у вершины или у ребра есть вес, отличающий его от другого) и невзвешенные.

И другие более сложные графы (мультиграф, псевдограф, изоморфный граф и другие).

Кратчайшим путем мы будем называть путь, если вершины соединены минимальным числом ребер (в случае, если граф невзвешенный) сумма ребер, соединяющих эти вершины, минимальна (для взвешенного графа).

Существует огромное количество алгоритмов, находящих кратчайший путь и один из них — это алгоритм Дейсктры. Еще один способом нахождения кратчайшего пути может служить «метод динамического программирования».

**Практическое занятие № 6. Алгоритмы и структуры данных**

Алгоритмы исследования элементарных функций, в частности – точного и приближенного решения квадратного уравнения с целыми и вещественными коэффициентами, определения экстремумов квадратичной функции на отрезке. Алгоритмы анализа и преобразования записей чисел в позиционной системе счисления. Алгоритмы, связанные с делимостью целых чисел. Алгоритм Евклида для определения НОД двух натуральных чисел.

## Алгоритмы. Разработка алгоритма решения задачи

Исключительно важно использовать язык блок-схем при разработке алгоритма решения задачи. Решение одной и той же задачи может быть реализовано с помощью различных алгоритмов, отличающихся друг от друга как по времени счета и объему вычислений, так и по своей сложности. Запись этих алгоритмов с помощью блок-схем позволяет сравнивать их, выбирать наилучший алгоритм, упрощать, находить и устранять ошибки.

Отказ от языка блок-схем при разработке алгоритма и разработка алгоритма сразу на языке программирования приводит к значительным потерям времени, к выбору неоптимального алгоритма. Поэтому необходимо изначально разработать алгоритм решения задачи на языке блок-схем, после чего алгоритм перевести на язык программирования.

При разработке алгоритма сложной задачи используется метод пошаговой детализации. На первом шаге продумывается общая структура алгоритма без детальной проработки отдельных его частей. Блоки, требующие детализации, обводятся пунктирной линией и на последующих шагах разработки алгоритма продумываются и детализируются.

В процессе разработки алгоритма решения задачи можно выделить следующие этапы:

* Этап 1 . Математическое описание решения задачи.
* Этап 2 . Определение входных и выходных данных.
* Этап 3 . Разработка алгоритма решения задачи.

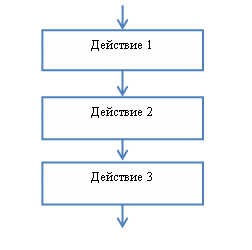
## Базовые алгоритмические конструкции

В теории программирования доказано, что для записи любого, сколь угодно сложного алгоритма достаточно трех базовых структур:

* Следование (линейный алгоритм);
* Ветвление (разветвляющийся алгоритм);
* Цикл-пока (циклический алгоритм).

### Линейные алгоритмы

Линейный алгоритм образуется из последовательности действий, следующих одно за другим. Например, для определения площади прямоугольника необходимо сначала задать длину первой стороны, затем задать длину второй стороны, а уже затем по формуле вычислить его площадь.



#### **Пример**

**ЗАДАЧА. Разработать алгоритм вычисления гипотенузы прямоугольного треугольника по известным значениям длин его катетов a и b.**

На примере данной задачи рассмотрим все три этапа разработки алгоритма решения задачи:

**Этап 1. Математическое описание решения задачи.**

Математическим решением задачи является известная формула:

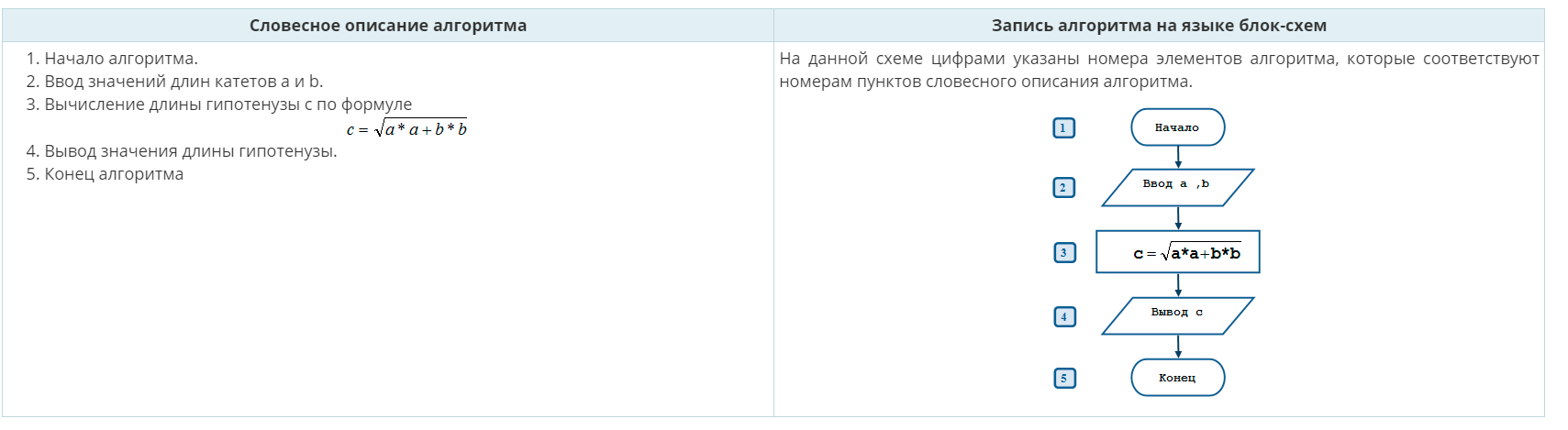
Формула,

где с-длина гипотенузы, a, b – длины катетов.

**Этап 2. Определение входных и выходных данных.**

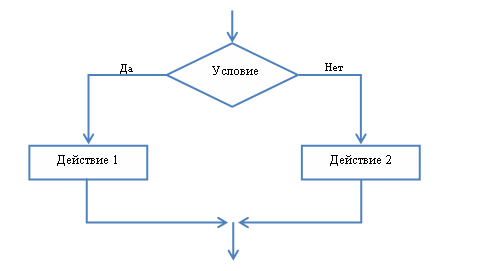
Входными данными являются значения катетов a и b. Выходными данными является длина гипотенузы – c.

**Этап 3. Разработка алгоритма решения задачи.**



### Разветвляющиеся алгоритмы

Алгоритм ветвления содержит условие, в зависимости от которого выполняется та или иная последовательность действий.



#### **Пример**

**ЗАДАЧА. Разработать алгоритм вычисления наибольшего числа из двух чисел x и y.**

**Этап 1. Математическое описание решения задачи.**

Из курса математики известно, если x > y, то наибольшее число x, если x < y, то наибольшее число y, если x = y, то число x равно числу y.

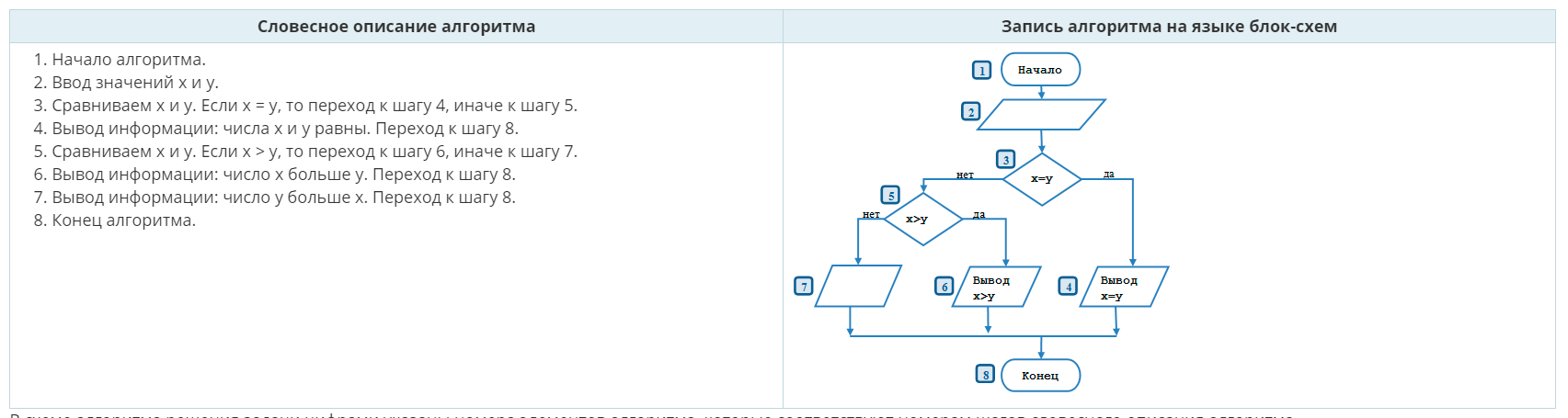
**Этап 2. Определение входных и выходных данных.**

Входными данными являются значения чисел x и y. Выходным данными являются:

* наибольшее число
* любое из чисел, если числа равны

Для решения задачи нам необходимо знать значения x и y.

**Этап 3. Разработка алгоритма решения задачи.**



В схеме алгоритма решения задачи цифрами указаны номера элементов алгоритма, которые соответствуют номерам шагов словесного описания алгоритма

В рассматриваемом алгоритме (рис.3) имеются три ветви решения задачи:

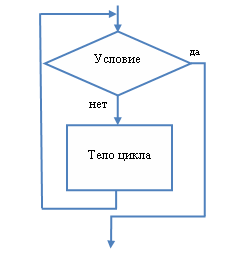
* первая: это элементы 1, 2, 3, 4, 8.
* вторая: это элементы 1, 2, 3, 5, 6, 8
* третья: это элементы 1, 2, 3, 5, 7, 8.

Выбор ветви определяется значениями x и y в элементах 3 и 5, которые являются условиями, определяющими порядок выполнения элементов алгоритма. Если условие (равенство), записанное внутри символа «решение», выполняется при введенных значениях x и y, то следующими выполняется элементы 4 и 8. Это следует из того, что они соединены линией с надписью «да» и направление (последовательность) вычислений обозначена стрелочкой.

Если условие в элементе 3 не выполняется, то следующим выполняется элемент 5. Он соединен с элементом 3 линией с надписью «нет». Если условие, записанное в элементе 5, выполняется, то выполняется элементы 6 и 8, в противном случае выполняются элементы 7 и 8.

### ****Циклические алгоритмы****

Циклический алгоритм **–**определяет повторение некоторой части действий (операций), пока не будет нарушено условие, выполнение которого проверяется в начале цикла. Совокупность операций, выполняемых многократно, называется телом цикла.

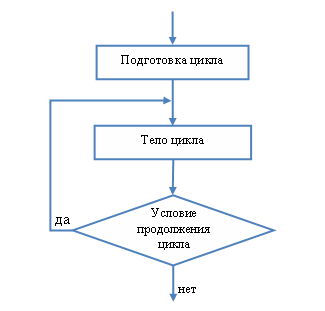


Алгоритмы, отдельные действия в которых многократно повторяются, называются *циклическими алгоритмами,* Совокупность действий, связанную с повторениями, называют *циклом*.

При разработке алгоритма циклической структуры выделяют следующие понятия:

* параметр цикла – величина, с изменением значения которой связано многократное выполнение цикла;
* начальное и конечное значения параметров цикла;
* шаг цикла – значение, на которое изменяется параметр цикла при каждом повторении.

Цикл организован по определенным правилам. Циклический алгоритм состоит из подготовки цикла, тела цикла и условия продолжения цикла.



В подготовку цикла входят действия, связанные с заданием исходных значений для параметров цикла:

* начальные значения цикла;
* конечные значения цикла;
* шаг цикла.

В тело цикла входят:

* многократно повторяющиеся действия для вычисления искомых величин;
* подготовка следующего значения параметра цикла;
* подготовка других значений, необходимых для повторного выполнения действий в теле цикла.

В условии продолжения цикла определяется допустимость выполнения повторяющихся действий. Если параметр цикла равен или превысил конечное значение цикла, то выполнение цикла должно быть прекращено.

#### **Пример**

**ЗАДАЧА. Разработать алгоритм вычисления суммы натуральных чисел от 1 до 100.**

**Этап 1. Математическое описание решения задачи.**

Обозначим сумму натуральных чисел через S. Тогда формула вычисления суммы натуральных чисел от 1 до 100 может быть записана так:

**сумма натуральных чисел**

где Xi – натуральное число X c номером i, который изменяется от 1 до n, n=100 – количество натуральных чисел.

**Этап 2. Определение входных и выходных данных.**

Входными данными являются натуральные числа: 1, 2, 3, 4, 5, …, 98, 99, 100.

Выходные данные – значение суммы членов последовательности натуральных чисел.

Параметр цикла**–**величина, определяющая количество повторений цикла. В нашем случае i – номер натурального числа.

Подготовка цикла заключается в задании начального и конечного значений параметра цикла.

* начальное значение параметра цикла равно 1,
* конечное значение параметра цикла равно n,
* шаг цикла равен 1.

Для корректного суммирования необходимо предварительно задать начальное значение суммы, равное 0.

Тело цикла. В теле цикла будет выполняться накопление значения суммы чисел, а также вычисляться следующее значение параметра цикла по формулам:

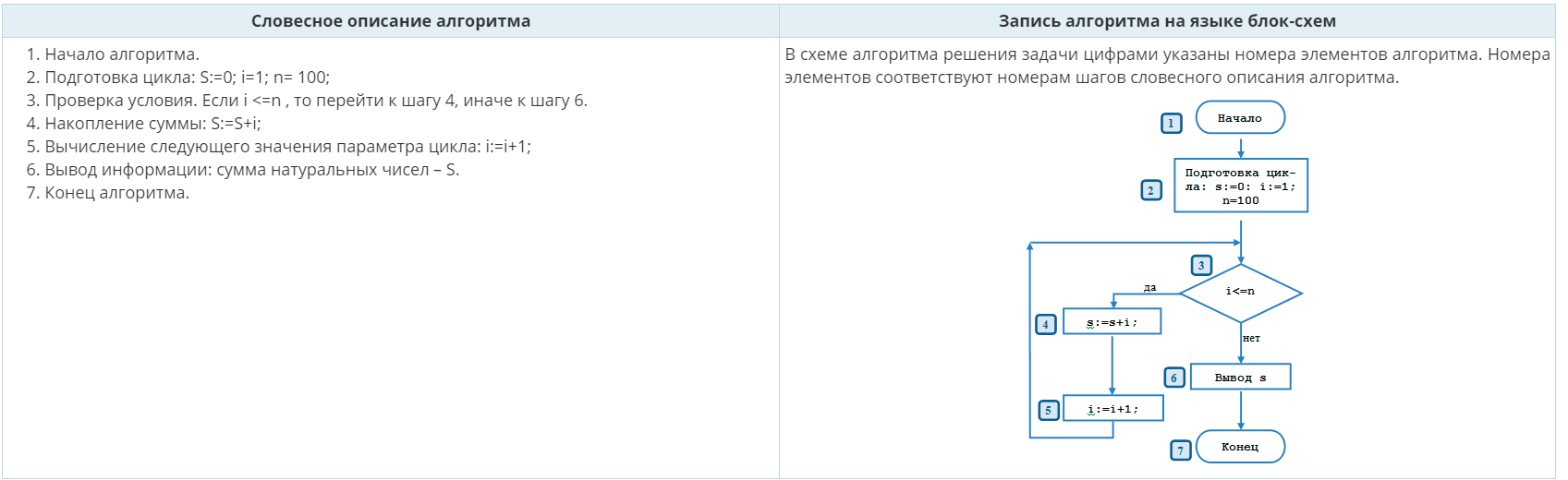
S=S+i;              I=I+1;

Условие продолжения цикла: цикл должен повторяться до тех пор, пока не будет добавлен последний член последовательности натуральных чисел, т.е. пока параметр цикла будет меньше или равен конечному значению параметра цикла.

**Этап 3. Разработка алгоритма решения задачи.**

Введем обозначения: S – сумма последовательности, i – значение натурального числа.

Начальное значение цикла i=1, конечное значение цикла i =100, шаг цикла 1.



**Практическое занятие № 7. Алгоритмы и структуры данных**

Алгоритмы линейной (однопроходной) обработки последовательности чисел без использования дополнительной памяти, зависящей от длины последовательности (вычисление максимума, суммы; линейный поиск и т.п.). Обработка элементов последовательности, удовлетворяющих определенному условию (вычисление суммы заданных элементов, их максимума и т.п.). Алгоритмы обработки массивов. Примеры: перестановка элементов данного одномерного массива в обратном порядке; циклический сдвиг элементов массива; заполнение двумерного числового массива по заданным правилам; поиск элемента в двумерном массиве; вычисление максимума и суммы элементов двумерного массива. *Вставка и удаление элементов в массиве.*

Типовые алгоритмы можно подразделить на алгоритмы обработки массивов, алгоритмы поиска и алгоритмы сортировки.

**Типовые алгоритмы обработки массивов:**

**1.****Суммирование двух массивов одинакового размера**

Задано: массивы A = (a1, a2, ..., an),    B = (b1,b2,...,bn).

Сформировать: массив   C =(c1, c2, ..., cn), где Сi = Ai + Bi; i = 1, 2, ..., n.

Задача сводится  к  организации цикла по i и вычислению Ci = Ai + Bi при каждом значении  i  от 1  до n.

Исходные данные:

N - размер массива;

A, B - массивы слагаемые размером N;

Результат:   массив  С -  размером N;

Вспомогательные  переменные:  I - индекс - управляющая переменная цикла.

Procedure SUM\_MAS (n : integer; A,B :mas; var C : mas);

{ где  mas  должен быть описан в главной программе в разделе описания типов, например так:

type mas = array[1..100 ] of  real ;

тогда это будет процедура для суммирования двух одномерных массивов размером не более 100 элементов  }

begin

for i := 1 to n do C[i] := A[i]+B[i];

end.

**2.  Суммирование  эле****ментов массива.**

Задано:  массив P = (P1,P2,...,Pn).

Определить:  сумму элементов массива.

Исходные данные:

N - размер массива;

P - массив  размером N;

Результат:   S - сумма элементов;

Вспомогательная переменная: I - индекс - управляющая переменная цикла.

Procedure SUMMA (n : integer; A :mas; var S : real );

{ процедура для суммирования элементов одномерного массива }

begin    S:=0; { обнуление переменной под сумму }

for i := 1 to n do   S := S+P[i]

end.

**3. Определ****ение числа элементов массива, удовлетворяющих  заданному условию.**

Задано:  массив  P = (P1,P2,...,Pn); T -  заданное число.

Определить: сколько элементов удовлетворяет заданному условию, например     Pi > T.

Исходные данные:

N - размер массива;

P - массив  размером N;

T - заданное значение, с которым сравниваются элементы массива.

Результат:  K - число элементов массива P, удовлетворяющих условию.

Вспомогательная переменная:  I- индекс - управляющая переменная цикла.

Procedure USLOVIE ( n : integer; P :mas; T: real; var K : integer);

{процедура  определения числа элементов, удовлетворяющих условию}

begin

k := 0;  { обнуление переменной под счетчик чисел }

for  i := 1  to  n  do  if  P[ i ] > T  then  k := k+1

end;

**4. Суммирова****ние  элементов  массива, удовлетворяющих  заданному  условию.**

Задано: массив P = (P1,P2,...,Pn); T - заданное число.

Определить: сумму элементов массива  P, удовлетворяющих заданному условию, например  Pi > T.

Исходные данные:

N - размер массива;

P - массив  размером N;

T - заданное значение, с которым сравниваются элементы массива;

Результат:  S - сумма элементов массива P, удовлетворяющих  условию.

Вспомогательная переменная :  I - индекс - управляющая переменная цикла.

Procedure SUM\_USLOV ( n : integer; P :mas; T: real; var S : real);

{процедура определения суммы элементов, удовлетворяющих  условию}

begin        S := 0;  {обнуление переменной под сумму  элементов}

for  i := 1  to  n  do  if  P [ i ] > T  then  S := S+1

end;

**5.  Инв****ертирование массива.**

Задано: массив C = (c1,c2,...,cn).

Требуется: изменить порядок следования элементов массива C на обратный, используя одну вспомогательную переменную.

Исходные данные:

N - размер массива;

C - массив  размером N;

Результат:

C - инвертированный  массив;

Вспомогательные переменные:

I -индекс, управляющая переменная цикла;

M = n/2 - вычисляется до входа в цикл для уменьшения объема вычислений; P - используется при перестановке двух элементов массива.

Procedure INVER\_MAS ( n : integer; C :mas; var C : mas);

Var   m :  integer;   p   : real;  { локальные переменные }

begin  m := n div 2 ;  {  целочисленное деление }

for i := 1 to  m do

begin  p := C[ i ];  C[i] := C[N-i+1]; C[N-i+1] := p  end;

end.

**6. Формирование массива из элементов другого массива, удовлетворяющих  заданному условию**

Задано: массив A = (a1, a2, ..., an), T - заданное число.

Сформировать: массив B = (b1, b2, ..., bn), состоящий из элементов массива, удовлетворяющих условию Ai > T.

Заметим, т.к. индексы элементов массивов A и B не совпадают (не все элементы массива Ai > T), то для обозначения индексов массива B  должна быть предусмотрена другая переменная.

Исходные данные:

N - размер массива;

A - массив размером N;

T - заданное значение;

Результат:

B - массив размером не больше N;

Y - число элементов массива  B;

Вспомогательная переменная: I - индекс - управляющая переменная цикла.

Procedure MAS\_NEW (n:integer;T:real;A:mas;var B: mas; var Y: byte);

{ где  mas  должен быть описан в главной программе в разделе описания типов, например так :

type mas = array[1..100 ] of  real ;

тогда это будет процедура для суммирования двух одномерных массивов размером не более 100 элементов  }

{ процедура включения в новый массив  элементов, удовлетворяющих условию }

begin   Y := 0; { обнуление ячейки под счетчик элементов массива В }

for i := 1 to n do

If  A[ i ] > T then  begin Y := Y+1; B[ Y ] := A[ i ]  end;

end.

**7.  Поиск  максимального  (минимального)  элемента в массиве с запоминанием  его  положения  в  массиве.**

Задано: массив A = (a1,a2,...,an).

Найти:  max (min)  элемент  массива  A  и  его индекс.

Исходные данные:

N - размер массива;

A - массив размером  N;

Результат:

A\_max - максимальный  элемент  массива A;

K -  его индекс.

Вспомогательная переменная: I - индекс управляющая переменная цикла.

Procedure MAX\_MAS1(n:integer; A :mas; var A\_max :real; var K byte);

{ процедура поиска  максимального элемента массива и его номера }

begin    A\_max := A[1]; K := 1;

for i := 2 to n do

If  A\_max<A[i] then  begin K := i; A\_max := A[i]  end;

end.

Примечание:  Если в массиве несколько max элементов (имеют одно и то же значение), то  в  K  будет запоминаться первый из них, а чтобы запоминался индекс последнего нужно заменить условие  на A\_max<=A(I). Поиск минимального элемента аналогичная процедура.

**8.  Поиск  заданного элемента в  массиве**

Задано: массив P=(p1,p2,...pn); элемент L (массив может быть как числовым так и символьным.

Найти: Есть ли в массиве P, элемент равный L. Результат присвоить символьной переменной.

Исходные данные:

N - размер массива;

P - массив размером N;

L - значение, которое ищется в массиве;

Результат: R - имеет значение "элемент, равный L есть" или "элемента, равного  L  нет" в  зависимости от результата поиска;

Вспомогательная переменная: I - индекс управляющая переменная цикла.

Procedure POISK ( n:integer; P :mas; L :integer; var R :string);

{ процедура поиска заданного значения среди элементов массива }

Label  m ;

begin

R :=" элемента равного L в массиве нет " ;

for i := 1 to n do

If  P[i] = L  then

begin  R := " элемент, равный  L  есть  ";  Goto m   end;

end.

Примечание. Если элемент, равный L, найден, то чтобы завершить цикл? используется оператор безусловного перехода Goto m, где локальная метка  m  обязательно  должна  быть  описана  в  процедуре.

**Практическое занятие № 8. Алгоритмы и структуры данных**

Рекурсивные алгоритмы, в частности: нахождение натуральной и целой степени заданного ненулевого вещественного числа; вычисление факториалов; вычисление n-го элемента рекуррентной последовательности (например, последовательности Фибоначчи). Построение и анализ дерева рекурсивных вызовов. Возможность записи рекурсивных алгоритмов без явного использования рекурсии.

Слово “**рекурсия**” происходит от латинского слова “**recursio**” – возвращение.

**Рекурсия** — определение, описание, изображение объекта или процесса внутри самого этого объекта или процесса, т.е. когда объект является частью самого себя.

В математике рекурсия имеет отношение к методу определения функций и числовых рядов: *рекурсивно заданная* функция определяет своё значение через обращение к себе самой с другими аргументами. С рекурсией тесно связана математическая индукция: она является естественным способом доказательства свойств функций на натуральных числах, рекурсивно заданных через свои меньшие значения.

**Рекурсия** в программировании — это способ организации вспомогательного алгоритма (подпрограммы), при котором эта подпрограмма обращается ***сама к себе*** напрямую (**прямая рекурсия**) или через другие функции (**косвенная рекурсия**).

Простейшим примером рекурсии является линейная рекурсия, когда функция содержит единственный условный вызов самой себя. В таком случае рекурсия становится эквивалентной обычному циклу.

**Пример 1.** Рекурсивная функция для расчета факториала заданного числа n:

n!=1⋅2⋅…⋅n, причем 0!=1 и 1!=1.

Кроме того, выполняется следующее соотношение:

n!=n⋅(n-1)!

Используя это соотношение, можно составить рекурсивную функцию для вычисления факториала:

**long int** F(**int** n)   
**{**  
 **if (**n > 1)  F=n \* F(n – 1);  
 **else** F=1;  
**}**

**Пример 2.** Вычисление n-го члена последовательности чисел Фибоначчи: 0, 1, 1, 2, 3, 5, 8, 13, 21 и т.д.

#include <iostream>

using namespace std;

int **fibonachi**(int n)

{

if (n>1)return(fibonachi(n-1)+fibonachi(n-2));

else return(n);

}

int main( )

{ int n,p;

cin>>n;

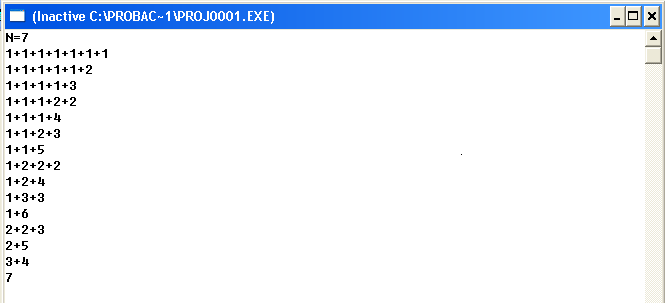
p=**fibonachi**(n);

cout<<"p="<<p;

}

**Разложение числа на слагаемые**

**Задача 1.** Вывести на экран все варианты разложения заданного натурального числа N на слагаемые. Разложения, отличающиеся только порядком слагаемых, считаются одинаковыми.



**Решение.**

Будем раскладывать число *N* на два слагаемых. Затем второе из полученных слагаемых снова будем раскладывать на 2 слагаемых и т.д., т.е. можно использовать рекурсию.

Чтобы исключить из рассмотрения повторяющиеся последовательности. Для этого будем рассматривать только неубывающие последовательности слагаемых (каждое следующее слагаемое не меньше предыдущего).

Обозначим:***n1*** *-* первое слагаемое в разложении.

Таким образом, на каждом шаге будем получать сумму двух слагаемых n1 и N-n1. Вновь полученное число N-n1 снова раскладывать на 2 слагаемых и т.д. Все слагаемые будем записывать в массив. Для этого нам понадобится массив a[N] из N элементов, поскольку самое длинное представление – это сумма N единиц.

Первый вызов подпрограммы надо выполнить при n1=1 (т.е. первое слагаемое будет равно 1), первое слагаемое в разложении хранится в массиве под номером 0 (т.е. pos=0). Поэтому первый вызов подпрограммы R(N, 1, 0).

Для поиска первого слагаемого в разложении (обозначим его *n1*) будем проверять все числа, начиная с 1 до N/2. Например, для числа 7 достаточно проверить в качестве первого слагаемого числа 1, 2, 3; для числа 10 – числа от 1 до 5).

После выбора первого слагаемого надо решить задачу разложения на слагаемые числа, равного N=N-n1. Параметрами создаваемой рекурсивной функции (ее имя — R) будут:

n – число, раскладываемое на слагаемые (не только заданное);

n1 – первое слагаемое числа n;

a[50] – массив, в котором хранятся слагаемые разложения (50 – максимальное с запасом кол-во возможных слагаемых в разложении).

Первый вызов подпрограммы надо выполнить при n1=1 (т.е. первое слагаемое будет равно 1), первое слагаемое в разложении хранится в массиве под номером 0 (т.е. pos=0). Поэтому первый вызов подпрограммы R(0, N, 1).

В подпрограмме в качестве первого слагаемого разложения будем перебирать числа от n1 до n/2 в цикле c параметром *i*:

1) очередное число *i* надо сохранить в массиве a[50] под номером pos, т.е. a[pos]=i;

2) снова обратиться к подпрограмме R. При очередном вызове подпрограммы надо увеличить номер в массиве на 1 (т.е. pos=pos+1), уменьшить раскладываемое на слагаемые число на *i* (т.е. n=n-i) и в качестве первого слагаемого нового числа снова выбрать число *i*, т.е. R(pos+1,n-i,i).

После перебора всех чисел *i* необходимо учесть в записи разложения также само число *n* (то, что от него осталось), и вывести полученный результат на экран.

#include <iostream>

using namespace std;

int a[50]; //глобальный массив для хранения слагаемых разложения

void R(int pos, long int n, int n1)

{int i;

for (i=n1 ; i<=n/2;i++)

{a[pos]=i;

R(pos+1,n-i,i);

}

for (i=0;i<=pos-1;i++)

cout<< a[i]<<"+";

cout<<n;

cout<<endl;

}

int main(void)

{

long int N;

cin>>N;

R(0,N,1);

}

*Например,* число n=7.

Цикл

for (i=n1 ; i<=n/2; i++)

будет работать при i=1, i=2 и i=3. При каждом i , будет выполняться рекурсивный вызов подпрограммы:

7

2

5

2

3

3

4

1

6

1

5

1

4

1

3

1

2

1

1

2

4

3

3

2

2

2

3

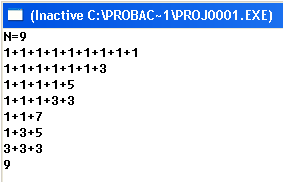
2

2

Надо распечатать элементы массива a[…] , начиная с 0-го по 1 (т.к. pos-1=1). Будет напечатано: 1+1+1+1+1+1+

Надо еще напечатать число 1 (т.е. последнее полученное значение числа n) и перейти на новую строку.

**Задача 2.** Вывести на экран все варианты разложения заданного натурального числа N на нечетные слагаемые.



**Решение.** Алгоритм несущественно отличается от предыдущего: слагаемые должны быть нечетными. Очередное разложение закончено, если полученное значение числа n является нечетным; в этом случае его надо распечатать.

#include <iostream>

using namespace std;

int a[50];

void R(int pos,long n, int n1)

{ int i;

for (i=n1;i<=n/2;i++)

**if (i%2!=0)**

{a[pos]=i;

R(pos+1,n-i,a[pos]);

}

**if (n%2!=0)**

{for (i=0;i<=pos-1;i++)

cout<< a[i]<<"+";

cout<<n;

cout<<endl;

}

}

int main( )

{

long int N;

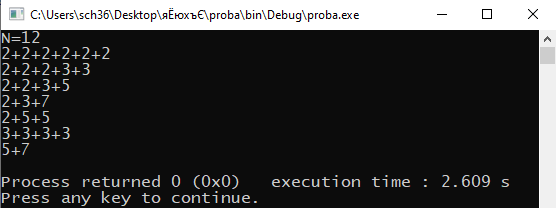
cin>>N;

R(0,N,1);

}

**Замечание.** Для разложения числа на четные слагаемые надо изменить выделенные строки на **if (i%2==0)** и на **if (n%2==0)**. При первом вызове первое слагаемое равно 2, т.е. R(0,N,2). Эта задача имеет решение, если N – четное число.

**Задача 3.** Вывести на экран все варианты разложения заданного натурального числа N на простые слагаемые.



**Решение.** Алгоритм отличается от предыдущего тем, что вместо проверки на нечетность будем проверять, является ли слагаемое простым числом. Это будет проверяться в подпрограмме **simple**.

#include <iostream>

using namespace std;

int a[50];

bool simple(int a)

{ int k=0;

for (int i=2; i<=a; i++)

if (a%i==0)k++;

if(k==1)return true;else return false;

}

void R(int pos,long n, int n1)

{ int i;

for (i=n1;i<=n/2;i++)

**if (simple(i))**

{a[pos]=i;

R(pos+1,n-i,a[pos]);

}

**if (simple(n))**

{for (i=0;i<=pos-1;i++)

cout<< a[i]<<"+";

cout<<n;

cout<<endl;

}

}

int main( )

{

long int N;

cout<<"N=";

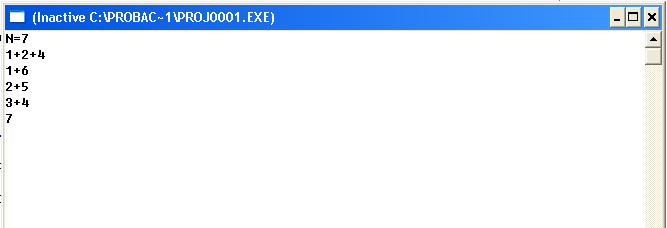
cin>>N;

R(0,N,1);

}

**Замечание.** Слагаемые в разложении числа могут удовлетворять различным условиям (например, являться числами Фибоначчи и др.). Для решения подобных задач можно использовать предыдущую программу, изменив подпрограмму для проверки заданного условия.

**Задача 4.** Вывести на экран все варианты разложения заданного натурального числа N на различные слагаемые в порядке возрастания (слагаемые не могут повторяться).



**Решение.** Алгоритм несущественно отличается от предыдущих: чтобы следующее слагаемое в разложении было больше предыдущего, подпрограмму надо вызывать с параметрами R(pos+1,n-i,**i+1**). Чтобы для четных чисел не было разложения на два одинаковых слагаемых, первое слагаемое разложения должно меняться от 1 до (n-1)/2 (например, для чисел 7 и 8 в качестве первого слагаемого могут быть числа 1,2,3).

#include <iostream>

using namespace std;

int a[50];

void R(int pos,long n, int n1)

{ int i;

for (i=n1 ; i<=**(n-1)/2**;**i++**)

{a[pos]=i;

R(pos+1,n-i,**i+1**);

}

for (i=0;i<=pos-1;i++)

cout<<a[i]<<"+";

cout<<n;

cout<<endl;

}

int main( )

{

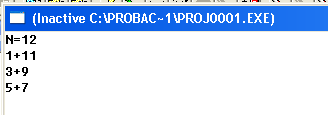
long N;

cin>>N;

R(0,N,1);

}

**Задача 4.** Вывести на экран все варианты разложения заданного натурального числа N на различные нечетные слагаемые в порядке возрастания (слагаемые не могут повторяться).



**Решение.** В предыдущую программу надо добавить два условия:

#include <iostream>

using namespace std;

int a[50];

void R(int pos,long n, int n1)

{int i;

for (i=n1 ; i<=(n-1)/2;i++)

**if (i%2!=0)**

{a[pos]=i;

R(pos+1,n-i,i+1);

}

**if (n%2!=0)**

{for (i=0;i<=pos-1;i++)

cout<<a[i]<<"+";

cout<<n;

cout<<endl;

}

}

int main( )

{

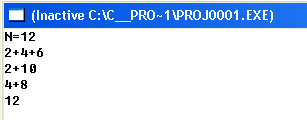
long N;

cin>>N;

R(0,N,1);

}

**Замечание.** Для разложения числа на различные четные слагаемые надо изменить выделенные строки на **if (i%2==0)** и на **if (n%2==0)**. При первом вызове первое слагаемое равно 2, т.е. R(0,N,2). Эта задача имеет решение, если N – четное число.



**Практическое занятие № 9. Алгоритмы и структуры данных**

Сортировка одномерных массивов. Квадратичные алгоритмы сортировки (пример: сортировка пузырьком). Слияние двух отсортированных массивов в один без использования сортировки. Алгоритмы анализа отсортированных массивов. Рекурсивная реализация сортировки массива на основе слияния двух его отсортированных фрагментов.

Алгоритмы анализа символьных строк, в том числе: подсчет количества появлений символа в строке; разбиение строки на слова по пробельным символам; поиск подстроки внутри данной строки; замена найденной подстроки на другую строку.

**Основные понятия.**

1. Массив – это совокупность элементов, которые имеют одно и то же имя и тип. Элементы массива различаются по индексу. Имя общее, индекс оригинальный. Упорядоченность данных в массиве позволяет обращаться к любому элементу массива по его номеру (индексу), а однотипность данных позволяет использовать циклическую обработку всех элементов
2. Массив описывается в разделе var оператором следующего вида:

var ИмяМассива: array [НИ .. ВИ] of Тип; где НИ (нижний индекс) ‒ целочисленный номер 1-

го элемента массива; **..** ‒ оператор диапазона Паскаля; ВИ (верхний индекс) ‒ целочисленный номер последнего элемента; Тип ‒ любой из известных типов данных Паскаля. Каждый элемент массива будет рассматриваться как переменная соответствующего типа.

1. Как правило, ввод, обработка и вывод массива осуществляются поэлементно, с использованием цикла с параметром. Простейший способ ввода ‒ ввод массива с клавиатуры
2. Над массивами нельзя выполнять арифметические действия (вычитать, складывать и др.). Все действия выполняются поэлементно.

**Задание**

1. Составить и записать алгоритм решения задачи в графическом и словесноформульном виде
2. Написать программу на языке Паскаль
3. Ввести программу и запустить еѐ на исполнение.
4. Протестировать алгоритм с различными данными.
5. Отладить программу (найти и исправить возможные ошибки).
6. Проанализировать программу и сделать вывод.

**Исходные данные:**

Дан целочисленный массив размера N. Вывести все содержащиеся в данном массиве отрицательные числа в порядке убывания их индексов, а также их количество.

**Решение:**

1. Алгоритм задачи:

а) т.к. выводить элементы массива будем с конца, то первым делом необходимо

ввести весь массив, для этого организуем цикл, в котором будет только ввод данных;

б) для вывода результата и подсчета отрицательных элементов организуем цикл с

параметром и с шагом – 1;

в) когда цикл закончит работу, выведем подсчитанное количество нужных элементов.

Получили алгоритм:

Начало

Ввод (

*N*

)

K

=0

Вывод

(

K

)

Конец

I=1,

N,1

K=K+1

Ввод (

A[I]

)

I=N,1

,1

A[I]<0

В

ы

вод (

A[I]

)

**+**

**‒**

* 1. Начало;
  2. Ввод (*N*);
  3. K=0;
  4. Для I = 1 до N выполнитьВвод (A[I]);
  5. Для I = N до 1 выполнить

Если A[I]<0 то

нц

Вывод (A[I]); K = K + 1; кц

* 1. Вывод (K);
  2. Конец.

1. Программа на языке Паскаль:

program massiv1**;** label m1; var I, N, K:integer;

A:array [1..100] of integer; begin m1:

write(‘Введите целое положительное двузначное число ’);

readln(N); if N>100 then begin

writeln(‘Число введено неправильно’); goto m1; end;

K:=0;

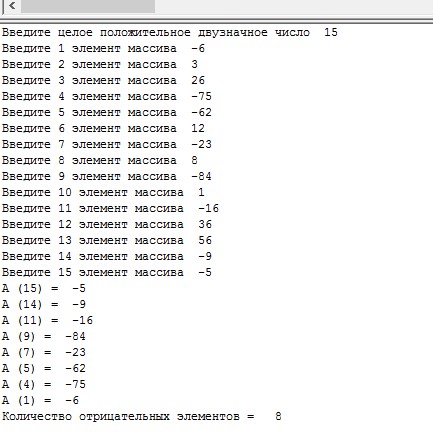
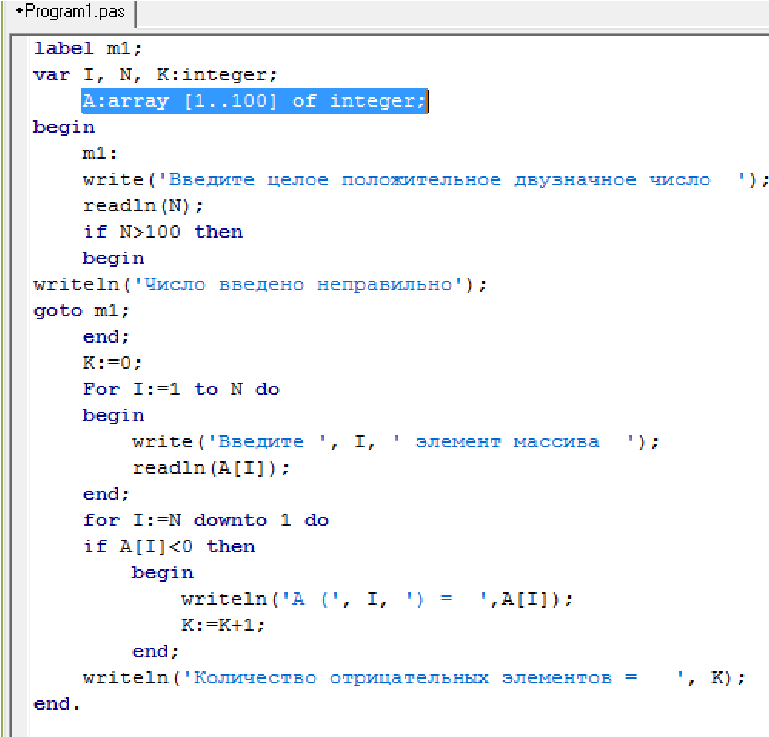
For I:=1 to N do begin

write(‘Введите ‘, I, ‘ элемент массива ’); readln(A[I]); end; for I:=N downto 1 do if A[I]<0 then begin

writeln(‘A (‘, I, ‘) = ’,A[I]); K:=K+1; end;

writeln(‘Количество отрицательных элементов = ’, K); end.

3 Программный код для запуска 4 Тестирование программы с различными данными:



5 Программа выдает верные значения отрицательных элементов и правильно подсчитывает их количество, но Паскаль не работает с динамическими массивами, поэтому размер массива не должен превышать 100 элементов если пользователь ввел число, большее 100, программа предлагает повторить ввод.

**Задания к практической работе.**

1. Дан целочисленный массив размера N. Вывести все содержащиеся в данном массиве четные числа в порядке возрастания их индексов, а также их количество.
2. Дан целочисленный массив размера N. Вывести все содержащиеся в данном массиве нечетные числа в порядке убывания их индексов, а также их количество.
3. Дан целочисленный массив размера N. Вывести вначале все содержащиеся в данном массиве четные числа в порядке возрастания их индексов, а затем все содержащиеся в данном массиве нечетные числа в порядке убывания их индексов.
4. Дан целочисленный массив размера N. Вывести все содержащиеся в данном массиве числа, кратные 3 в порядке возрастания их индексов, а также их количество и сумму.
5. Дан массив размера N. Вычислить среднее арифметическое элементов массива и вывести результат.
6. Дан массив размера N. Вывести первый отрицательный элемент массива и его номер, а также сумму элементов данного массива.
7. Дан массив размера N. Вывести максимальный элемент данного массива и его номер.
8. Дан массив размера N. Вывести минимальный элемент данного массива и его номер.
9. Дан массив размера N. Вывести второй положительный элемент массива и его номер.
10. Дан массив размера N. Вывести второй отрицательный элемент массива и его номер.

**Практическое занятие № 10. Алгоритмы и структуры данных**

Построение графика функции, заданной формулой, программой или таблицей значений*.* Алгоритмы приближенного решения уравнений на данном отрезке, например, методом деления отрезка пополам. Алгоритмы приближенного вычисления длин и площадей, в том числе: приближенное вычисление длины плоской кривой путем аппроксимации ее ломаной; приближенный подсчет методом трапеций площади под графиком функции, заданной формулой, программой или таблицей значений. *Приближенное вычисление площади фигуры методом Монте-Карло. Построение траекторий, заданных разностными схемами. Решение задач оптимизации*. *Алгоритмы вычислительной геометрии. Вероятностные алгоритмы.* Сохранение и использование промежуточных результатов. Метод динамического программирования. Представление о структурах данных.Примеры: списки, словари, деревья, очереди. *Хэш-таблицы.*

Нелинейные уравнения можно разделить на 2 класса - алгебраические и трансцендентные. Алгебраическими уравнениями называют уравнения, содержащие только алгебраические функции (целые, рациональные, иррациональные). В частности, многочлен является целой алгебраической функцией. Уравнения, содержащие другие функции (тригонометрические, показательные, логарифмические и др.) называются трансцендентными.

Методы решения нелинейных уравнений делятся на две группы:

* точные методы;
* итерационные методы (за счет последовательных приближений получить решение уравнения с необходимой точностью).

Точные методы решения уравнений основываются на поиске равносильных преобразований алгебраических выражений, например, перенос слагаемых из одной части уравнения в другую с противоположным знаком, деление обеих частей уравнения на одинаковое число не равное 0, а также точные способы решений позволяют записать корни уравнения в виде некоторого конечного соотношения (формулы). Точные решения существуют только для некоторых уравнений определенного вида (линейные, квадратные, тригонометрические и др.), поэтому для большинства уравнений приходится использовать методы приближенного решения с заданной точностью (графические или численные). В первую очередь это относится к большинству трансцендентных уравнений. Доказано также, что нельзя построить формулу, по которой можно было бы решить произвольное алгебраическое уравнение выше четвертой степени.



Точные методы решения Приближенные методы решения



Например, уравнение нельзя решить путем равносильных алгебраических преобразований. Но это уравнение можно решать приближенно графическими и численными методами.

Решение уравнения проводят численно в два этапа. На первом этапе производится отделение корней - поиск интервалов, на которых содержится только по одному корню. Второй этап решения связан с уточнением корня на выбранном интервале (определением значения корня с заданной точностью). Далее будут рассмотрены несколько численных методов и приведены алгоритмы нахождения корней уравнений.

Отделение корней уравнения может проводиться графически, т.е. путем построения графика функции *y=f(x).* Для уравнения вида *f (x) = 0*, где *f(x)* – некоторая непрерывная функция, корень (или корни) этого уравнения являются точкой (или точками) пересечения графика функции с осью абсцисс.

Численные методы

Метод половинного деления

Метод хорд

Метод касательных

Комбинированный метод

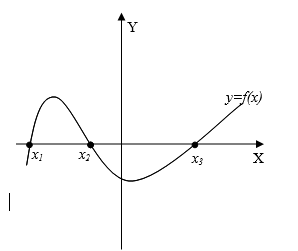
Графический метод

f(x)=0,   
где f(x) - непрерывная функция

Решение уравнений с заданной точностью

Отделение корней

Приближенные методы



Отделение корней уравнения можно осуществить путем построения компьютерных моделей:

* построение графика функции с помощью одного из языков программирования (в данном случае Free Pascal);
* построение графика функции в электронных таблицах Microsoft Excel путем построения диаграммы типа *График*.

При построении графика функции корни уравнения можно получить лишь с небольшой степенью точности. Поэтому, чтобы эти значения получить с любой заданной степенью точности, необходимо применять методы, которые позволяют «уточнять» найденные значения.

Рассмотрим методы уточнения корней и их основные идеи. Отметим следующий момент: при прочих равных условиях, тот метод уточнения корней будет более эффективен, в котором результат с той же погрешностью найден за меньшее число раз вычисления функции *f(x).*

**1.1. Метод половинного деления**

Самый простой из них – метод половинного деления, или иначе метод дихотомии. Метод дихотомии получил свое название от древнегреческого слова διχοτομία, что в переводе означает деление надвое. Его мы используем довольно часто. Допустим, играя в игру "Угадай число", где один игрок загадывает число от 1 до 100, а другой пытается его отгадать, руководствуясь подсказками "больше" или "меньше". Логично предположить, что первым числом будет названо 50, а вторым, в случае если оно меньше - 25, если больше - 75. Таким образом, на каждом этапе неопределенность неизвестного уменьшается в 2 раза. Т.е. даже самый невезучий в мире человек отгадает загаданное число в данном диапазоне за 7 предположений вместо 100 случайных утверждений.

Алгоритм метода половинного деления основан на теореме Больцано - Коши о промежуточных значениях непрерывной функции и следствии из неё.

Теорема Больцано - Коши: если непрерывная функция принимает два значения, то она принимает любое значение между ними.

Следствие (теорема о нуле непрерывной функции): если непрерывная функция принимает на концах отрезка положительное и отрицательное значения, то существует точка, в которой она равна 0.

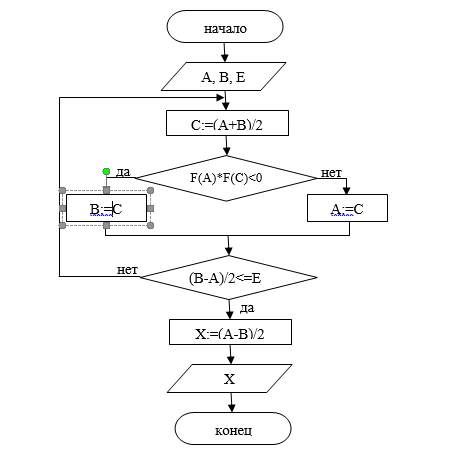
Алгоритм:

1. Задать отрезок [a,b] и погрешность *ε*.
2. Вычислить *c=(a+b)/2*
3. Определить интервал дальнейшего поиска: если *f(a)* и *f(c)* имеют разные знаки, т.е. *f(a)\*f(c)<0*, то *b:=c*, в противном случае *a:=c.*

|  |  |
| --- | --- |
| *f(a)\*f(c)<0 (да)* | *f(a)\*f(c)<0 (нет)* |
|  |  |

1. Если длина нового отрезка , то вычислить значение корня c=(a+b)/2 и остановиться, в противном случае перейти к шагу 2.

Блок-схема метода половинного деления



**1.2. Метод хорд**

При решении уравнения методом хорд нелинейная функция *f(x)* на отделенном интервале [a, b] заменяется линейной, в качестве которой берется хорда – прямая, стягивающая концы нелинейной функции. Вычисляются значения функции на концах отрезка, и строится "хорда", соединяющая точки *(a, f(a))* и *(b, f(b))*.

При решении нелинейного уравнения методом хорд задаются интервал [a,b] на котором существует только одно решение, и точность ε. Затем через две точки с координатами *(a, f(a))* и *(b, f(b))* проводим отрезок прямой линии (хорду) и определяем точку пересечения этой линии с осью абсцисс, точка *c*. Если при этом *f(a)∙f(c)<0*, то правую границу интервала переносим в точку *с (b=c)*. Если указанное условие не выполняется, то в точку *c* переносится левая граница интервала *(а=с)*. Поиск решения прекращается при достижении заданной точности *|f(c)|<* ε.

|  |  |
| --- | --- |
| *f(a)∙f(c)<0 (да)* | *f(a)∙f(c)<0 (нет)* |
|  |  |

Запишем уравнение прямой, проходящей через точки с координатами *(a, f(a))* и   
*(b, f(b))*:





Прямая заданная полученным уравнением пересекает ось абсцисс при условии *у=0*. Найдем точку пересечения хорды с осью *Х*.



Итак, 

Далее необходимо вычислить значение функции в точке *с*. Если | f(c) | < 0, то полученное число и есть корень уравнения с выбранной точностью, иначе необходимо построить следующую хорду и выполнить все рассмотренные ранее действия.

**Практическая часть**

Рассмотрим решение уравнения .

Компьютерная модель приближенного решения уравнений вида *f(x)=0* состоит из двух частей:

1. построение графика функции *y = f(x)* с целью определения промежутков, на которых находится один корень уравнения;
2. уточнение корня уравнения методами: половинного деления, хорд, касательных и комбинированного.
   1. **Компьютерная модель построения графика функции на языке программирования Free Pascal**

//Установка графического режима экрана, описание переменных, необходимых при //работе программы, задание функции

Uses Graph,Crt;

label 1;

var gr,gm,n,i:integer;

x,y,A,B,E,C:real;

function f(x:real): real;

begin

f:=x\*x\*x-sin(x);

end;

begin

gr:=0;

InitGraph (gr,gm, '');

if GraphResult < > grOk then Halt (1);

SetColor (3); //задание цвета

//построение системы координат

Line (200,100,200,400);

Line (450,270,100,270);

Line (200,100,195,105);

Line (200,100,205,105);

Line (450,270,445,265);

Line (450,270,445,275);

OutTextXY (455,270, 'x');

OutTextXY (185,100, 'y');

OutTextXY (230,280, '1');

OutTextXY (150,280, '-1');

OutTextXY (210,240, '1');

OutTextXY (210,300, '-1');

//шкала по оси Х

x:=110;

while x<450 do

begin

line (trunc(x),265,trunc(x),275);

x:=x+30

end;

//шкала по оси Y

y:=120;

while y<400 do

begin

line (195,trunc(y),205,trunc(y));

y:=y+30

end;

//построение графика

x:=-3;

while x<3 do

begin

x:=x+0.001;

y:=f(x);

PutPixel (trunc(x\*30)+200,-trunc(y\*30)+270,5)

end;

Для построения графика используется алгоритмическая конструкция «цикл». График строится путем построения точек с координатами (*х; у*) значения аргумента меняются от   
-3 до 3 с шагом 0,001, а значения функции вычисляются по формуле . Полученные точки строим с помощью оператора PutPixel, в скобках указываем координаты точек, которые надо построить и номер цвета, которым будет построен график.

Координаты точек, которые строятся, должны быть целыми числами, поэтому используется функция *trunc,* чтобы отбросить дробную часть.

Пиксель – это очень маленькая точка экрана, поэтому для построения графика функции координаты *х* и *y* необходимо умножить на величину единичного отрезка который я взяла (т.е. на 30, тогда увеличивается масштаб). Начало компьютерной системы координат расположено в левом верхнем углу, а наша система координат смещена на 200 пикселей по оси *Х* и на 270 пикселей по оси *Y*, поэтому прибавляем 200 и 270. Ось *Y* на компьютере направлена сверху вниз, наша ось *Y* снизу вверх, поэтому еще необходим знак «минус» перед значением функции *y*.

График функции .



По графику функции можно сделать вывод, что рассмотренное уравнение имеет три корня, расположенные на отрезках [-1; -0,7], [-0,3, 0,3], [0,7; 1].

Далее в пунктах 2.2 – 2.5 будут рассмотрены компьютерные модели второй части программы приближенного решения уравнения.

* 1. **Компьютерная модель метода половинного деления.**

CloseGraph; // закрытие графического режима и возврат в текстовый режим

write ('vvedi kolichestvo korney ');

readln (n);

for i:=1 to n do

begin

write ('vvedite levuy granicu otrezka ');

readln (A);

write ('vvedite pravuy granicu otrezka ');

readln (B);

write ('vvedite tochnost ');

readln (E);

Repeat

C:=(A+B)/2;

writeln(c:10:8);

if abs (f(C))<=0.0001 then goto 1;

if f(A)\*f(C)<0

then B:=C

else A:=C

until (B-A)/2<=E;

1: writeln ('koren uravneniya ', c:10:8);

end;

Результаты компьютерного эксперимента

vvedi kolichestvo korney 3

vvedite levuy granicu otrezka -1

vvedite pravuy granicu otrezka -0.7

vvedite tochnost 0.001

-0.85000000

-0.92500000

-0.96250000

-0.94375000

-0.93437500

-0.92968750

-0.92734375

-0.92851563

koren uravneniya -0.92851563

vvedite levuy granicu otrezka -0.3

vvedite pravuy granicu otrezka 0.3

vvedite tochnost 0.001

0.00000000

koren uravneniya 0.00000000

vvedite levuy granicu otrezka 0.7

vvedite pravuy granicu otrezka 1

vvedite tochnost 0.001

0.85000000

0.92500000

0.96250000

0.94375000

0.93437500

0.92968750

0.92734375

0.92851563

koren uravneniya 0.92851563

Таким образом, решения уравнения

* 1. **Компьютерная модель метода хорд.**

begin

clrscr;

write ('vvedi kolichestvo korney ');

readln (n);

for i:=1 to n do

begin

write ('vvedite levuy granicu otrezka ');

readln (A);

write ('vvedite pravuy granicu otrezka ');

readln (B);

write ('vvedite tochnost ');

readln (E);

repeat

c:=a-(f(a)/(f(b)-f(a)))\*(b-a);

writeln (c:10:8);

if f(c)\*f(a)>0 then a:=c else b:=c;

until abs(f(c))<e;

writeln ('koren uravneniya ',c:10:8);

end;

readln

end.

Результаты компьютерного эксперимента

vvedi kolichestvo korney 3

vvedite levuy granicu otrezka -1

vvedite pravuy granicu otrezka -.7

vvedite tochnost 0.001

-0.89655455

-0.92513533

-0.92825863

koren uravneniya -0.92825863

vvedite levuy granicu otrezka -0.3

vvedite pravuy granicu otrezka 0.3

vvedite tochnost 0.001

0.00000000

koren uravneniya 0.00000000

vvedite levuy granicu otrezka 0.7

vvedite pravuy granicu otrezka 1

vvedite tochnost 0.001

0.89655455

0.92513533

0.92825863

koren uravneniya 0.92825863

Таким образом, решения уравнения

**Практическое занятие № 11. Языки программирования**

Подпрограммы (процедуры, функции). Параметры подпрограмм. Рекурсивные процедуры и функции. Логические переменные. Символьные и строковые переменные. Операции над строками. Двумерные массивы (матрицы). *Многомерные массивы.* Средства работы с данными во внешней памяти. Файлы.

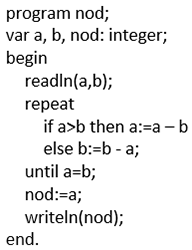
Для любой логической задачи можно составить алгоритм, используя лишь три базовых структуры: следование, ветвление и повторение. Эта теорема является одним из ключевых положений структурного программирования, о котором мы с вами сегодня и будем говорить.

Основные принципы структурного программирования заключаются в том, что:

1. Любая программа строится из трёх базовых управляющих конструкций: последовательность, ветвление, цикл.
2. В программе базовые управляющие конструкции могут быть вложены друг в друга произвольным образом.
3. Повторяющиеся фрагменты программы можно оформить в виде подпрограмм. В виде подпрограмм можно оформить логически целостные фрагменты программы, даже если они не повторяются.
4. Все перечисленные конструкции должны иметь один вход и один выход.
5. Разработка программы ведётся пошагово, методом «сверху вниз».

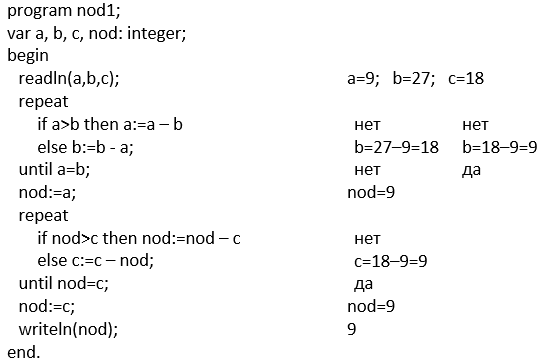
Давайте узнаем, что такое подпрограммы, какими они могут быть и как их использовать.

На прошлом уроке мы рассмотрели один из базовых алгоритмов обработки числовых данных — алгоритм нахождения НОД натуральных чисел.



Вернемся к нему для решения задачи нахождения НОД трех натуральных чисел *a*, *b*, *c*. Для решения воспользуемся следующим математическим фактом: если *a, b, c* — три натуральных числа, то НОД (*a, b, c*) = НОД (НОД (*a, b*)*, c*). Иначе говоря, нужно найти НОД двух величин, а затем НОД полученного значения и третьего числа.

Получаем:



Аналогичным образом можно было решить задачу нахождения НОД четырех, пяти и т. д. чисел. Понятно, что при этом код программы увеличился бы за счет добавления однотипных блоков. А можно ли каким-либо образом сократить запись алгоритма? Давайте разберемся.

В программировании ситуация, когда приходится выполнять одни и те же действия, является весьма типичной. Мы имеем алгоритм, который предназначен для решения подзадачи, выделенной в основной задаче. Выписывать алгоритм для решения подзадачи каждый раз заново в том месте программы, где он встречается, нерационально. Язык программирования Паскаль позволяет выделить любой алгоритм, описывающий подзадачу, и записать его только один раз, дав ему имя, а уже затем использовать его столько раз, сколько необходимо. Такой алгоритм называется **подпрограммой** и располагается в разделе описания до основной программы.

Подпрограммы подразделяются на **процедуры** (*procedure*) и **функции** (*function*).

**Процедуры**

Описание процедуры в Паскале имеет следующий формат:

https://resh.edu.ru/uploads/lesson_extract/5818/20190215142943/OEBPS/objects/c_info_11_4_1/627be478-208e-485f-a955-70b2d81aa77a.png

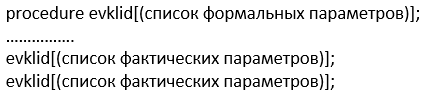
Оформим в нашем примере в качестве процедуры подпрограмму нахождения НОД двух натуральных чисел.



Теперь разберемся, какая служебная информация находится в круглых скобка рядом с именем процедуры.

Подпрограмма реализует некоторый частный алгоритм, который вызывается в основной программе. Для реализации алгоритма подпрограмме могут понадобиться некоторые данные из основной программы. Обработав эти данные, подпрограмма возвращает результат основной программе, если в этом есть необходимость. Передавать информацию из основной программы в подпрограмму и наоборот мы можем только через переменные, которые носят специальное название — параметры. Именно они и указываются в круглых скобках рядом с именем процедуры.

Переменные, указанные непосредственно в заголовке процедуры, называются формальными параметрами. То, что подставляется вместо них при вызове процедуры в основной программе, называется фактическими параметрами:



Квадратные скобки указывают на то, что список формальных параметров может отсутствовать, т.е. возможна процедура без параметров. Параметры могут быть параметрами-переменными и параметрами-значениями. Параметры-переменные записываются следующим образом:

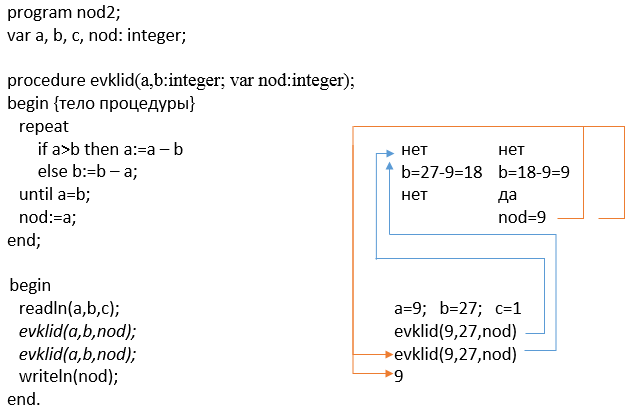
https://resh.edu.ru/uploads/lesson_extract/5818/20190215142943/OEBPS/objects/c_info_11_4_1/b31a3cc0-543e-43d5-9181-13746991dabe.png

Параметры-значения указываются так:

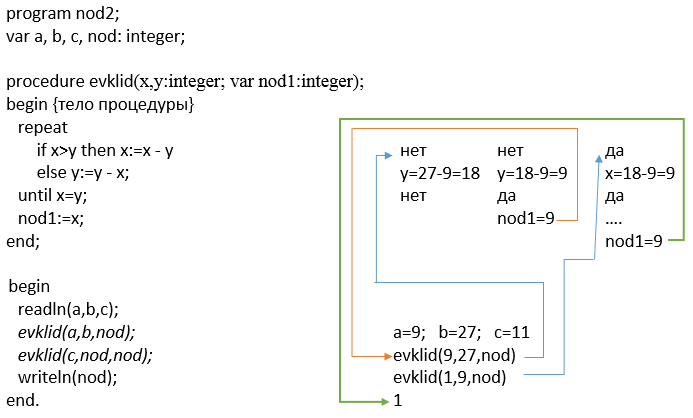
https://resh.edu.ru/uploads/lesson_extract/5818/20190215142943/OEBPS/objects/c_info_11_4_1/391e6772-b8c6-4754-a4e5-5c5510eb197a.png

Чаще всего параметры-значения служат для работы с входной в процедуру информацией, а параметры-переменные — для работы с выходной информацией.

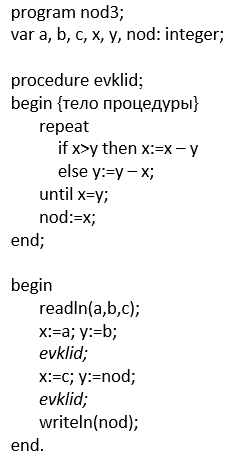
Вернемся к программе nod1. Наша процедура evklid должна получить на вход две переменные, найти для них НОД и передать его в основную программу. Получаем:



Здесь мы замечаем, что наша процедура evklid дважды вызывается одинаковым образом. Но это нарушает саму идею решения и может привести к неверному ответу. Чтобы исправить эту ошибку, введем формальные параметры, имена которых отличны от фактических:



Теперь рассмотрим другой вариант программы, решающий ту же задачу. В ней используется процедура без параметров.



Чтобы разобраться в этом примере, введем одно важное понятие — **область действия описания**.

Областью действия описания любого программного объекта (переменной, типа, константы и т. д.) является тот блок, на который это описание распространяется. Если данный блок вложен в другой (подпрограмма), то присутствующие во вложенном блоке описания являются локальными. Они действуют только в пределах внутреннего блока. Описания же, расположенные во внешнем блоке, называются глобальными по отношению к внутреннему блоку. Если глобально описанный объект используется во внутреннем блоке, то на него распространяется внешнее (глобальное) описание.

В программе nod2 переменные x, y, nod1 являются локальными внутри процедуры; переменные a, b, c — глобальные. Однако внутри процедуры переменные a, b, c не используются. Связь между внешним блоком и процедурой осуществляется через параметры.

В программе nod2 все переменные являются глобальными. В процедуре evklid нет ни одной локальной переменной. Здесь обмен значениями между основной программой и процедурой осуществляется через глобальные переменные.

**Функции**

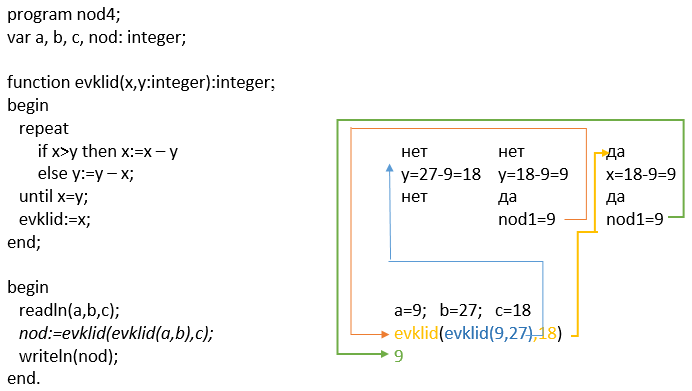
Теперь выясним, что такое подпрограмма-функция. Обычно функция используется в том случае, когда результатом работы подпрограммы должна быть скалярная (простая) величина. Тип результата называется типом функции.

Формат описания функции следующий:

https://resh.edu.ru/uploads/lesson_extract/5818/20190215142943/OEBPS/objects/c_info_11_4_1/b85c080e-1ce8-4c9e-bc46-f48906d95c04.png

Как и у процедуры, у функции служебная информация содержит список формальных параметров, среди которых могут присутствовать как параметры-значения, так и параметры-переменные. Все это — аргументы функции. Параметры могут вообще отсутствовать, если аргументы передаются глобально.

Программа решения рассмотренной выше задачи с использованием функции будет выглядеть следующим образом:



Сравнивая приведенные выше программы, можно сделать вывод, что программа nod4 имеет определенные преимущества перед другими. Функция позволяет получить результат путем выполнения одного оператора присваивания. Здесь также демонстрируется возможность того, что фактическим параметром при обращении к функции может быть эта же функция.

**Рекурсия**

Среди всевозможных подпрограмм особый интерес представляют так называемые рекурсивные подпрограммы.

Рекурсивная подпрограмма (функция, процедура) — подпрограмма, содержащая в своем описании вызов самой себя.

**Пример 1***.* Как известно, факториал натурального числа определяется следующим образом:

n! = 1·2·3·…·n.

0! = 1.

Иначе это можно записать так:

F(n)=1 при n ≤ 1;

F(n)=F(n–1) · n при n > 1.

В определении факториала через рекурсию имеется условие n ≤ 1, при достижении которого вызов рекурсии прекращается. Такое условие (оно называется граничным) в рекурсивном определении должно присутствовать обязательно!

**Пример 2***.* Алгоритм вычисления функции F(n), где n — натуральное число, задан следующими соотношениями:

F(n)=1 при n ≤ 2;

F(n)=F(n–1) +3· F(n–2) при n > 2.

Требуется выяснить, чему равно значение функции F(7).

По условию, F(1) = F(2) = 1.

F(3) = F(2) + 3 · F(1) = 1 + 3 · 1 = 4.

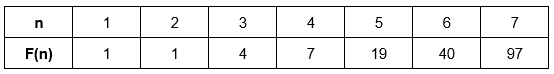
F(4) = F(3) + 3 · F(2) = 4 + 3 · 1 = 7.

F(5) = F(4) + 3 · F(3) = 7 + 3 · 4 = 19.

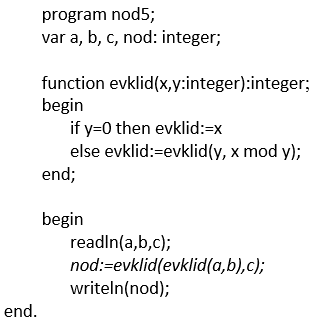
F(6) = F(5) + 3 · F(4) = 19 + 3 · 7 = 40.

F(7) = F(6) + 3 · F(5) = 40 + 3 · 19 = 97.

Подобные вычисления можно проводить в уме, а их результаты фиксировать в таблице:



И в завершение вспомним задачу вычисления НОД и модифицируем с использованием рекурсивной функции.



Применение рекурсивных методов для решения вычислительных задач не всегда эффективно. В большинстве случаев для решения той же задачи можно построить оптимальный не рекурсивный алгоритм. В то же время существуют задачи не вычислительного содержания, решить которые без использования рекурсии оказывается крайне проблематичным. К числу таких задач относится, например, известная головоломка под названием «Ханойские башни», с которой вы можете познакомиться в дополнительном материале.

**Практическое занятие № 12. Языки программирования**

Подробное знакомство с одним из универсальных процедурных языков программирования. Запись алгоритмических конструкций и структур данных в выбранном языке программирования. Обзор процедурных языков программирования. *Представление о синтаксисе и семантике языка программирования.*

*Понятие о непроцедурных языках программирования и парадигмах программирования. Изучение второго языка программирования.*

Язык программирования Паскаль, который был создан швейцарским профессором Никлаусом Виртом в 1969 году и получил свое название в честь Блеза Паскаля — изобретателя первого вычислительного механического устройства.

Основными элементами языка являются:

1. Алфавит, состоящий из латинских букв, цифр и специальных символов.
2. Служебные слова, значение которых в языке программирования строго определено.
3. Постоянные и переменные величины, которые характеризуются своими типами.
4. Знаки операций.
5. Стандартные функции и выражения — формулы, по которым вычисляется некоторое значение.

Порядок выполнения операций определяется скобками и приоритетом операций.

Программа на языке Pascal имеет следующую структуру:

— заголовок программы;

— раздел описания переменных;

— раздел описания постоянных величин, т. е. констант;

— собственно программы, которая начинается со слова begin и заканчивается словом end с точкой.

Обязательным разделом является только сама программа — набор операторов, обозначающих действия, выполняемые над данными.

Все операторы отделяются друг от друга символом “;”.

В языке Паскаль есть следующие операторы:

— оператор ввода с клавиатуры;

— оператор вывода на экран;

— оператор присваивания;

— условный оператор;

— оператор цикла с предусловием;

— оператор цикла с постусловием;

— оператор с увеличивающимся или с уменьшающимся параметром;

— составной оператор. Он применяется в случае, когда на месте одного оператора должны стоять сразу несколько операторов.

При решении многих задач программирования очень часто используются некоторые базовые алгоритмы. Например:

1. Обмен значениями двух переменных.
2. Нахождение наибольшего среди трех различных чисел.
3. Нахождение НОД двух чисел.
4. Нахождение суммы некоторого конечного количества чисел.
5. Нахождение количества чисел, удовлетворяющих некоторому условию.

При анализе готовых программ и проверки их соответствия решаемым задачам удобно использовать **трассировочные таблицы**. Они бывают двух типов:

Первый — каждая строка этих таблиц отражает результат одного действия.

Второй — каждая строка отражает результат выполнения группы действий.

**Упражнения:**

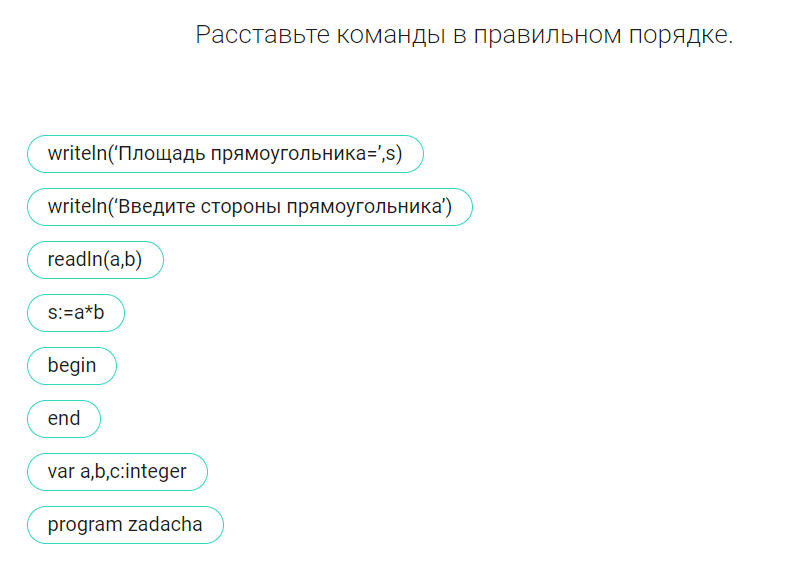
**Задание 1.**



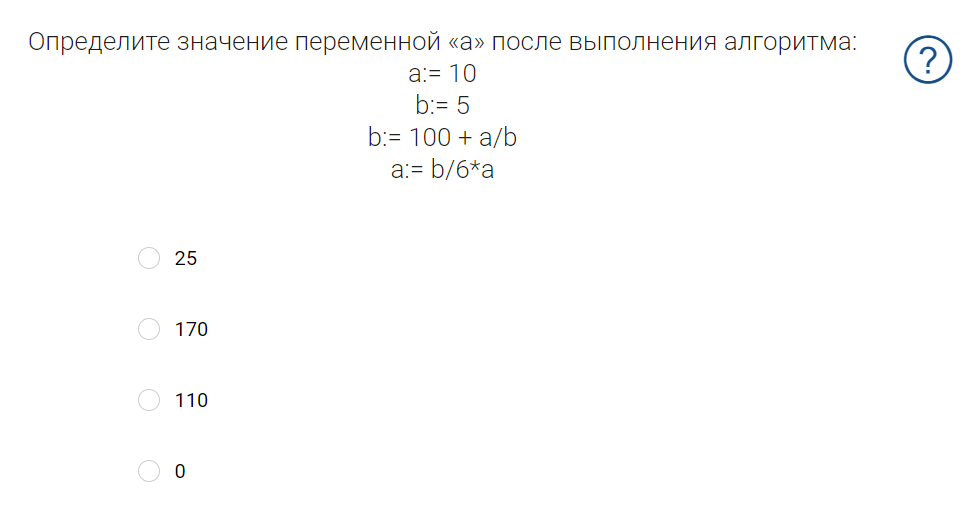
**Задание 2.**



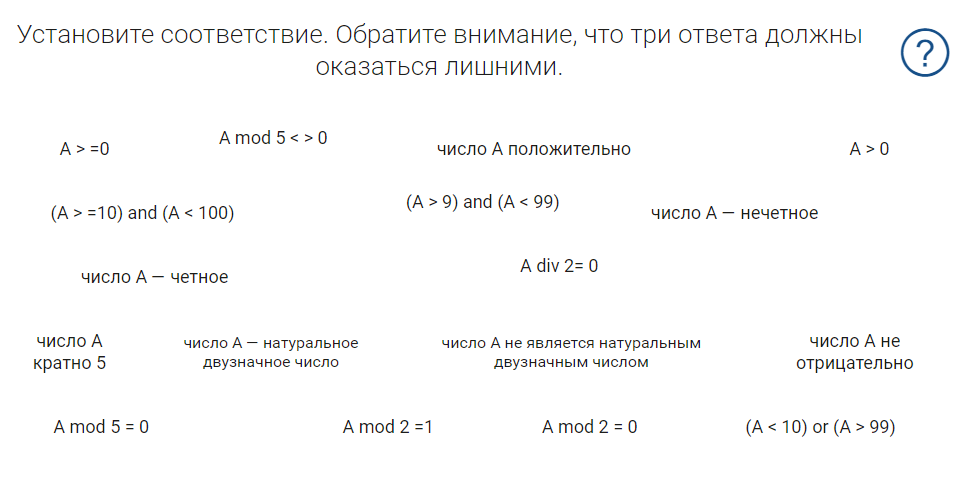
**Задание 3.**



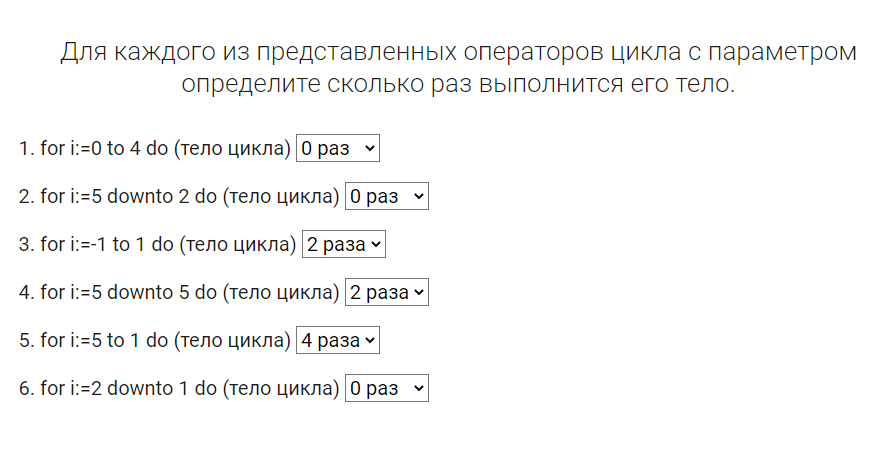
**Задание 4.**



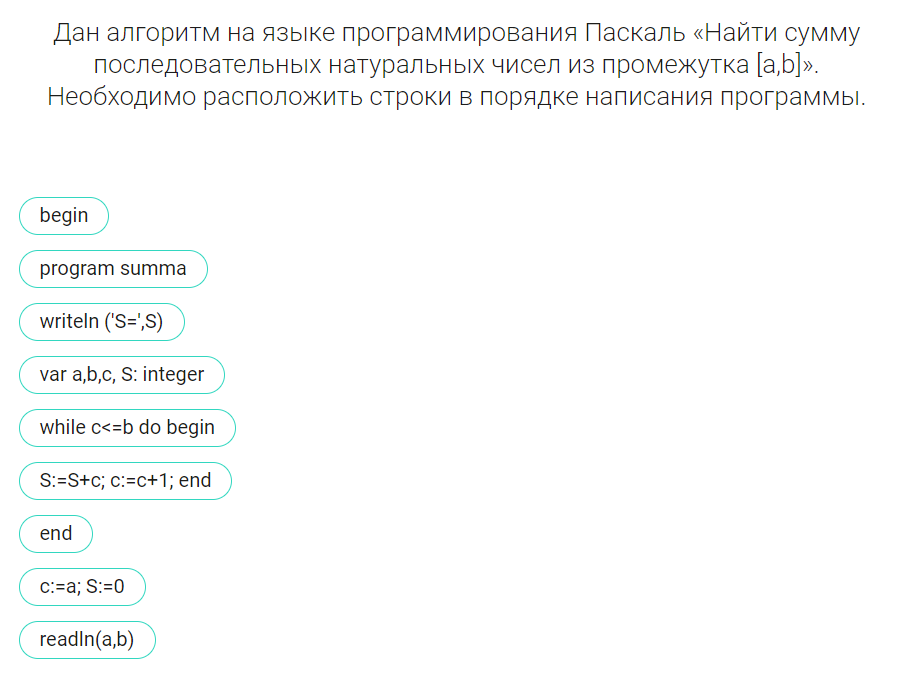
**Задание 5.**



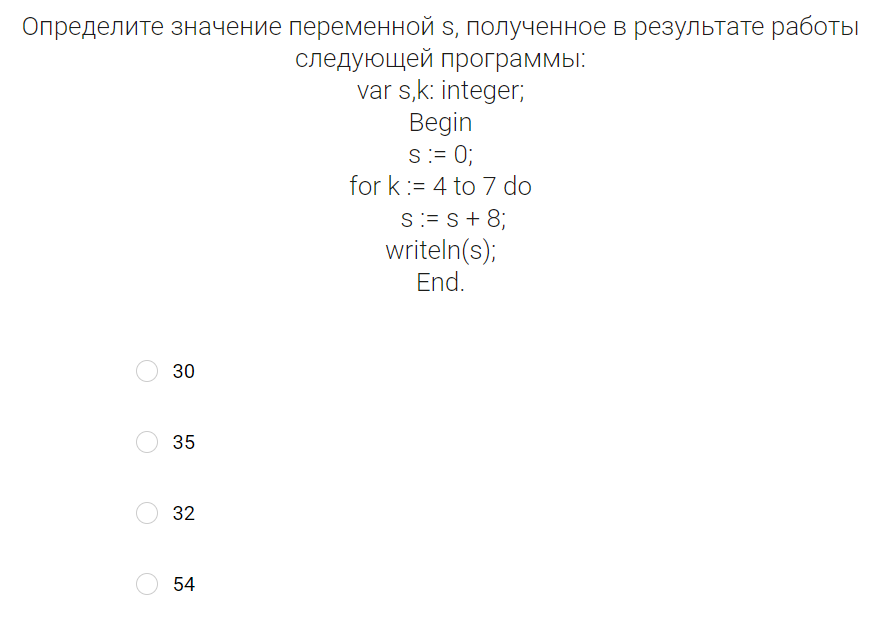
**Задание 6.**



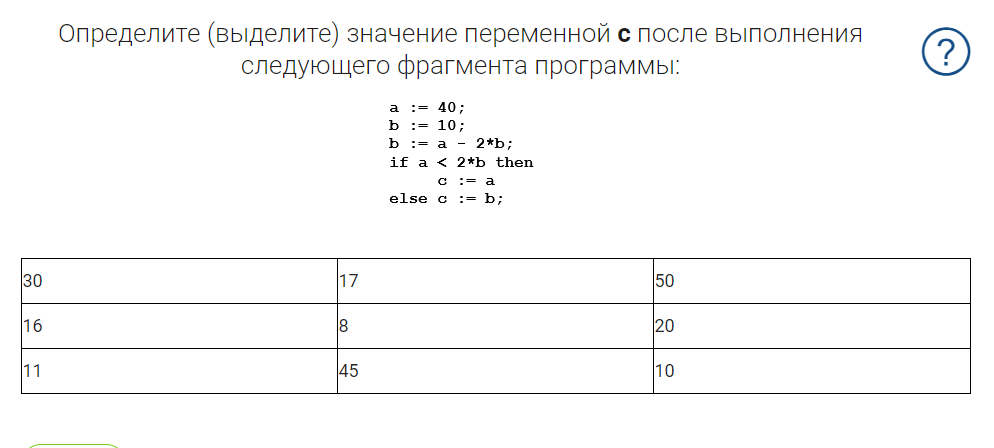
**Задание 7.**



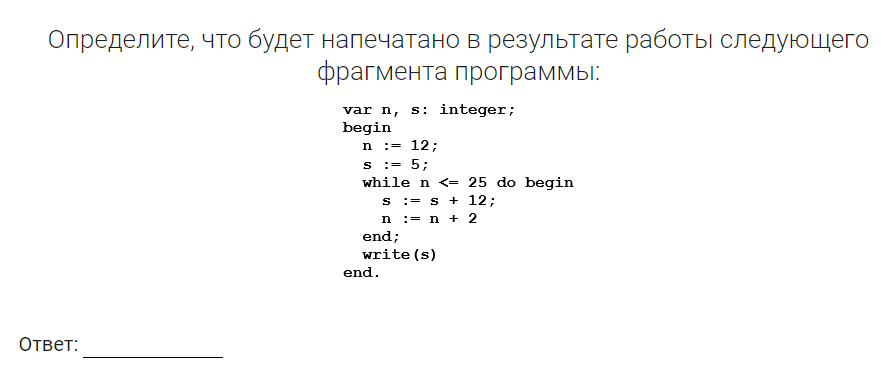
**Задание 8.**



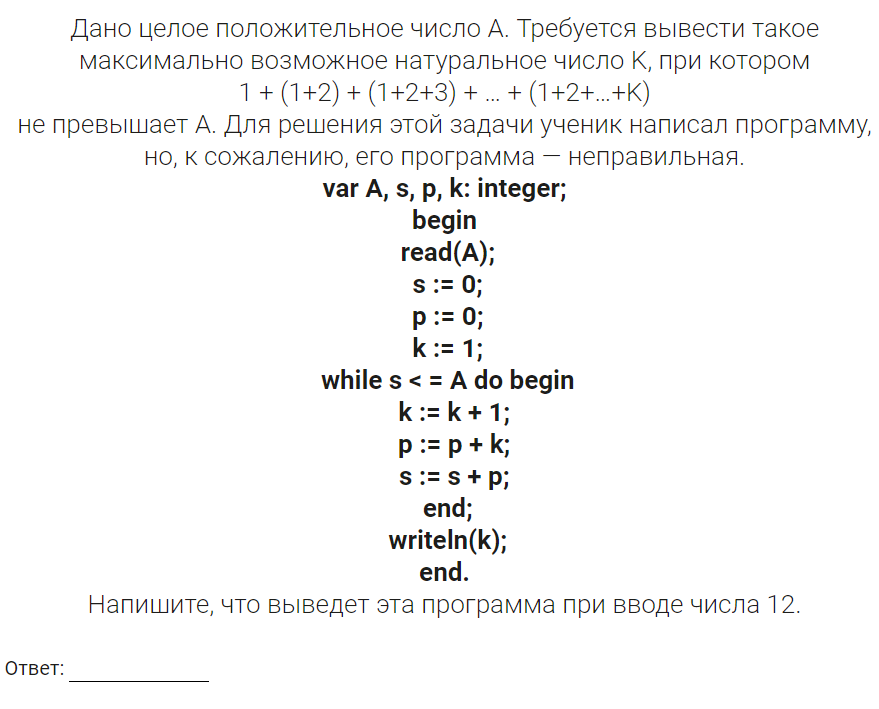
**Задание 9.**



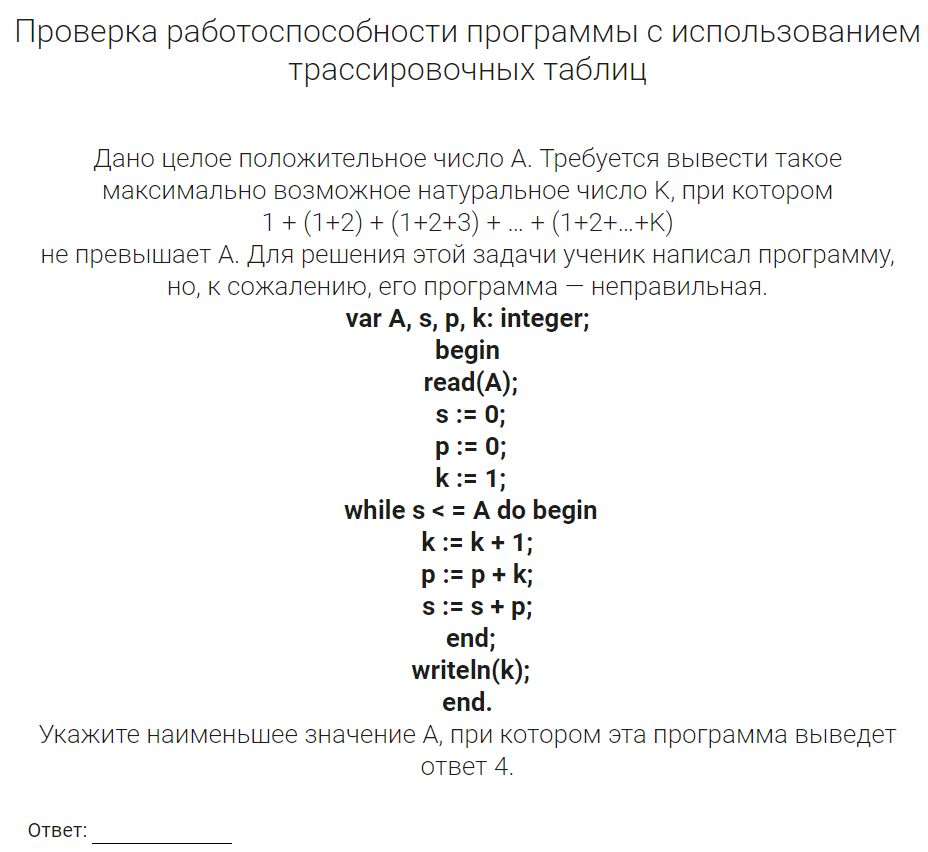
**Задание 10.**



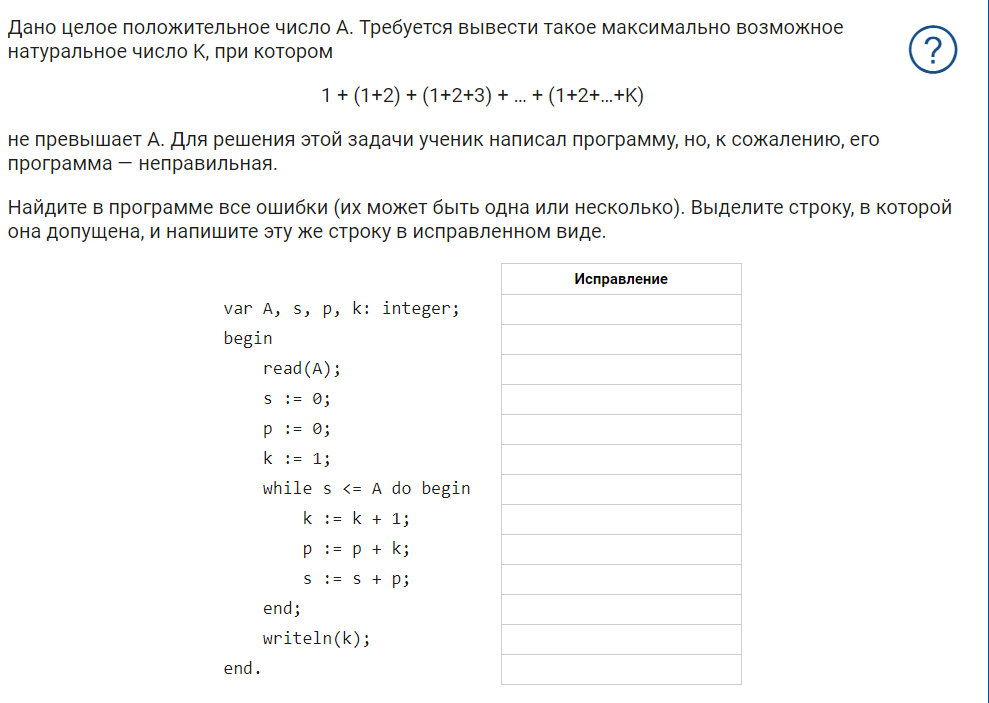
**Задание 11.**



**Задание 12.**



**Задание 13.**



**Практическое занятие № 13. Разработка программ**

Этапы решения задач на компьютере. Структурное программирование. Проверка условия выполнения цикла до начала выполнения тела цикла и после выполнения тела цикла: постусловие и предусловие цикла. Инвариант цикла. Методы проектирования программ «сверху вниз» и «снизу вверх». Разработка программ, использующих подпрограммы. Библиотеки подпрограмм и их использование.

Теоретическая часть

**Основные этапы решения задач на ЭВМ**

Рассмотрим этапы решения задачи на ЭВМ в виде этапов*жизненного цикла* программы как промышленного изделия.

Период существования (разработки и эксплуатации) сравнительно небольшой программы можно разбить на следующие этапы (в скобках указано примерное распределение затрат на разработку между этапами).

1. *Проектирование программы* (17%).
   1. *Постановка задачи* - точная формулировка решаемой программой задачи в виде *технического задания (ТЗ)* на разработку программы.
   2. Выбор или разработка метода решения задачи.
   3. *Алгоритмизация* - проектирование структуры данных и алгоритма программы.

*Программа = Данные + Алгоритм*

Разработка структуры данных проводится параллельно с разработкой алгоритма, несколько опережая ее: на каждом шаге сначала уточняются данные (операнды), а затем - операции над ними.

1. *Программирование* (8%) - перевод алгоритма на язык программирования – программная реализация алгоритма. Программирование можно рассматривать как последний шаг проектирования.

Иногда этот этап называют *кодированием*, а текст программы – (программным) *кодом*. Эти термины сохранились с тех времен, когда программы вручную кодировались в двоичной системе счисления.

1. *Отладка программы* (25%) - выявление, поиск и исправление ошибок. Отладка завершается официальным *испытанием*, *сдачей* и *опытной эксплуатацией* программы. Затем начинается ее *производство* (копирование программы вместе с документацией), постоянная эксплуатация и сопровождение.
2. *Сопровождение программы* (50%) - устранение ошибок и доработка программы (т.е. продолжение разработки) в течение всего периода ее эксплуатации.

На устранение ошибок при отладке и сопровождении уходит 75% затрат разработки. Образно говоря, один день программисты делают ошибки и три дня их исправляют. Борьба с ошибками - главная проблема программирования. Проще предотвращать ошибки, чем потом выискивать их. Особенно важны начальные этапы разработки: чем раньше сделана и позже обнаружена ошибка, тем больше вреда от нее.

Наиболее изучен самый простой этап разработки - программирование (всего 8% затрат). Ему посвящена почти вся литература (большинство книг называется "Программирование на языке ..."). По более сложным вопросам алгоритмизации значительно меньше книг, по отладке очень мало, по сопровождению - почти нет, а именно на эти этапы уходит основная доля затрат труда, времени и средств. В исследовании и совершенствовании этих этапов разработки программ скрыты большие резервы экономии затрат.

Ближайшие разделы учебника посвящены методам и средствам представления и разработки алгоритмов [18, 20, 21, 37].

# Структурное программирование

*Алгоритмизация* - это представление неформального, неточного и неполного описания известного метода решения задачи в виде четкого алгоритма.

Это - не простая проблема. Систематические методы алгоритмизации появились лишь в начале 70-х годов и связаны, прежде всего, с двумя независимыми друг от друга идеями: структурное программирование и разработка сверху вниз.

Эти идеи произвели настоящую революцию в программировании и способствовали его индустриализации. Они лежат в основе современной технологии программирования. В принципе, обе идеи достаточно просты и используются не только в программировании.

*Структурное программирование*  - это метод программирования, в котором используются только алгоритмы, построенные из стандартного набора базовых структур (так называемые *структурные алгоритмы*).

Идея структурного программирования - стандартизация для борьбы с ошибками: ограничить возможную структуру алгоритмов, сделав их более простыми и наглядными (Э. Дейкстра, 1968).

При этом облегчается понимание, разработка, изменение, отладка и верификация алгоритма, уменьшается количество возможных ошибок, упрощается их поиск и, в конечном счете, увеличивается производительность труда программистов и повышается качество программ. В частности, повышаются их надежность и мобильность, упрощается модернизация. При этом, правда, алгоритм может стать более громоздким.

Как и любая стандартизация, структурное программирование рассчитано, прежде всего, на индустриальный подход - коллективную разработку больших и сложных программ в промышленных масштабах. Его роль повышается с ростом размеров и сложности разрабатываемых программ.

*Алгоритм* называется *структурным* (иногда говорят

"структурированным"), если он имеет одну из базовых структур. Каждый блок этих структур сам может иметь внутри любую из этих допустимых структур и т.д. Таким образом из базовых структур можно построить структурный алгоритм любой сложности.

В качестве *базовых структур* обычно используют *последовательность, ветвление и цикл* (рис. 3.1).

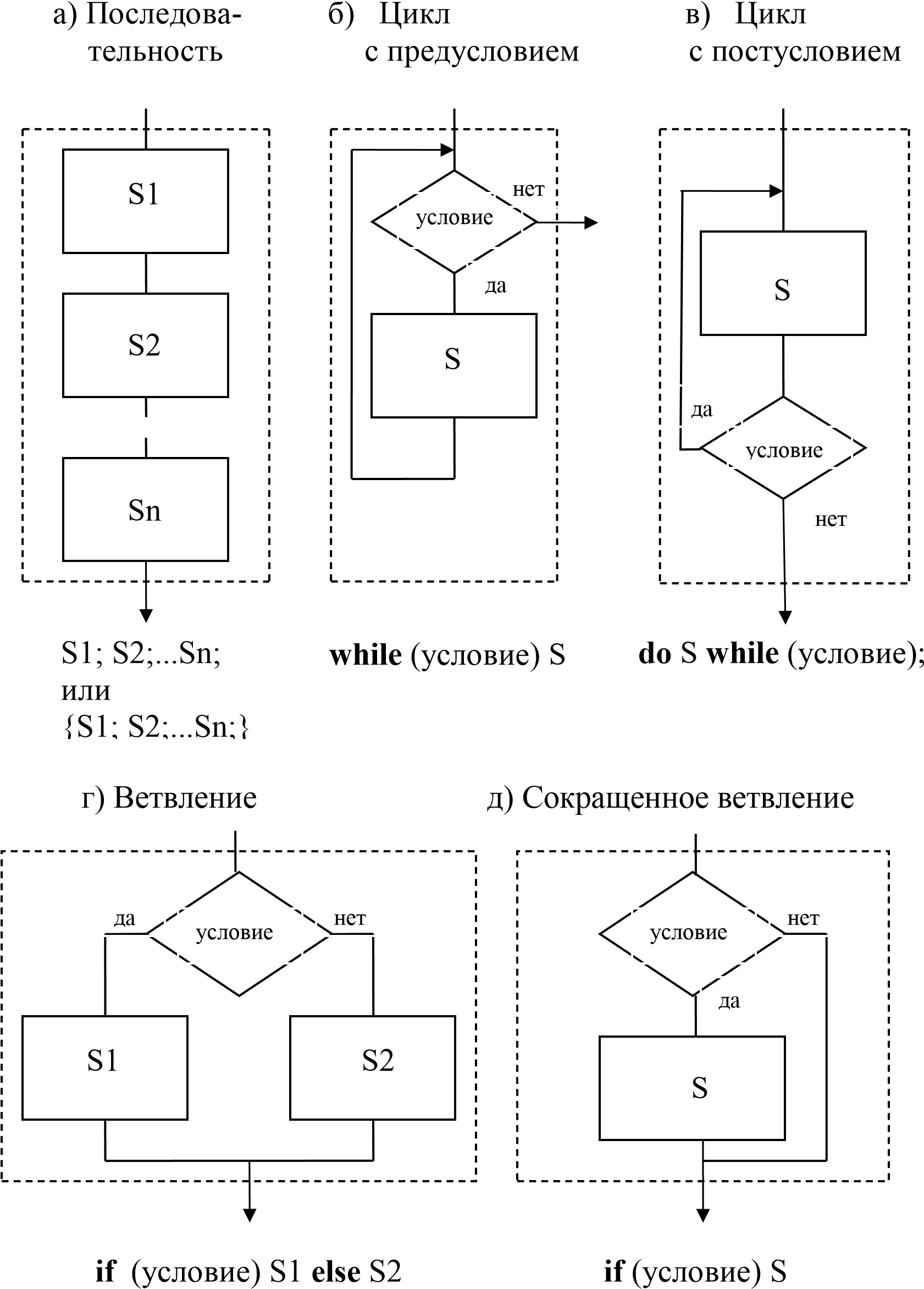


Рис. 3.1. Базовые структуры алгоритма и их запись на языке C

(S - оператор, S1 - оператор, S2 - оператор, Sn - оператор)

## Разработка сверху вниз и снизу вверх

Любая система состоит из частей, каждую из которых также можно разложить на составные части и т. д. Например, университет делится на факультеты, факультеты включают кафедры и курсы, разделенные на лаборатории и группы, которые, в конечном счете, состоят из сотрудников и студентов.

Структуру самых разнообразных систем: технических, организационных, биологических, грамматических и т. п. часто изображают в виде дерева, обычно с корнем наверху (рис. 3.3).



А



В



С



D



Е



F



G



H



I



J



K



L



Сверху вниз



Снизу вверх

# Древовидная (иерархическая) структура алгоритма

Таким же образом можно представить любой алгоритм или программу, разбивая их на более мелкие алгоритмы, блоки и т. д. до команд.

Каждый узел дерева обозначает некоторый алгоритм. Отходящие от него вниз линии ведут к составным частям этого алгоритма. Так, на рис. 3.3 алгоритм A включает в себя алгоритмы B, D, C; алгоритм D, в свою очередь, состоит из K и L, и т. д.

Алгоритм можно разрабатывать, двигаясь по древовидной структуре *сверху вниз*: от целого к деталям, или *снизу вверх*: от частей к целому, т. е. в *нисходящем* или *восходящем* направлении.

Аналогичным образом можно выполнять построение, исследование, изучение, описание и другие действия с любой системой. Например, на лекциях, как в данной книге, материал может излагаться, в основном, сверху вниз: от более крупных понятий к деталям. Параллельно, на практических и лабораторных занятиях, программирование изучается снизу вверх: от примеров простых программ и отдельных деталей языка C к их обобщению и использованию в более крупных и сложных программах.

Разработка сверху вниз начинается от главной цели: на каждом этапе разработки решаемая задача (поставленная цель) разбивается на более простые подзадачи (подцели), с которыми затем поступают таким же образом.

Так, при написании книги или сочинения сначала составляется план:

книга разбивается на части; а затем уже уточняются детали, т. е. пишутся планы и тексты этих частей.

На первом этапе проектируемый алгоритм представляется в виде одного блока. Затем определяется структура этого блока (например, выбирается одна из базовых структур структурного программирования). Таким образом, исходный алгоритм разбивается на части.

Далее разработка продолжается аналогично: каждый блок разбивается на более мелкие действия, пока весь алгоритм не будет разложен на достаточно простые операции, "понятные" процессору (имеющиеся в выбранном языке программирования).

Другие названия нисходящей разработки: *последовательное уточнение, пошаговая детализация* и т. п. Принцип такого подхода можно выразить словами "разделяй и властвуй".

В разделе 1.7 алгоритм разрабатывался снизу вверх: простые операции, из которых состоял алгоритмический процесс (1.2), объединялись в более крупные алгоритмы (блоки, шаги). Из этих блоков затем был составлен структурный алгоритм 1.2.

Восходящий метод менее удобен, чем нисходящий, т. к. трудно заранее предугадать, из каких мелких частей будет состоять искомый алгоритм. Это подобно тому, как, не имея общего плана сочинения, писать его отдельные страницы: многие из них потом могут оказаться ненужными, и большую часть придется переписывать.

Структурное программирование обычно сочетают с проектированием алгоритма сверху вниз. Иногда даже считают разработку сверху вниз частью структурного программирования. Это не совсем правильно: структурное программирование возможно и снизу вверх.

Действительно, базовые структуры можно рассматривать как допустимые в структурном программировании способы и для разделения сложной операции на более простые, и, наоборот, для составления из мелких операций более крупной операции.

Нисходящая и восходящая разработка программы имеют свои достоинства и недостатки и их надо умело сочетать, но в целом предпочтительнее подход сверху вниз, как более целенаправленный.

**Практическое занятие № 14. Разработка программ**

Интегрированная среда разработки программы на выбранном языке программирования. Пользовательский интерфейс интегрированной среды разработки программ. Понятие об объектно-ориентированном программировании. Объекты и классы. *Инкапсуляция, наследование, полиморфизм*. Среды быстрой разработки программ. Графическое проектирование интерфейса пользователя. Использование модулей (компонентов) при разработке программ.

Пример 1.

Возведение вещественного числа в вещественную степень xy

Для примера вычислим значения x3, x4, x5+x6

***program DemoPower;***

***var***

***x, sum: real;***

***function Pow (x,y: real): real;***

*{функция вычисляет x в степени y}*

***begin***

***Pow:=exp(y\*ln(x));***

***end;***

***Begin***

*{начало основной программы}*

***writeln (‘Введите x’);***

***readln (x);***

***writeln (Pow(x,3));****{вычисление и одновременный вывод на экран* x3*}*

***writeln (Pow(x,4));****{вычисление и одновременный вывод на экран* x4*}*

*{Внимание. Далее имя функции используется в выражении как операнд}*

***sum:= Pow (x,5)+Pow(x,6);******));****{вычисление*x5+x6*}*

***writeln (sum);***

***End.***

Пример 2.

Возведение вещественного числа в натуральную степень.

Вычислить 2,7k+(a+1)-5, где *a* и *k*вводятся с клавиатуры.

***Program Kalk;***

***var***

***a, sum: real;***

***k: word;***

***function Step(x: real; n: word): real;***

***var***

***i: word;***

***st: real;***

***begin***

***st:=1;***

***for i:=1 to n do***

***st:=st\*x;***

***step:=st;****{имени функции необходимо присвоить результат}*

***end;***

***Begin***

***writeln (‘Введите K’);***

***readln (k);***

***writeln (‘Введите A’);***

***readln (a);***

*{Внимание. Далее имя функции используется в выражении как операнд}*

***sum:=Step(2.7, k)+1/Step((a+1), 5);***

***writeln (sum:6:2)***

***End.***

Пример 3.

Необходимо в виде подпрограммы оформить вывод элементов одномерного массива на экран.

Рассуждения.

1. Так как подпрограмма будет осуществлять именно набор действий (вывод элементов массива), а не производить вычисление, то оформлять подпрограмму следует в виде процедуры, а не функции.

2. Параметрами процедуры будут имя массива и количество элементов массива. Оба параметра необходимо подать процедуре на вход, то есть сообщить ей имя массива и количество его элементов. В основную программу не надо будет возвращать никаких значений. Все  действия будут выполнены внутри процедуры. Следовательно, параметры процедуры будут параметрами-значениями.

**type**

**mas: array [1..100] of real;**

**.......**

**procedure  Vivod (A:mas, kol: byte);**

{напомним, что здесь необходимо предварительное описание типа Массив}

**begin**

**for i:=1 to kol do**

**write(A[i]:5:2,’ ‘ );**

**end;**

Пример 4.

Необходимо в виде подпрограммы оформить ввод элементов одномерного массива.

Рассуждения.

1. Так как подпрограмма будет осуществлять именно набор действий (ввод элементов массива), а не производить вычисление, то оформлять подпрограмму следует в виде процедуры, а не функции. Кроме того, результатом функции не может быть массив.

2. Параметрами процедуры будут имя массива и количество элементов массива. Количество элементов массива введем в основной программе, поэтому этот параметр необходимо подать на вход (передать) процедуре. Имя массива также необходимо передать процедуре. Однако, если количество элементов известно основной програме (так как ввод этого значения произведется в основном блоке), то значения элементов массива необходимо будет вернуть (передать) в основную программу из процедуры. Следовательно, параметр – количество элементов массива - должен быть параметром-значением, а параметр – массив – параметром-переменной.

**type**

**mas: array [1..100] of real;**

**……..**

**procedure  Vvod (var A:mas; kol: byte);**

{обратим внимание, здесь параметры различных типов в отличие

 от предыдущего примера разделены точкой с запятой}

**begin**

**for i:=1 to kol do**

**begin**

**write (‘Введите ‘, i, ‘  элемент массива’);**

**readln (A[i]:5:2,’ ‘ );**

**end;**

**end;**

**……..**

{Основная программа}

**……..**

**write (‘Введите количество элементов массива’);**

**readln (kol );**

**……..**

Пример 5.

Вычисление факториала числа.

Обоснование выбора способа реализации.

Обратим внимание на то, что   вычислить факториал числа N можно следующим образом:

**N! = N \* (N-1)! = N \* (N-1) \* (N-2)! и так далее**

То есть для вычисления факториала числа N требуется вычислить факториал числа (N-1), для вычисления факториала числа (N-1) необходимо вычислить факториал числа (N-2) и так далее.

Заметим, что вычисление факториала числа сводится к вычислению факториала числа, на единицу меньшего самого числа.

Реализуем такой алгоритм с использованием механизма рекурсии.

Так как подпрограмма будет производить вычисление значения, то реализовывать ее будем в виде функции.

**function Fact (n: byte) : integer;**

**begin**

**if n = 0  then   Fact := 1**

**else   Fact := n \* Fact(n-1);**

**end;**

Здесь имени функции сразу присваивается результат вычисления.

При вызове функции Fact(n-1) согласно оператору Fact := n \* Fact(n-1), где т – параметр функции, вместо n подставится параметр (n-1) и, следовательно, вычислится строка

            n \* (n-1) \* Fact ((n-1) -1) и так далее.

Рекурсивное обращение к функции Fact будет продолжаться до тех пор, пока n не станет равным 0.

**Практическое занятие № 15. Элементы теории алгоритмов**

Формализация понятия алгоритма. Машина Тьюринга – пример абстрактной универсальной вычислительной модели. Тезис Чёрча–Тьюринга. *Другие универсальные вычислительные модели* (*пример:* *машина Поста). Универсальный алгоритм. Вычислимые и невычислимые функции. Проблема остановки и ее неразрешимость.*

*Абстрактные универсальные порождающие модели (пример: грамматики).*

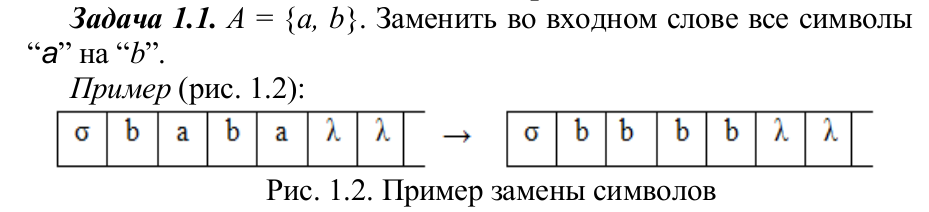
***Цель: Закрепление общего понятия «алгоритм». Решение простейших алгоритмических задач***

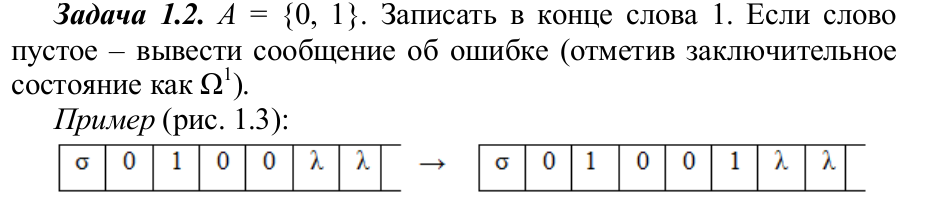
Задание 1. Определить площадь трапеции по введенным значениям оснований (a и b) и высоты (h).

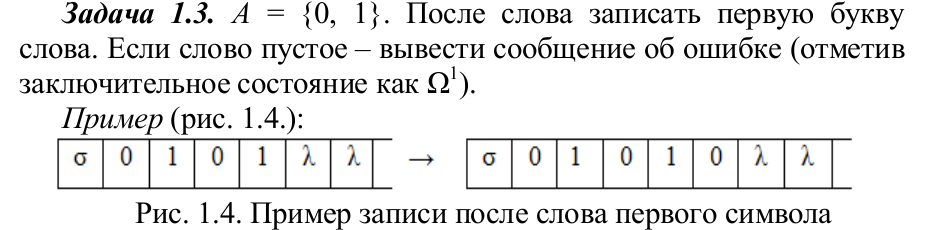
Задание 2. Определить среднее арифметическое двух чисел, если a положительное и частное (a/b) в противном случае.

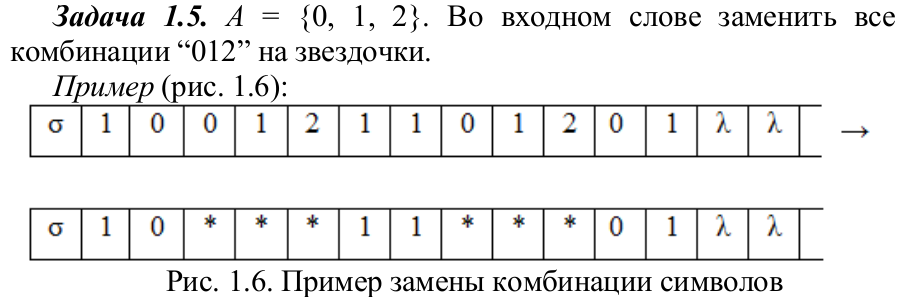
**Задание 3.** Составить алгоритм нахождения суммы целых чисел в диапазоне от 1 до 10.

***Цель: Получение практических навыков в написании программ для «Машины Тьюринга»***

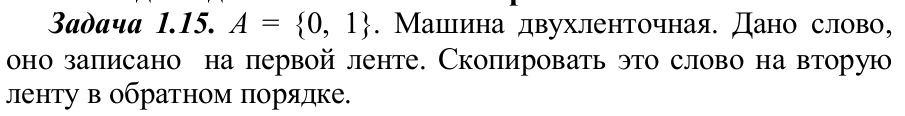


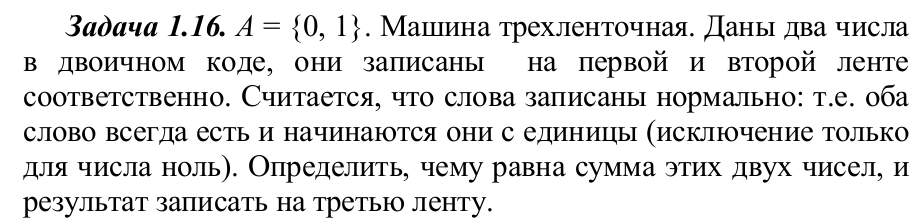


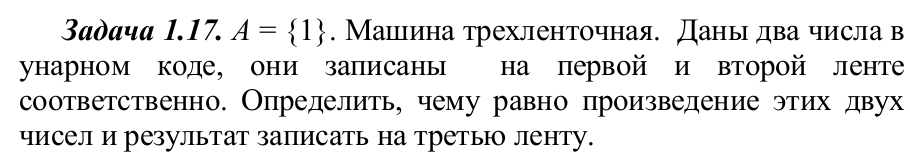




***Цель: Получение практических навыков в написании программ для «Машины Поста»***







**Практическое занятие № 16. Элементы теории алгоритмов**

Сложность вычисления: количество выполненных операций, размер используемой памяти; их зависимость от размера исходных данных. Сложность алгоритма сортировки слиянием (MergeSort). Примеры задач анализа алгоритмов: определение входных данных, при которых алгоритм дает указанный результат; определение результата алгоритма без его полного пошагового выполнения. *Доказательство правильности программ.*

*Цель: Получение практических навыков при решении задач для составления линейных алгоритмов*

* + - 1. Вычислить и вывести на экран значение функции: Y = sin (x+30)/(a + x)+ba.



2. Вычислить медианы треугольника со сторонами *a*, *b*, *c* по формулам:

 ;

 ;

 ;

3. Вычислить площадь круга и длину окружности по введенному значению радиуса.

4. Вычислить площадь *S* и периметр *L* эллипса по введенным значениям полуосей a и b:

 ;

 .

5. Вычислить объем *V* и площадь боковой поверхности цилиндра *S* по введенным значениям радиуса основания *R* и высоты цилиндра *H*.

 ;

 *.*

6. Вычислить объем *V* и площадь боковой поверхности конуса *S* по введенным значениям радиуса основания *r*, высоты *h* и образующей l:

 ;

 .

7. Вычислить объем *V* и площадь поверхности *S* сферы по введенному значению радиуса r:

 ;

 .

**Практическое занятие № 17-18. Математическое моделирование**

Практическая работа с компьютерной моделью по выбранной теме. Проведение вычислительного эксперимента. Анализ достоверности (правдоподобия) результатов компьютерного эксперимента. Представление результатов моделирования в виде, удобном для восприятия человеком. Графическое представление данных (схемы, таблицы, графики). Построение математических моделей для решения практических задач. Имитационное моделирование.

*Моделирование систем массового обслуживания. Использование дискретизации и численных методов в математическом моделировании непрерывных процессов. Использование сред имитационного моделирования (виртуальных лабораторий) для проведения компьютерного эксперимента в учебной деятельности. Компьютерный (виртуальный) и материальный прототипы изделия. Использование учебных систем автоматизированного проектирования.*

Цель:

1. Определение этапов решения задач на компьютере.
2. Закрепление основных понятий: модель, объект, система, формализация.
3. Классификация видов моделей.
4. Определение правил составления модели таблица.

Студент должен знать:

1. Этапы решения задач на компьютере.
2. Определения основных понятий.
3. Виды моделей.
4. Свойства моделей.
5. Правила оформления таблицы.

Студент должен уметь:

1. Выполнять построение вычислительных таблиц.
2. Выполнять построение графов.

Теория

*Моделирование* – это метод познания, состоящий в создании и исследовании моделей.

*Модель* – это некий новый объект, который отражает существенные особенности изучаемого объекта, процесса или явления.

Все модели можно разбить на 2 класса:

1 – *предметные или материальные модели*. Воспроизводят геометрические, физические и другие свойства объектов. Например, глобус, анатомические муляжи, модели кристаллических решеток.

2 – *информационные модели*. Представляют объекты в образной или в знаковой форме. Например, учебные плакаты.

Знаковые информационные модели строятся с использованием различных языков (знаковых систем). Знаковая информационная модель может быть представлена в форме текста (например, программы на языке программирования), формулы или таблицы (например, периодическая таблица элементов Д.И.Менделеева)

Процесс построения информационных моделей с помощью формальных языков называется *формализацией*.

Способы представления информационных моделей.

*Граф* – это средство для наглядного представления состава и структуры системы (например, блок – схемы).

Граф состоит из вершин, связанных дугами или ребрами. Вершины могут быть изображены кругами, овалами, прямоугольниками или точками. Связи между вершинами изображаются линиями. Если линия направленная (т.е. со стрелкой), то она называется дугой, если не направленная (без стрелки), то ребром. Принято считать, что одно ребро заменяет две дуги, направленные в противоположные стороны. Граф, в котором все линии направленные, называется ориентированным графом. Две вершины, соединенные дугой или ребром, называются смежными.

Взвешенный граф – это граф, в котором с вершинами или линиями связана некоторая дополнительная информация. Эта информация называется весом вершины или линии. Чаще всего вес задается в виде надписи на вершине или линии.

Дерево – это граф, предназначенный для отображения таких связей между объектами как вложенность, подчиненность, наследование и т.п.

Таблицы. Структура.

Таблица состоит из следующих частей: головки, боковика и прографки. Кроме того над таблицей могут располагаться табличный номер и заголовок (которые могут и отсутствовать). Прографка состоит из строк и граф. В некоторых случаях головка таблицы и боковик могут оказаться многоуровневыми. Уровни головки называются – ярусами, уровни боковика – ступенями.

Практические задания

1. Построить граф, отображающий структуру мотострелкового батальона СССР.

**Мотострелковый батальон армии СССР**

В середине 70-х гг. мотострелковый батальон Советской Армии на­считывал 395 человек и имел следующую структуру. Во главе стоял командир батальона. Ему подчинялись управление, штаб, 3 мото­стрелковые роты, взвод связи, минометная батарея, противотанко­вый взвод, отделение технического обслуживания, взвод снабжения и батальонный медицинский пункт. В управление батальоном вхо­дили сам комбат, заместитель по политической части, заместитель по технической части и техник батальона. Штаб состоял из началь­ника штаба, начальника связи, инструктора - дозиметриста, писаря и водителя бронетранспортера. Начальник связи являлся командиром взвода связи (еще 12 чел.). Минометная батарея состояла из управ­ления (10 чел.) и двух взводов по 20 чел., в каждом — по 3 120-мм миномета. Противотанковый взвод состоял из отделения станковых противотанковых гранатометов (8 чел., 2 гранатомета СПГ-9) и двух отделений противотанковых управляемых ракет (по 6 чел. и по 2 ПТУ PC в отделении). Отделение технического обслуживания: ко­мандир отделения, водитель - автослесарь и старший механик. Взвод снабжения: командир взвода, его заместитель, хозяйственная часть (3 чел.) и автотранспортное отделение (4 чел.). Батальонный меди­цинский пункт: начальник пункта, шофер-санитар и 2 санитара. Мотострелковая рота состояла из управления (командир роты, за­меститель по политической части, старшина роты), пулеметного от­деления и 3 мотострелковых взводов. Пулеметное отделение состоя­ло из командира отделения, водителя бронетранспортера и двух пулеметных расчетов, в каждом пулеметчик и помощник пулемет­чика. Мотострелковый взвод имел командира взвода, заместителя командира и 3 мотострелковых отделения. В каждом отделении: ко­мандир отделения, пулеметчик, гранатометчик, помощник гранато­метчика, старший автоматчик, 3 автоматчика и водитель бронетранспортера.

1. Построить таблицу по данным текстовой задачи.

**Распределение суши и воды на поверхности Земного шара**

Часть площади Земли в целом, занятая сушей, составляет 29,2%. Часть площади южного полушария, занятая водой, составляет 81,0%. Площадь, которую суша занимает в северном полушарии, составляет 100,5 млн. кв. км. Общая площадь Земли в целом — 510,2 млн. кв. км. Площадь, которую вода занимает в южном полушарии, — 206,6 млн кв. км. Общая площадь северного полушария — 255,1 млн. кв. км. Площадь, которую вода занимает на Земле в целом, составляет 361,2 млн. кв. км. Часть площади северного полушария, занятая водой, составляет 60,6%. Площадь, которую суша занимает в южном полушарии, составляет 48,5 млн. кв. км. Часть площади южного полу­шария, занятая сушей, составляет 19,0%. Общая площадь южного по­лушария — 255,1 млн. кв. км. Площадь, которую вода занимает в се­верном полушарии, составляет 154,6 млн. кв. км. Площадь, которую суша занимает на Земле в целом, составляет 149,0 млн. кв. км. Часть площади северного полушария, занятая сушей, составляет 39,4%. Часть площади земли в целом, занятая водой, составляет 70,8%.

1. Построить таблицу по данным текстовой задачи.

**Территория и население по континентам**

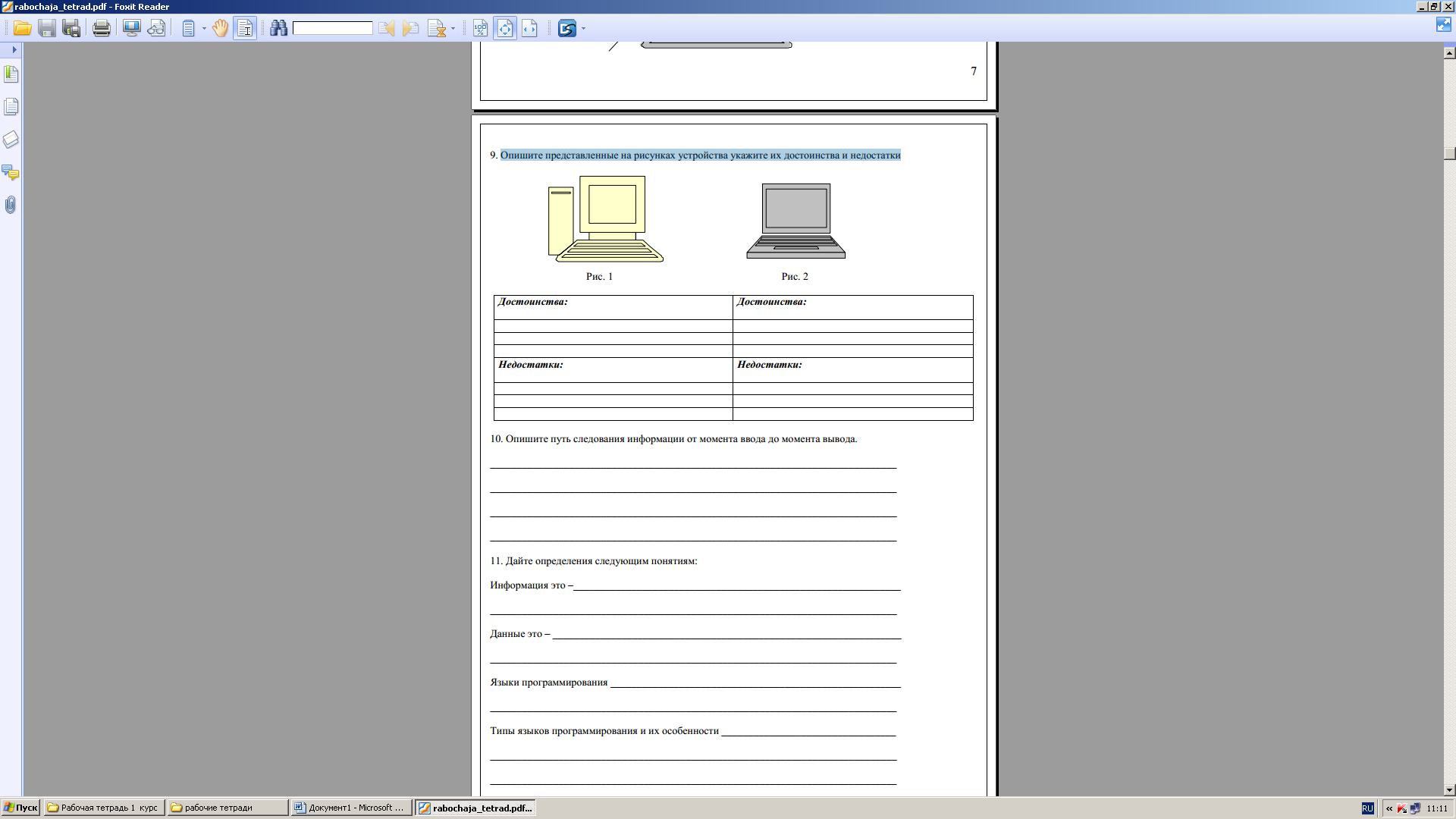
Территория Австралии и Океании — 8,5 млн. кв.км. Плотность на­селения в Африке в 1989 г. была 21 человек на кв.км. Население Ев­ропы в 1989 г. составило 701 млн. человек. Территория Южной Аме­рики — 17,8 млн. кв.км. Население Северной и Центральной Америки в 1989 г. составило 422 млн. человек. Плотность населения в Северной и Центральной Америке в 1970 г. была 13 человек на кв.км. Территория всего мира — 135,8 млн. кв.км. Плотность насе­ления в Австралии и Океании в 1989 г. была 3 человека на кв.км. На­селение Южной Америки в 1989 г. составило 291 млн. человек. Тер­ритория Африки — 30,3 млн. кв.км. Население Австралии и Океании в 1989 г. составило 26 млн. человек. Плотность населения во всем мире в 1970 г. была 27 человек на кв.км. Территория Азии — 44,4 млн. кв.км. Население всего мира в 1989 г. составило 5201 млн. человек. Территория Северной и Центральной Америки — 24,3 млн. кв. км. Население Азии в 1970 г. составило 2161 млн. человек. Плотность населения в Европе в 1989 г. была 67 человек на кв.км. Плотность населения в Азии в 1970 г. была 49 человек на кв.км. На­селение Африки в 1970 г. составило 361 млн. человек. Население Ав­стралии и Океании в 1970 г. составило 19 млн. человек. Население Южной Америки в 1970 г. составило 190 млн. человек. Плотность населения в Африке в 1970 г. была 12 человек на кв.км. Население Северной и Центральной Америки в 1970 г. составило 320 млн. чело­век. Плотность населения в Южной Америке в 1970 г. была 11 чело­век на кв.км. Население Африки в 1989 г. составило 628 млн. чело­век. Плотность населения в Австралии и Океании в 1970 г. была 2 человека на кв.км. Население Европы в 1970 г. составило 642 млн. человек. Плотность населения во всем мире в 1989 г. была 38 чело­век на кв.км. Территория Европы — 10,5 млн. кв.км. Плотность на­селения в Северной и Центральной Америке в 1989 г. была 17 чело­век на кв.км. Плотность населения в Европе в 1970 г. была 61 человек на кв.км. Население Азии в 1989 г. составило 3133 млн. че­ловек. Плотность населения в Южной Америке в 1989 г. была 16 че­ловек на кв.км. Население всего мира в 1970 г. составило 3693 млн. человек. Плотность населения в Азии в 1989 г. была 71 человек на кв.км.

**Практическое занятие № 19. Аппаратное и программное обеспечение компьютера**

Аппаратное обеспечение компьютеров. Персональный компьютер. Многопроцессорные системы. *Суперкомпьютеры*. *Распределенные вычислительные системы и обработка больших данных.* Мобильные цифровые устройства и их роль в коммуникациях. *Встроенные компьютеры. Микроконтроллеры. Роботизированные производства.* Соответствие конфигурации компьютера решаемым задачам. Тенденции развития аппаратного обеспечения компьютеров. Программное обеспечение (ПО) компьютеров и компьютерных систем. Классификация программного обеспечения. Многообразие операционных систем, их функции. Программное обеспечение мобильных устройств.

|  |  |
| --- | --- |
| 1. Основные типы ЭВМ – |  |
|  | |
|  | |

2. Опишите представленные на рисунках устройства укажите их достоинства и недостатки



|  |  |
| --- | --- |
| Достоинства: | Достоинства: |
|  |  |
|  |  |
|  |  |
|  |  |
| Недостатки: | Недостатки: |
|  |  |
|  |  |
|  |  |
|  |  |

3. Подпишите группы клавиш на клавиатуре



4. Найди 15 знакомых названий клавиш. Слова расположены вертикально или горизонтально.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| E | S | C | K | C | O | L | M | U | N |
| B | A | C | K | S | P | A | C | E | D |
| S | K | C | O | L | S | P | A | C | E |
| H | C | H | O | M | E | A | N | T | L |
| I | T | A | Y | E | N | D | U | A | E |
| F | R | L | E | N | T | E | R | B | T |
| T | L | T | I | F | D | B | F | H | E |
| Q | P | A | G | E | D | O | W | N | H |
| S | C | R | O | L | L | L | O | C | K |
| A | W | W | J | P | U | E | G | A | P |

1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

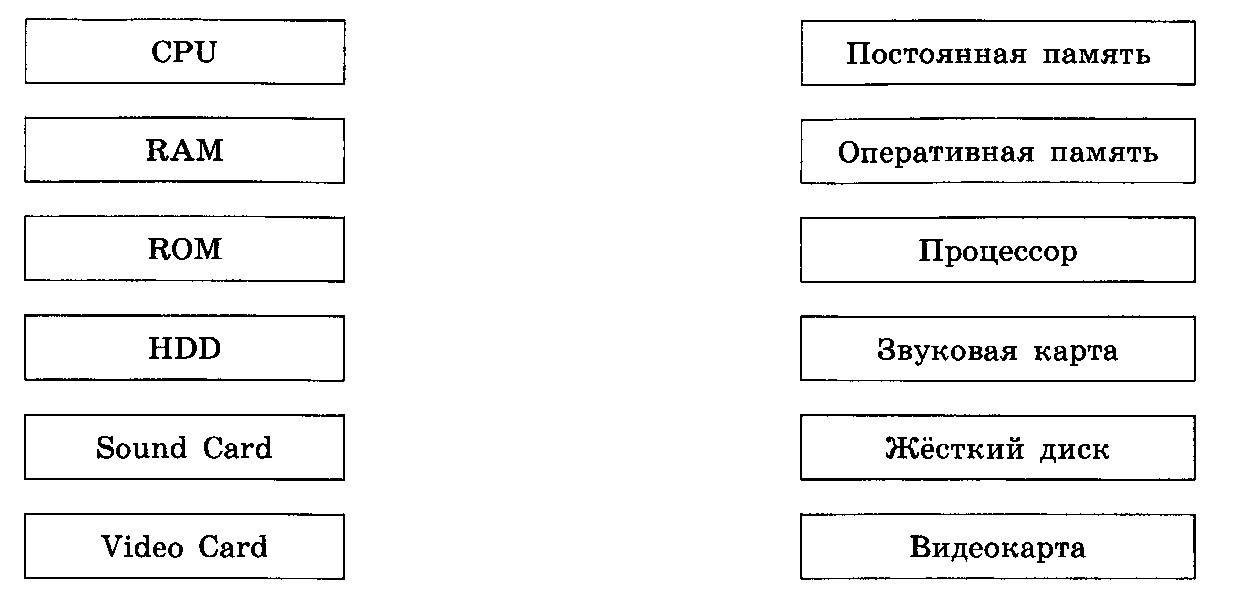
4. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 5. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 6. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

7. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 8. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 9. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

10. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 11. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 12. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

13. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 14. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 15. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Установите соответствие



6. Из перечня устройств выберите (отметьте галочкой) те, которые находятся в системном блоке

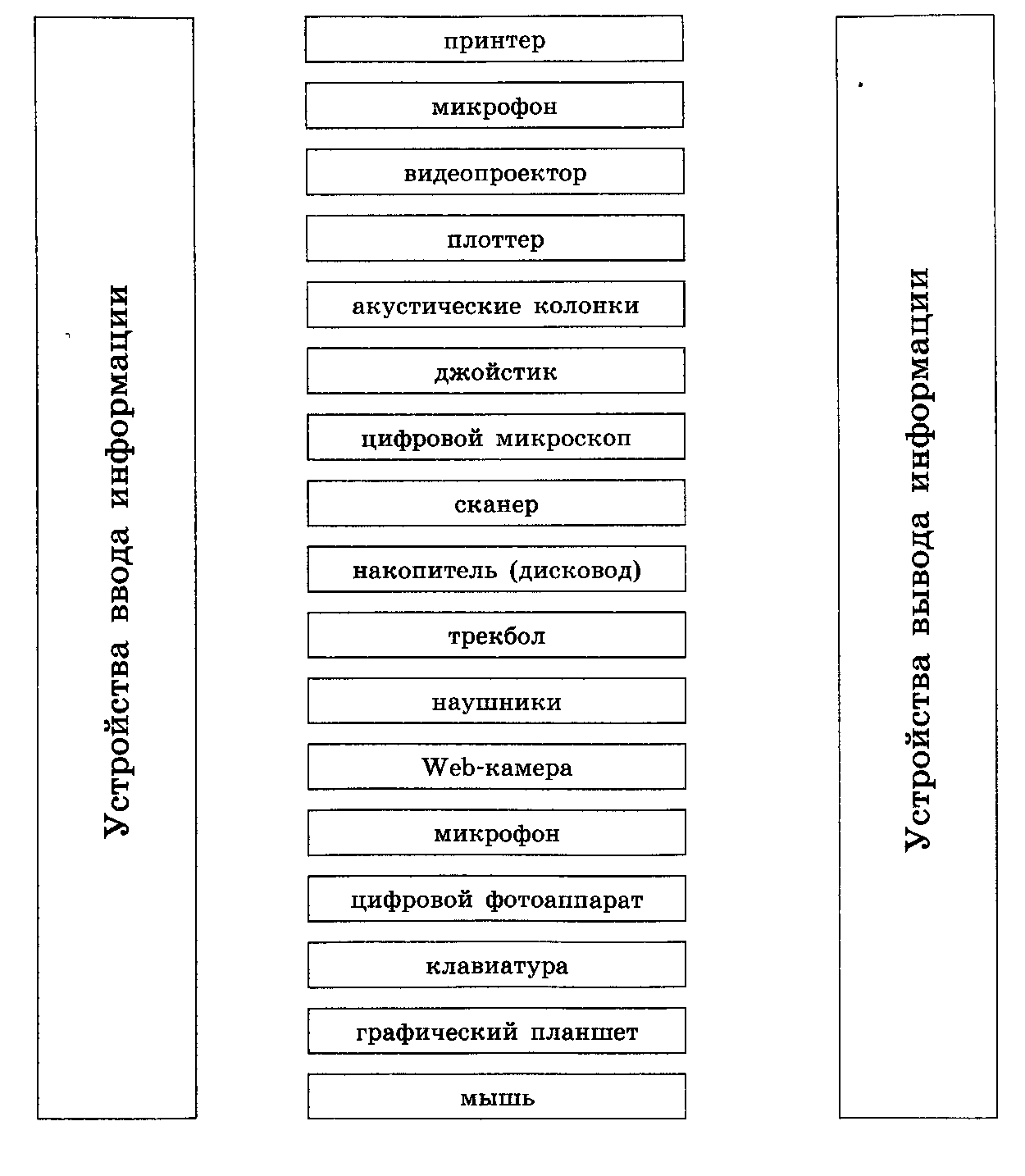
|  |  |  |  |
| --- | --- | --- | --- |
| 🞎 | Процессор | 🞎 | Блок питания |
| 🞎 | Сетевая карта | 🞎 | Сканер |
| 🞎 | Flash-память | 🞎 | Накопитель (дисковод) |
| 🞎 | Оперативная память | 🞎 | Трекбол |
| 🞎 | Материнская плата | 🞎 | Источник бесперебойного питания |
| 🞎 | Плоттер | 🞎 | Web-камера |
| 🞎 | Видеокарта | 🞎 | ПЗУ |

1. Установите соответствие

|  |  |
| --- | --- |
| 1. Принтеры | 1. Планшетные |
| 1. Лазерные |
| 1. Барабанные |
| 1. Мониторы | 1. Жидкокристаллические |
| 1. Струйные |
| 1. Матричные |
| 1. Сканеры | 1. С электронно-лучевой трубкой |
| 1. Ручные |

Ответ: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

8. Установи соответствие.



9. В сети Интернет найдите информацию о современных информационных носителях и заполните таблицу

|  |  |
| --- | --- |
| **Информационный носитель** | **Информационная емкость** |
| Жесткий диск |  |
| CD |  |
| DVD |  |
| Flash-память |  |
| Blue-ray |  |

10. Заполните таблицу

|  |  |  |
| --- | --- | --- |
| **Характеристики** | **CD** | **DVD** |
| *Сходство* |  | |
|  | |
|  | |
| *Различие* |  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

11. Соедини линиями название устройств и название выполняемого им действия.

Выводит информацию

Хранит информацию

Джойстик

Мышь

Сканер

Принтер

Микрофон

Световое перо

Колонки

Монитор

Графопостроитель

Вводит информацию

Обрабатывает информацию

*Примечание. Одно устройство может выполнять несколько действий.*

12. Заполните таблицу, используя ресурсы Интернет

|  |  |  |
| --- | --- | --- |
| **Характеристики современных ПК** | | |
| *Устройство* | *Наименование параметра* | *Характеристика* |
| Процессор | Модель |  |
| Тактовая частота |  |
| Количество ядер |  |
| Оперативная память | Тип модуля |  |
| Объем |  |
| Монитор | Тип |  |
| Разрешающая способность |  |
| Количество отображаемых оттенков |  |
| Кэш-память | Вид |  |
| Объем |  |
| Принтер | Тип |  |
| Скорость печати |  |
| Внешний накопитель на МД | Емкость |  |
| Накопитель на компакт-дисках | Тип |  |
| Емкость |  |
| Флеш-накопитель | Тип |  |
| Емкость |  |

13. В рекламных материалах часто приводятся характеристики компьютеров в виде «формул устройств». На основе приведенных ниже «формул» для ПК расшифруйте характеристики этих машин.

|  |  |  |  |
| --- | --- | --- | --- |
| **Модель ПК** | **«Формула устройства»** | **Характеристики** | |
| Ноутбук Fujitsu-Siemens AMILO Pro V2055 | Intel Celeron M420 1.6ГГц/ 256Мб/ 40Гб/ DVD+CDRV/ WLAN/ Linux | Процессор |  |
| ОЗУ |  |
| Жесткий диск |  |
| Привод |  |
| Сеть |  |
| ОС |  |
| Ноутбук MSI M670-033 | AMD TIRIOD 1.6ГГц/ 1024Мб/ 100Гб/ 15,4 “WXGA/ DVD-RW/ модем 802.11 Гб/ Windows Vista |  | AMD TIRIOD 1.6ГГц |
|  | 1024 Мб |
|  | 100Гб |
|  | 15,4 “WXGA |
|  | DVD-RW |
|  | модем 802.11 Гб |
|  | Windows Vista |

|  |  |  |  |
| --- | --- | --- | --- |
| **Модель ПК** | **«Формула устройства»** | **Характеристики** | |
| Ноутбук Acer Aspire 3693WLMi | Intel Celeron M410 1.7ГГц/ 512Мб/ 80Гб/ DVD-RW/ модем 802.11 Гб | ОЗУ |  |
| Жесткий диск |  |
| Привод |  |
| Сеть |  |

14. Поставьте соответствие

|  |  |
| --- | --- |
| 1. Поколение 1 | 1. Транзисторы |
| 1. Поколение 2 | 1. Микропроцессор |
| 1. Поколение 3 | 1. Электронные лампы |
| 1. Поколение 4 | 1. Интегральные схемы |

Ответ: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

15. Разгадай кроссворд.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  | **1** |  |  |  |  |  |
|  |  |  |  |  | **2** |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | **1** |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | **3** |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  | **4** |  |  |
|  | **2** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | **5** |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | **6** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **3** |  |  |  |  |  |  |  | **7** |  |  |  |  |  |  |  |
|  |  |  |  |  |  | **4** |  |  |  |  |  |  |  |  |  |
|  |  | **5** |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | **6** |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

*По вертикали:*

1. Кого считают отцом современного ПК?
2. Машина, которую сконструировал Лейбниц.
3. Электромеханическое … - переключатель с двумя позициями: включено – выключено.
4. В 1949 году была построена ЭВМ с архитектурой этого математика.
5. Большая электронно-счетная машина.
6. Академик, под чьим руководством создан первая ЭВМ в Советском Союзе.
7. Первый электронный компьютер, созданный в США в 1946 году.

*По горизонтали:*

1. Древние счеты.
2. Холлерит создал счетно-… машину.
3. Первый в мире программист.
4. Первая ЭВМ, созданная в Советском Союзе.
5. Вычислительная машина с программным управлением на релейных и механических элементах.
6. Математик, сконструировавший первую в мире механическую счетную машину.

*Модель информационной системы «клиент–сервер». Распределенные модели построения информационных систем. Использование облачных технологий обработки данных в крупных информационных системах.* Инсталляция и деинсталляция программного обеспечения. *Системное администрирование.* Тенденции развития компьютеров. *Квантовые вычисления.* Техника безопасности и правила работы на компьютере. Гигиена, эргономика, ресурсосбережение, технологические требования при эксплуатации компьютерного рабочего места. *Проектирование автоматизированного рабочего места в соответствии с целями его использования.* *Применение специализированных программ для обеспечения стабильной работы средств ИКТ. Технология проведения профилактических работ над средствами ИКТ: диагностика неисправностей.*

***Цель:****научиться устанавливать программное обеспечение и использовать по назначению*

**Теоретическая часть**

**Программное обеспечение (ПО)**— это совокупность программ, обеспечивающих функционирование вычислительных средств и их применение по назначению. По функциональному признаку ПО делится на системное и прикладное.

**Системное программное обеспечение** (СПО) используется, в первую очередь, для управления всеми ресурсами ЭВМ, выполнения и разработки программных продуктов, а также для предоставления пользователям определенных услуг. Оно является необходимым дополнением к техническим средствам ЭВМ и без него машина фактически безжизненна.

**Прикладное программное обеспечение** (ППО) предназначено для создания программных продуктов в любой проблемной области, включая СПО.

**Инструментальное ПО** - совокупность программ, которые используют при разработке программ прикладного и системного ПО.

Системное программное обеспечение.

СПО включает в себя операционные системы (ОС), сетевое ПО, средства расширения функций ОС, средства тестирования и диагностики ЭВМ, а также средства разработки программ (трансляторы, редакторы связей, отладчики и пр.).

Программное обеспечение – это то, что «оживляет» компьютер, который без программ был бы нагромождением ненужной и очень дорогой электроники. Программы обычно являются последовательным набором команд, сообщающих компьютеру, что ему нужно делать в той или иной ситуации.

Классификация программ. Все программы можно разделить на несколько классов, каждый из которых занимает определенную ступень в иерархической лестнице программного обеспечения.

**Встроенная система ввода-вывода (BIOS).** Встроенная система ввода-вывода хранится в постоянной памяти компьютера. Она раскладывает на примитивные операции все команды, относящиеся к вводу или выводу данных из компьютера. Так, дисковод понимает только команды типа поместить голову дисковода на такую-то дорожку, считать информацию из сектора и т.п. И если каждая программа будет содержать в себе команды такого уровня, то она будет неэффективно работать и занимать много места. Помимо этого BIOS осуществляет начальную (при включении питания) загрузку операционной системы с дисковода или жесткого диска.

**Операционная система.**Операционная систем загружается при каждом включении компьютера. Она является как бы прослойкой между базовой системой ввода-вывода и всеми остальными программами. Ее основная задача – распределение ресурсов компьютера, запуск прикладных программ и обработка запросов этих программ на выполнении операций ввода-вывода. Операционная система осуществляет также диалог между пользователем и компьютером: позволяет работать с данными и программами.

**Оболочки систем.** Следующим классом программ являются оболочки систем. Их задача – максимально упростить диалог пользователя с компьютером; в основе их обычно лежит принцип «что вижу то и делаю». Кроме того, она предоставляет множество дополнительных сервисных функций. В результате работа с системой строится по принципу «сиди и нажимай», что, несомненно, очень удобно. В последнее время прослеживается тенденция включения функций оболочки непосредственно в операционную систему.

**Прикладные программы.**Самый распространенный класс программных продуктов, представляющий наибольший интерес для пользователя. Прикладные программы призваны решать самые разные задачи: редактирование текста, создание различных рисунков, работа с таблицами и многие другие. Все эти программы пишутся по принципу максимального удобства для пользователя, обладают дружественным интерфейсом.

Встроенные, нерезидентные и резидентные программы. Программные продукты разделяются не только на классы, но и по способу работы.

**Встроенные программы** постоянно хранятся внутри компьютера, они всегда готовы к работе, их не надо загружать. Основное назначение встроенных программ – обеспечивать связь между аппаратной частью компьютера и другими работающими в этот момент программами. Типичнейший представитель этого семейства базовая система ввода-вывода (BIOS).

**Коммерческие и некоммерческие** программные продукты. Подавляющее большинство программных продуктов являются коммерческими: прежде чем их использовать, вы должны заплатить. Покупая программу, вы не становитесь ее собственником; она передается вам в пользование в соответствии с лицензионным соглашением. Вы не имеете права копировать дистрибутив, передавать его другим людям и устанавливать программу на несколько компьютеров.

Некоммерческие программы распространяются совершенно свободно, предоставляя вам возможность делать с ними все что угодно. Однако чаще всего качество таких программ оставляет желать лучше.

**Программы прикладного ПО** для решения определенных целевых задач из проблемных областей часто называют приложениями.

* Из всего разнообразия прикладного ПО можно выделить группу наиболее распространенных программ (типовые пакеты и программы), которые используются во многих областях человеческой деятельности.

К типовому ППО относят следующие программы:

* текстовые процессоры;
* табличные процессоры;
* системы иллюстративной и деловой графики (графические процессоры);
* системы управления базами данных;
* экспертные системы;
* программы математических расчетов, моделирования и анализа экспериментальных данных.

Предлагаемые на рынке ПО приложения, в общем случае, могут быть выполнены как отдельные программы, либо как интегрированные системы. Интегрированными системами обычно являются экспертные системы, программы математических расчетов, моделирования и анализа экспериментальных данных, а также офисные системы. Примером мощной и широко распространенной интегрированной системы является офисная система Microsoft Office.

**Практическая часть**

**Порядок выполнения работы:**

Повторить требования по соблюдению техники безопасности.

ВключениеПК должно производиться в следующей последовательности:

* включить монитор;
* включить системный блок.

Перед выключением компьютера завершите все работающие программы и подождите 1-2 сек. (это необходимо, если на ва­шем ПК предусмотрено кэширование дисков). Далее необходимо:

• выключить системный блок;

• выключить монитор.

1. Ознакомиться с пунктами практической работы, название, цель работы, задание данной практической работы. Сделайте вывод о проделанной работе.

**Задание 1.**

1. Установить на ПК программу **Wise Calculator** (калькулятор для инженерных и научных вычислений с очень большими возможностями)

Сайт программы: [http://www.wisecalculator.chat.ru/ras.html](https://infourok.ru/go.html?href=http%3A%2F%2Fwww.wisecalculator.chat.ru%2Fras.html)

2. Опишите этапы процесса установки программы

**Задание 2.**

1. Установить на ПК программу **Stamina**(программа-тренажер для тех, кто хочет научиться набирать текст на клавиатуре всеми десятью пальцами)

Сайт программы: [http://stamina.ru](https://infourok.ru/go.html?href=http%3A%2F%2Fstamina.ru%2F)

2. Опишите этапы процесса установки программы

**Задание 3.**

1.Установить на ПК программу **CCleaner** (утилита для чистки мусора в операционной системе)

Сайт программы: [http://www.ccleaner.com/](https://infourok.ru/go.html?href=http%3A%2F%2Fwww.ccleaner.com%2F)

2. Опишите этапы процесса установки программы

3. Удалите программу CCleaner

4. Опишите этапы процесса удаления программы

**Задание 4.**

1. Установить на ПК программу **Кто хочет стать отличником** (игровая система тестирования)

Сайт программы: [http://otlichnik5.net/](https://infourok.ru/go.html?href=http%3A%2F%2Fotlichnik5.net%2F)

2. Опишите этапы процесса установки программы

3. Удалите программу CCleaner

4. Опишите этапы процесса удаления программы

**Контрольные вопросы (ответы в тетради)**

1. Что такое дистрибутив?
2. На какие группы программ делится программное обеспечение ПК?
3. Перечислите программы относящиеся к системным.
4. Какие программы называют прикладными?
5. Что такое утилиты?
6. Для чего предназначены драйвера?

**Практическое занятие № 20-21. Подготовка текстов и демонстрационных материалов**

Технологии создания текстовых документов.

Цель: изучить интерфейс текстового процессора MS Word, правила редактирования документов.

Студент должен знать

интерфейс текстового процессора MS Word

Студент должен уметь

использовать на практике правила редактирования документов

Теоретическая часть

Основные возможности Текстового редактора MicrosoftWord

MicrosoftWord - текстовый редактор, предназначенный для выполнения всех процессов обработки текста: набора и верстки, проверки орфографии, вставки в текст графики, печати текста. В документах наряду с текстом могут встречаться рисунки, таблицы, формулы.

К основным возможностям программы относятся следующие операции:

* набор и редактирование текста;
* исправление орфографических и грамматических ошибок;
* оформление внешнего вида документа;
* создание таблиц, графиков и рисунков;
* оформление шаблонов деловых писем, визитных карточек и других документов;
* расчёт простейших формул в таблицах;
* слияние документов;
* защита документа паролем;
* вывод документа на печать;
* подготовка pdf-документа;
* совместная работа с документом и т.п.

Документы MicrosoftWord имеют расширения docx.

 Набор и корректировка текста

Принято различать в документе MicrosoftWord такие структурные элементы:

1. *символ* - это буква, цифра, знаки препинания, специальные знаки (типа табуляции);
2. *абзац* - это набор символов между двумя признаками конца абзацев;
3. *раздел* - структурная единица документа, имеющая определённые параметры страницы.

Клавиши навигации по тексту

В процессе чтения и корректировки текста необходимо свободно перемещаться по тексту, используя соответствующие клавиши.

*Стрелка влево / стрелка вправо* - переход по символам.

*Ctrl+стрелка влево / стрелка вправо* - переход по словам.

*Home / End* - переход в начало / конец строки.

*Ctrl+Home / End* - переход в начало / конец документа.

*Стрелка вверх / вниз* - переход по строкам.

*Ctrl+стрелка вверх / стрелка вниз* - переход к началу предыдущего / следующего абзаца.

*PgUp / PgDn* - переход по экранным страницам.

*Ctrl+PgUp / PgDn* - переход по реальным печатным страницам документа.

*Alt+стрелка вверх / стрелка вниз -* переход по предложениям.

*F5 или Ctrl+G* - перейти на страницу по номеру, введя номер страницы, нажав Enter, затем нажав клавишу Esc.

*Shift+F5* - при открытии существующего документа вернуться к месту последних сделанных изменений

Работа с фрагментами текста

С фрагментами текста можно выполнять следующие операции:

1. копировать и перемещать, как в рамках одного документа, так и переносить их в другие документы;
2. удалять;
3. заменять вновь набираемым текстом;
4. изменять параметры форматирования.

Для совершения операций с фрагментом текста необходимо вначале его выделить. Способы выделения текста:

*Shift+стрелка вправо/ влево* - текущий или предыдущий символ;

*Shift+Ctrl+стрелка вправо / влево* - текущее или предыдущее слова;

*Shift+стрелка вниз /вверх* - следующую или предыдущую строку (предварительно поместив курсор в начало / конец строки);

*Shift+Ctrl+стрелка вниз / вверх* - текущий или предыдущий абзац;

*Shift+PgDn / PgUp* - текущую или предыдущую экранную страницу;

*Shift+Home / End* - фрагмент текста от позиции курсора до начала или конца текущей строки;

*Shift+Ctrl+Home / End* - фрагмент текста от позиции курсора до начала или конца документа;

*Ctrl+A* - весь текст документа.

Практические задания

1. *Создайте новый документ и именем* ***Редактирование*** *в своей сетевой паке* ***WORD****. Введите текст.*

 Рабочее окно MicrosoftWord.

Окно программы состоит из строки заголовка, ленты вкладок, рабочего поля и строки состояния. В строке заголовка находятся название файла документа, имя программы, стандартные значки "Свернуть", "Развернуть", "Восстановить", "Закрыть". При запуске программы создаётся новый пустой документ с названием Документ 1, расположенный в рабочем поле.

Под строкой заголовка находится лента, состоящая из нескольких вкладок. С помощью опций на ленте можно выполнять различные операции с текстом: изменять шрифт, размер, цвет, создавать таблицы, вставлять рисунки и многое другое. Для перехода на ленту нажмите клавишу Alt или F10. Под лентой вкладок находится рабочее поле, в котором осуществляется набор и форматирование текста. При открытии документа курсор находится в первой позиции первой строки.

Ниже рабочего поля располагается строка состояния, в которой отображается информация о документе: количество страниц и номер текущей страницы, язык ввода текста, статистика, состояние проверки правописания, кнопки режимов отображения документа.

1. *На следующей странице ранее созданного документа введите текст:*

[1] По умолчанию установлен режим просмотра документа "Разметка страницы".

[2] Web - документ - применяется для публикации документа в виде Web-страницы.

[3] Существуют следующие режимы просмотра документов:

[4] Режим чтения - используется для чтения документа.

[5] Черновик - используется для быстрого редактирования документа.

[6] Структура - документ отображается в виде структуры, содержащей главы, параграфы, пункты.

[7] Разметка страницы - видно, как документ выглядит на печатной странице с колонтитулами и номерами страниц.

*Разместите пронумерованные фрагменты текста в следующем порядке: первый абзац – [3]; второй – [7],третий -[2], четвертый - [4], пятый - [6], шестой -[5], седьмой -[1]*

Контрольные вопросы

1. Что такое зона выделения?
2. Каковы наиболее часто используемые способы выделения текста?
3. Как выполняется форматирование абзаца? Каковы его основные возможности?

**Теоретические сведения**

Для совершения операций с фрагментом текста необходимо вначале его выделить. Способы выделения текста:

*Shift+стрелка вправо/ влево* - текущий или предыдущий символ;

*Shift+Ctrl+стрелка вправо / влево* - текущее или предыдущее слова;

*Shift+стрелка вниз /вверх* - следующую или предыдущую строку (предварительно поместив курсор в начало / конец строки);

*Shift+Ctrl+стрелка вниз / вверх* - текущий или предыдущий абзац;

*Shift+PgDn / PgUp* - текущую или предыдущую экранную страницу;

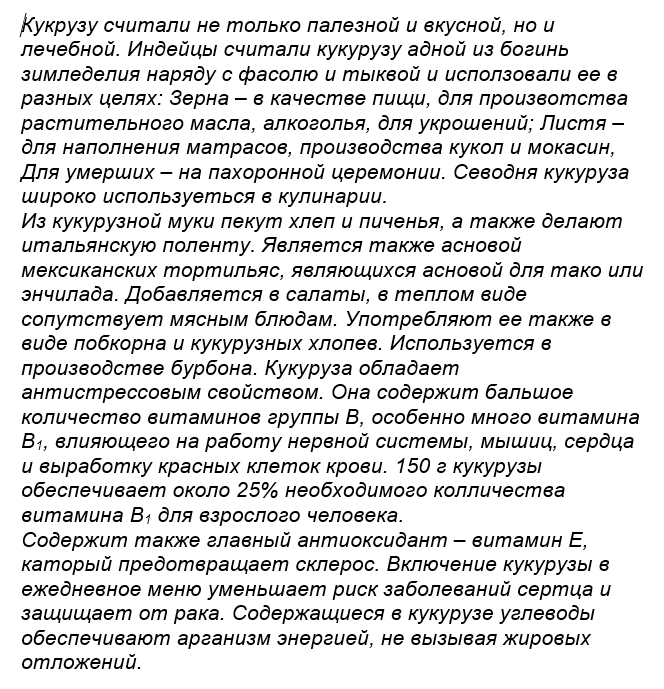
*Shift+Home / End* - фрагмент текста от позиции курсора до начала или конца текущей строки;

*Shift+Ctrl+Home / End* - фрагмент текста от позиции курсора до начала или конца документа;

*Ctrl+A* - весь текст документа.

**Практические задания**

Откройте файл Кукуруза. Отредактируйте текст.



*Задание 1. Наберите текст, примените видоизменения по образцу*

ВИДОИЗМЕНЕНИЕ

~~Зачеркнутый~~

~~Двойное зачеркивание~~

Надстрочный

Подстрочный

Малые прописные

ВСЕ ПРОПИСНЫЕ

*Задание 2. Наберите текст, примените соответствующий вид регистра*

РЕГИСТР

как в предложениях

Все Строчные

все прописные

начинать с прописных

изменить регистр

*Задание3. Наберите текст, примените соответствующий вид текстового эффекта*

ТЕКСТОВЫЕ ЭФФЕКТЫ

Сплошная заливка текста красным цветом

Контур текста сплошная желтая линия

Тень – заготовки - снаружи

Тень – заготовки - внутри

Перспектива

Отражение – заготовки - Полное отражение - касание

Свечение и сглаживание – заготовки

*Задание 4. Наберите текст. Для заголовка используйте шрифт Cambria, 15,* ***Ж****. Для основного текста используйте шрифт Tahoma, 14, К. Подчеркните названия атрибутов символов двойной синей линией.*

Атрибуты формата отдельных символов:

шрифт — общий дизайн символов (вид шрифта);

размер — высота символов, измеряемая в пунктах (1 пункт = 1/72 дюйма);

начертание — внешний вид (обычный, полужирный, курсив и т. д.);

подчеркивание символа — одинарное, двойное, пунктирное или только слова (пробелы не подчеркиваются);

эффекты — выделение символов: зачеркивание, верхний индекс, нижний индекс, скрытый, малые прописные и все прописные;

цвет — цвет символов на экране монитора или на цветном принтере;

интервал — расстояние добавляемое или отнимаемое от межсимвольного интервала для получения растянутого или сжатого текста;

смещение — расстояние, на которое символы поднимаются или опускаются, при этом размер символов не уменьшается;

кернинг — символы в некоторых парах (например, А и W) сдвигаются ближе. Word автоматически регулирует кернинг пар символов, размер которых равен или превышает значение, указанное в поле «Пунктов и более».

Контрольные вопросы:

1. Какие объекты являются основными объектами форматирования?
2. Как выполнить форматирование шрифта?
3. Какие параметры шрифта можно задавать при форматировании?
4. Что называется гарнитурой шрифта?
5. В каких единицах задается размер шрифта?
6. Что такое кернинг?

**Практическое занятие № 22. Подготовка текстов и демонстрационных материалов**

Вставка графических объектов, таблиц. Средства создания и редактирования математических текстов.

Цель: получить представление о работе с таблицами в MS Word, изучить команды пункта меню Таблица, назначение кнопок панели инструментов таблицы и границы

Теоретические сведения

Таблица Word состоит из строк и столбцов ячеек. Таблицы могут содержать цифры, текст и рисунки. Таблицы используются для упорядочения и представления данных.

Для создания новой таблицы Word используется панель инструментов **Таблицы** вкладки **Вставка**. Существует три способа создания таблиц:

1. Нарисовать
2. Вставить
3. Создание на основе существующих данных (текста, чисел)

Практические задания

*Задание 1. Нарисуйте таблицу. Залейте каждую строчку разным цветом.*

|  |  |  |
| --- | --- | --- |
|  | |  |
|  |  |  |
|  |  |  |

*Задание 2. Вставьте таблицу. Оформите внешние границы двойной линией , шириной 1,5 пт.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |

*Задание 3. Вставьте таблицу состоящую из 4 столбцов и 6 строк. Примените любой стиль таблицы.*

**Практические задания**

1. Наберите объявление

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| * **разговорный язык и письмо**   Английский язык   * **индивидуально и в группах** * **опыт работы** * **апробированная методика**   🕿123-45-67 | | | | | | | |
| ***123-45-67 англ.язык*** | ***123-45-67 англ.язык*** | ***123-45-67 англ.язык*** | ***123-45-67 англ.язык*** | ***123-45-67 англ.язык*** | ***123-45-67 англ.язык*** | ***123-45-67 англ.язык*** | ***123-45-67 англ.язык*** |

2. Наберите формулировку теоремы и выполните к ней чертеж в соответствии с предложенным образцом

Теорема 16.1

*Пересекающиеся прямые, соответственно параллельные перпендикулярным прямым, сами перпендикулярны.*

А

С

В

α

А1

В1

С1

α1

*Работа с формулами. Вставка и редактирование формул.*

**Теория**

Вставка формул.

Вставка формул в редакторе WORD осуществляется с помощью формульного редактора.

Вызов формульного редактора Equation Editor из Word можно осуществить следующей последовательностью действий:

* поместите курсор в то место, где должна быть вставлена формула;
* в меню "вставка" выберите команду "обьект";
* выберите закладку "создание";
* В окне "тип обьекта" выберите "Microsoft Equation 3.0 (2.0)";

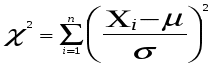
Более удобный вызов редактора математических формул выполняется при помощи кнопки редактора формул, которая помещается на панель инструментов. Размещение кнопки на панели выполняется следующим образом: через меню Вид/Панели инструментов/Настройка на вкладке Команды отыскивается в списке категорий пункт Вставка, в котором выбирается кнопка Редактор формул и перетаскивается на любую панель инструментов.

В результате вызова Редактора формул на экране появляется панель инструментов, состоящая из двух рядов пиктограмм:

В первом ряду расположено 10 пиктограмм, за каждой из которых находится группа символов (математические операции, греческие символы и т. д.) Во втором ряду находятся пиктограммы для вызова шаблонов наиболее распространенных структурных формул (матрицы, суммы и т. д.). Кроме того главное меню Word заменяется на меню редактора математических формул.

*Задание 1.*

С помощью формульного редактора Equation Editor наберите формулу:

  
  
*Методические указания по выполнению задания 1.*

Вызовите формульный редактор;

В палитре шаблонов выберите третий слева шаблон с индексами;

В открывшемся списке выберите левый в первом ряду;

Наберите греческое "Хи" (в палитре символов второе поле справа - греческие символы).

Обратите внимание на различный вид курсоров. Вводимый символ вставляется в позицию, определяемую вертикальной чертой курсора!

Подведите курсор в поле верхнего индекса и нажмите 2;

Введите "=" после Х;

— из палитры шаблонов выберите знак суммы с верхним и нижним индексами, и введите индексы;

Выберите из палитры шаблонов объект с верхним индексом (первый в четвёртом ряду);

Выберите шаблон со скобками;

Выберите шаблон для дроби;

— Выберите шаблон с нижним индексом, введите Хи, переведите курсор в следующую позицию (стрелкой -> или шелчком мыши), наберите "-", затем "m"

В знаменателе введите "σ"

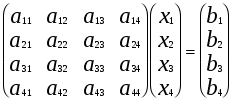
В месте верхнего индекса наберите 2;

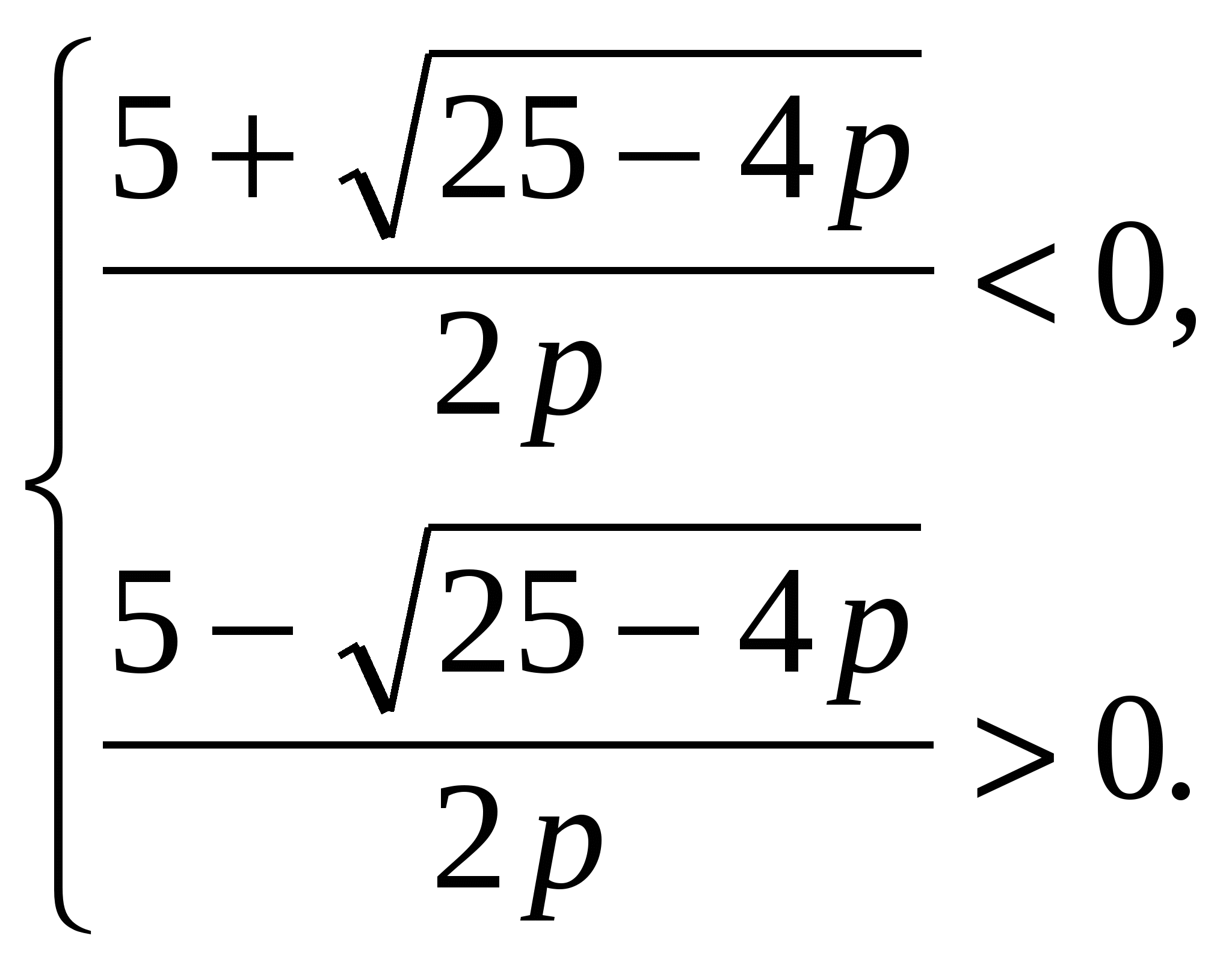
Выйдите из редактора формул, щёлкнув левой кнопкой мыши вне поля редактирования.

Сохраните формулу в файле.

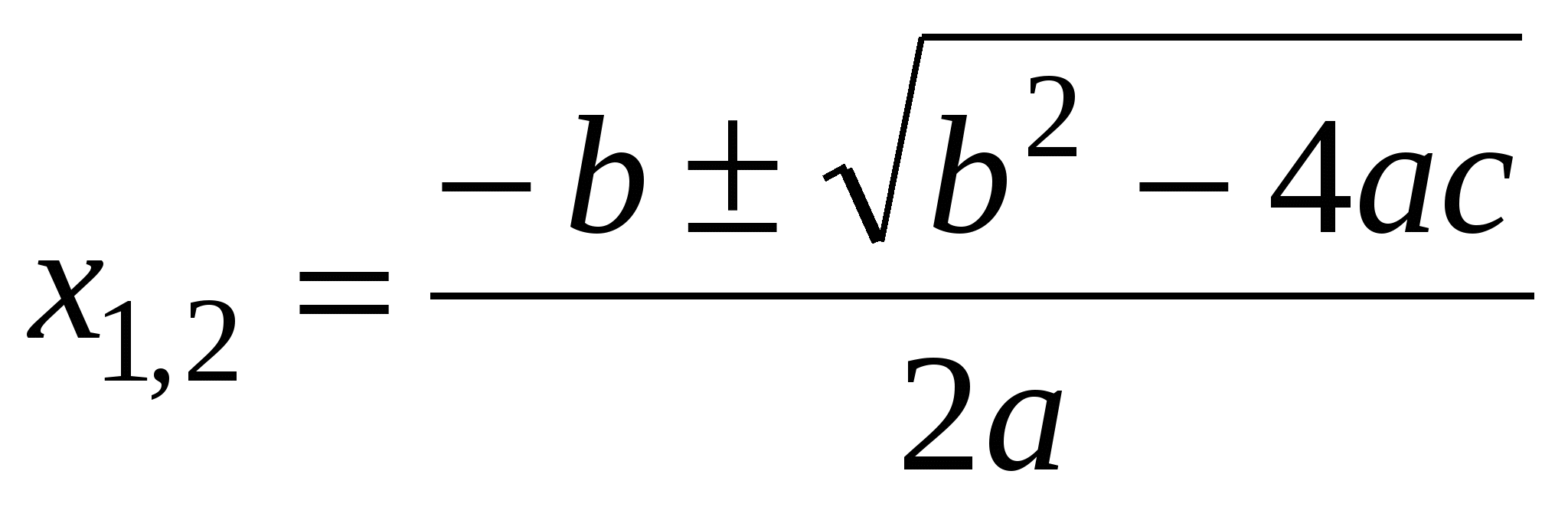
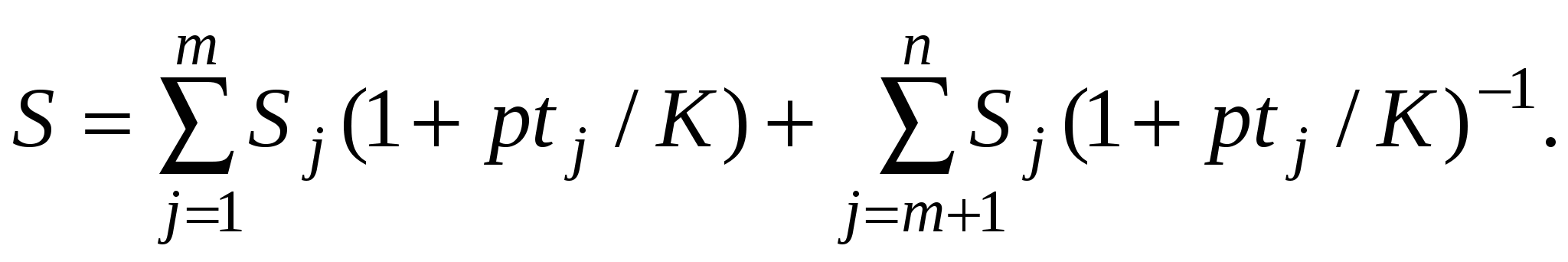
*Задание 2.*

Наберите систему линейных уравнений в матричной записи в виде:

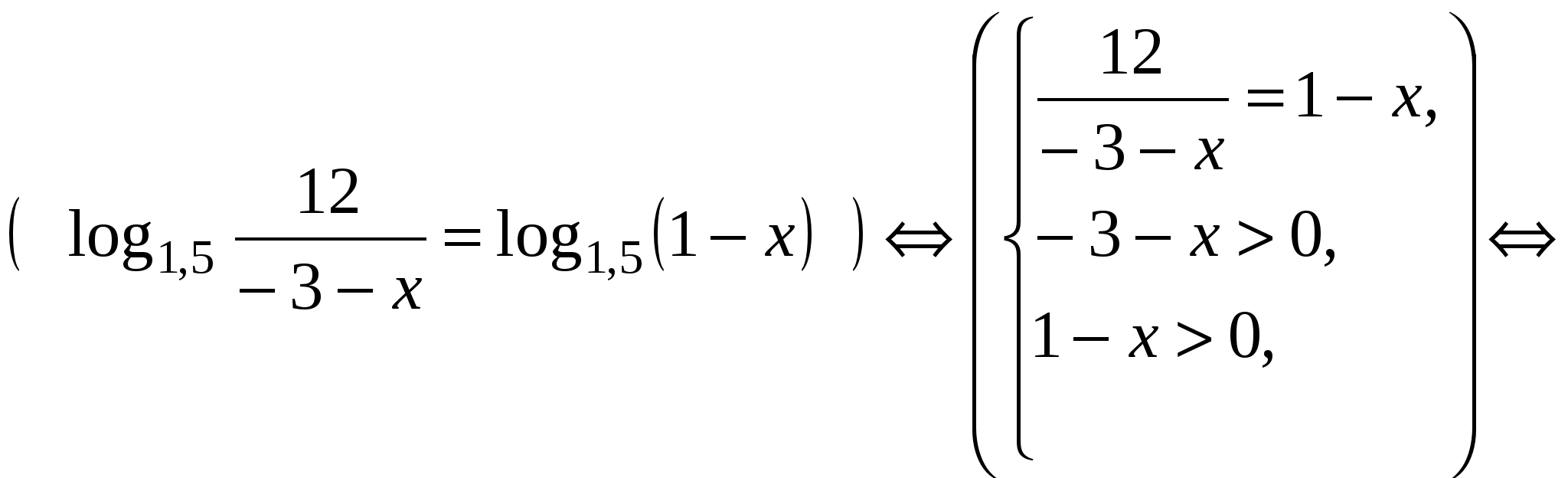


Задание 3. Наберите систему неравенств   


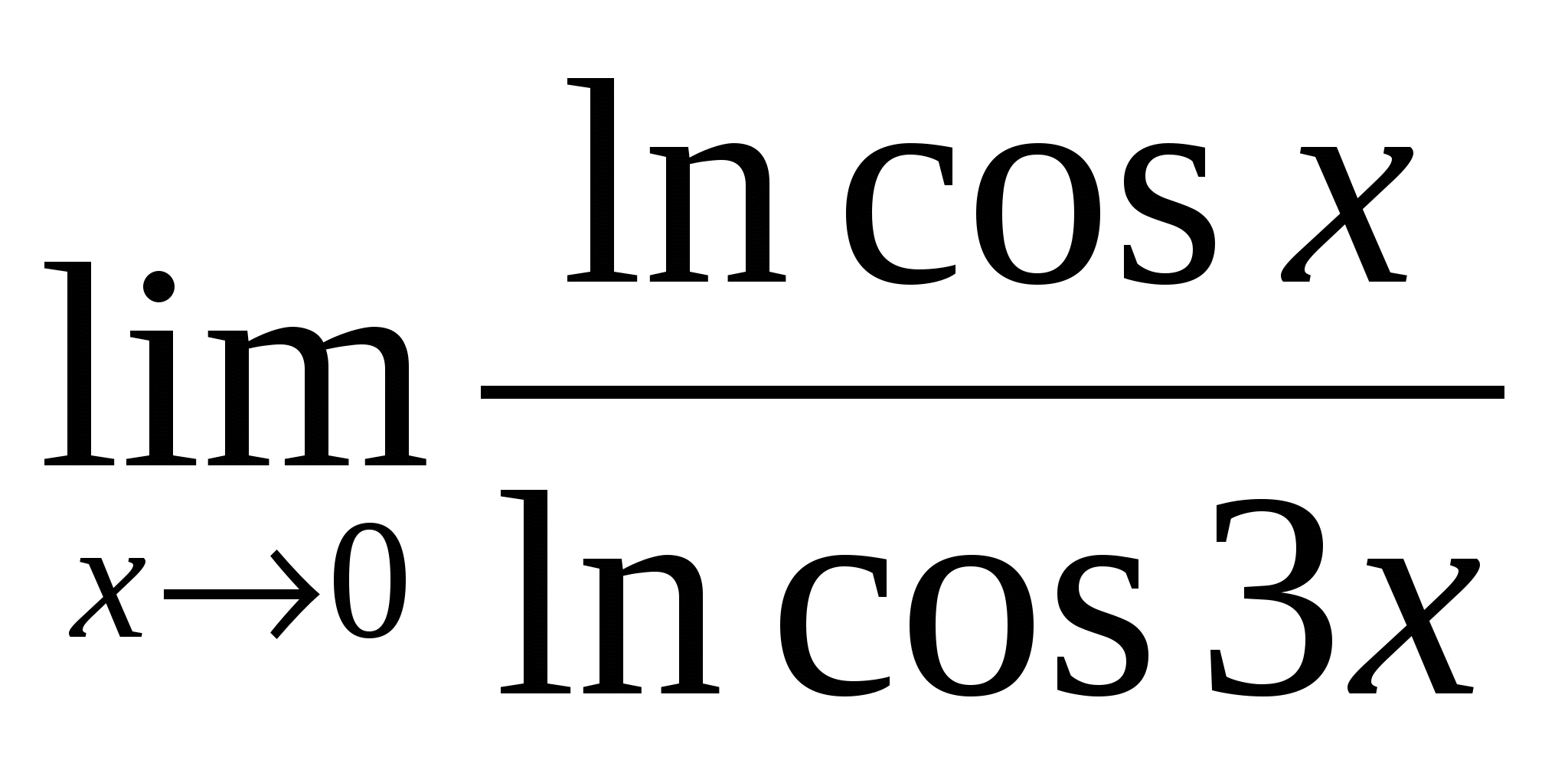
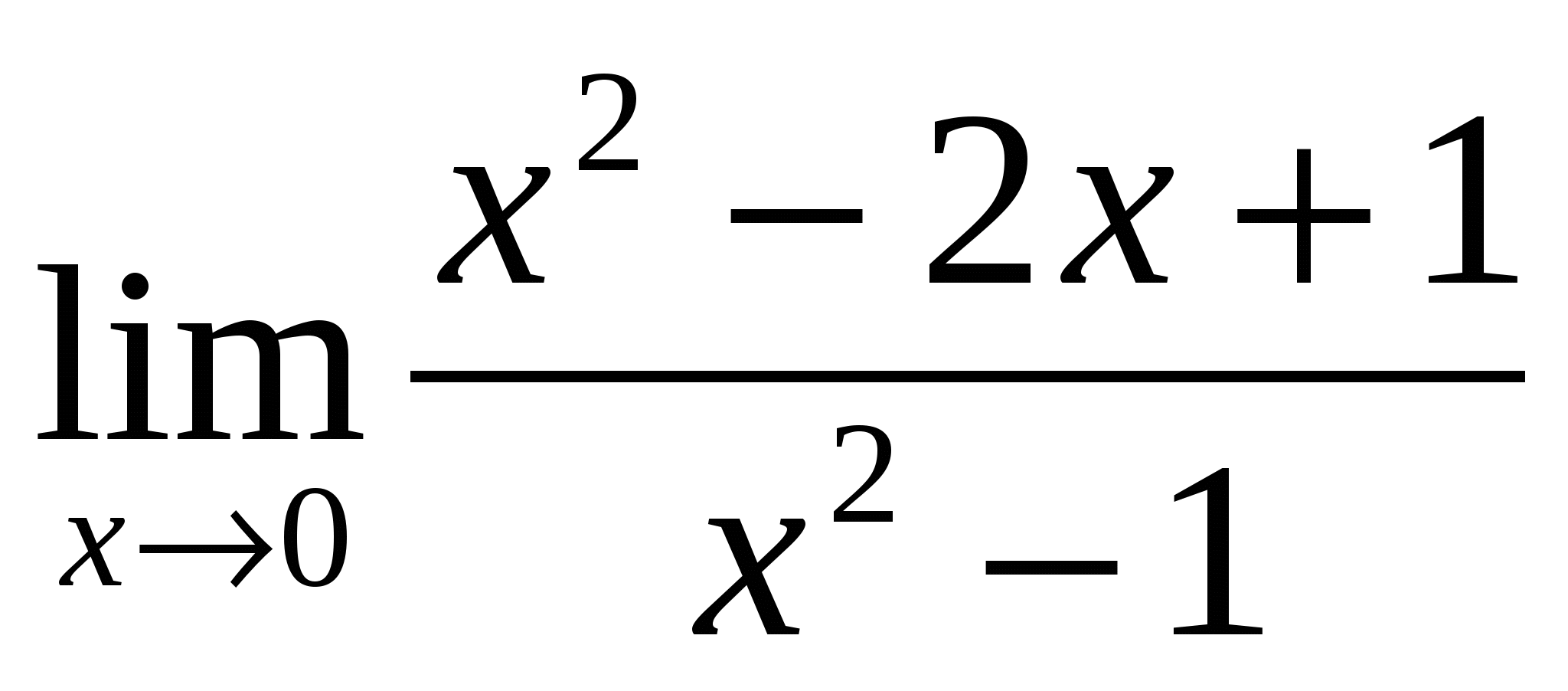
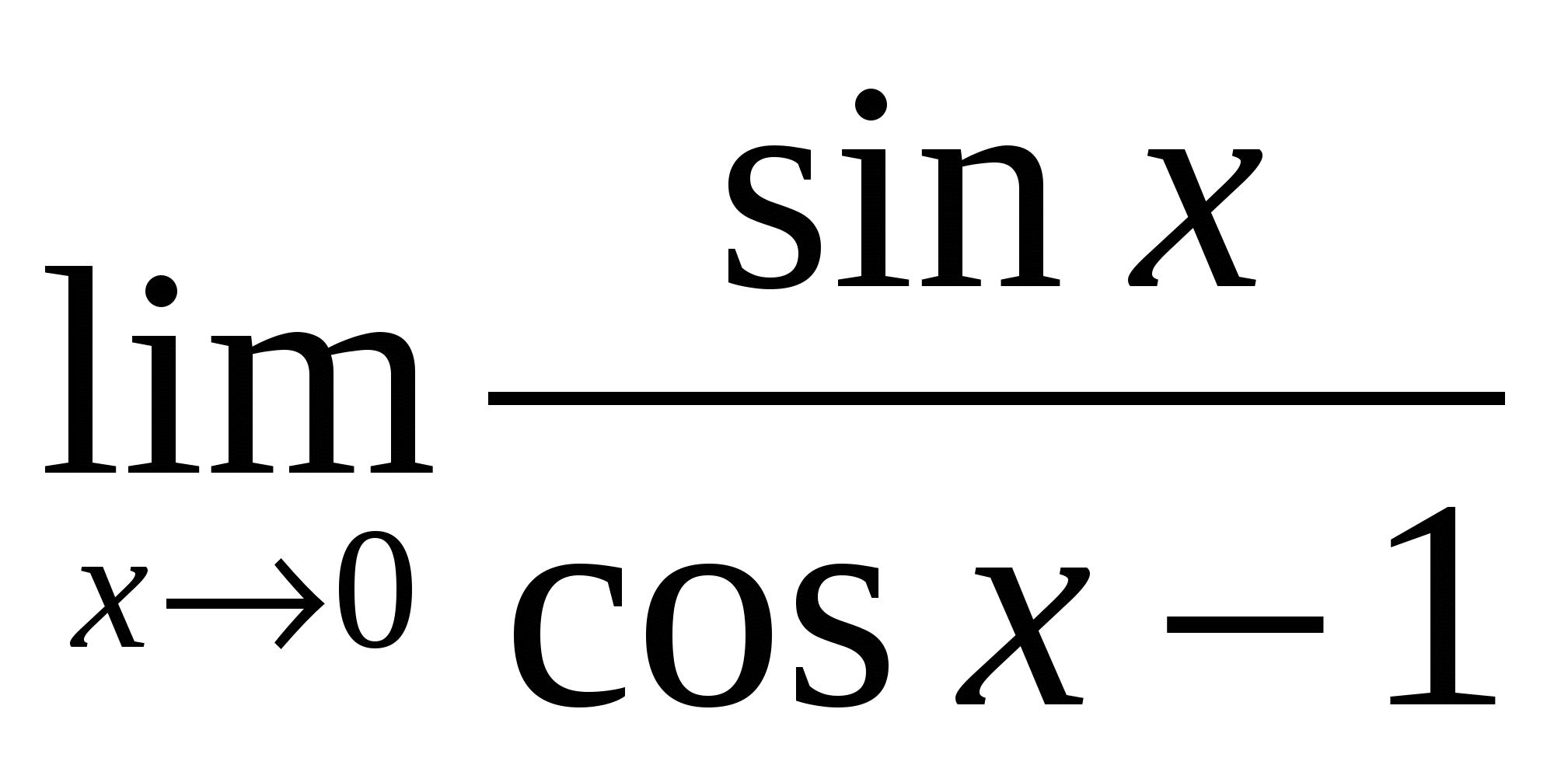
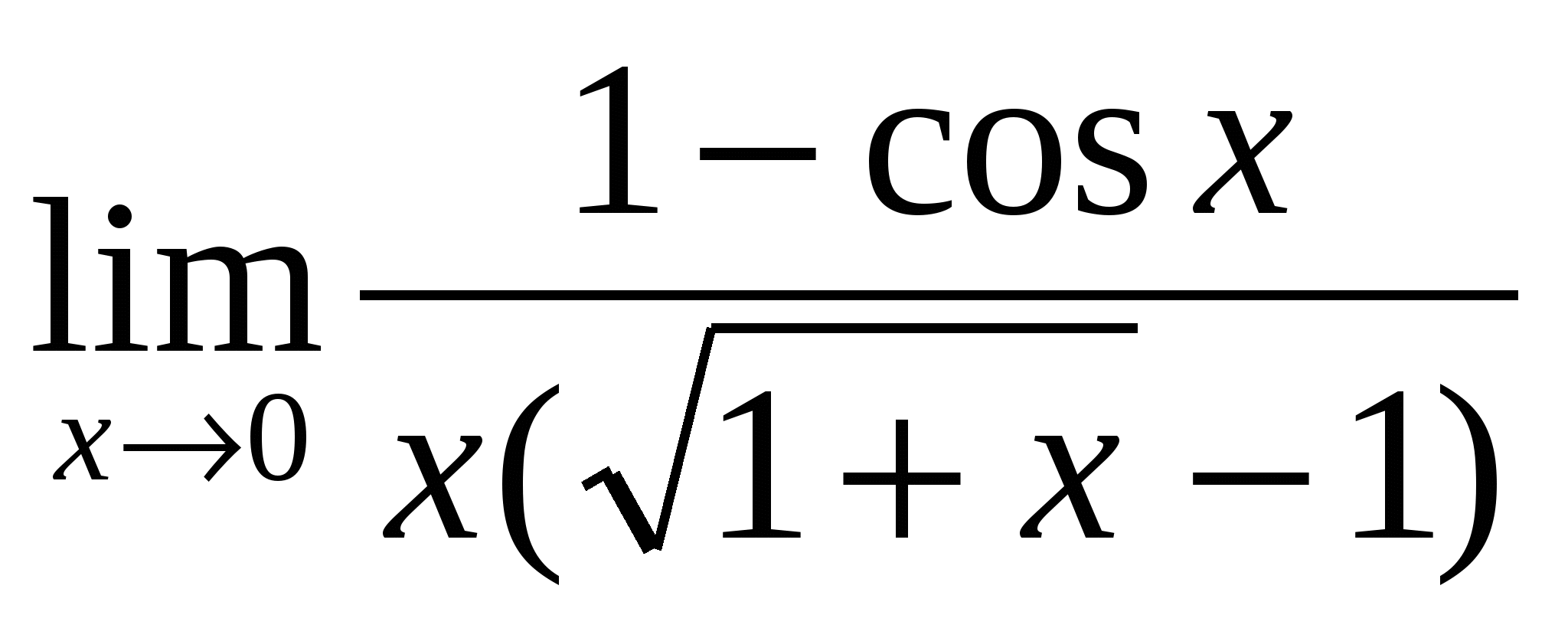
Задание 4. Наберите формулу вычисления корней квадратного уравнения

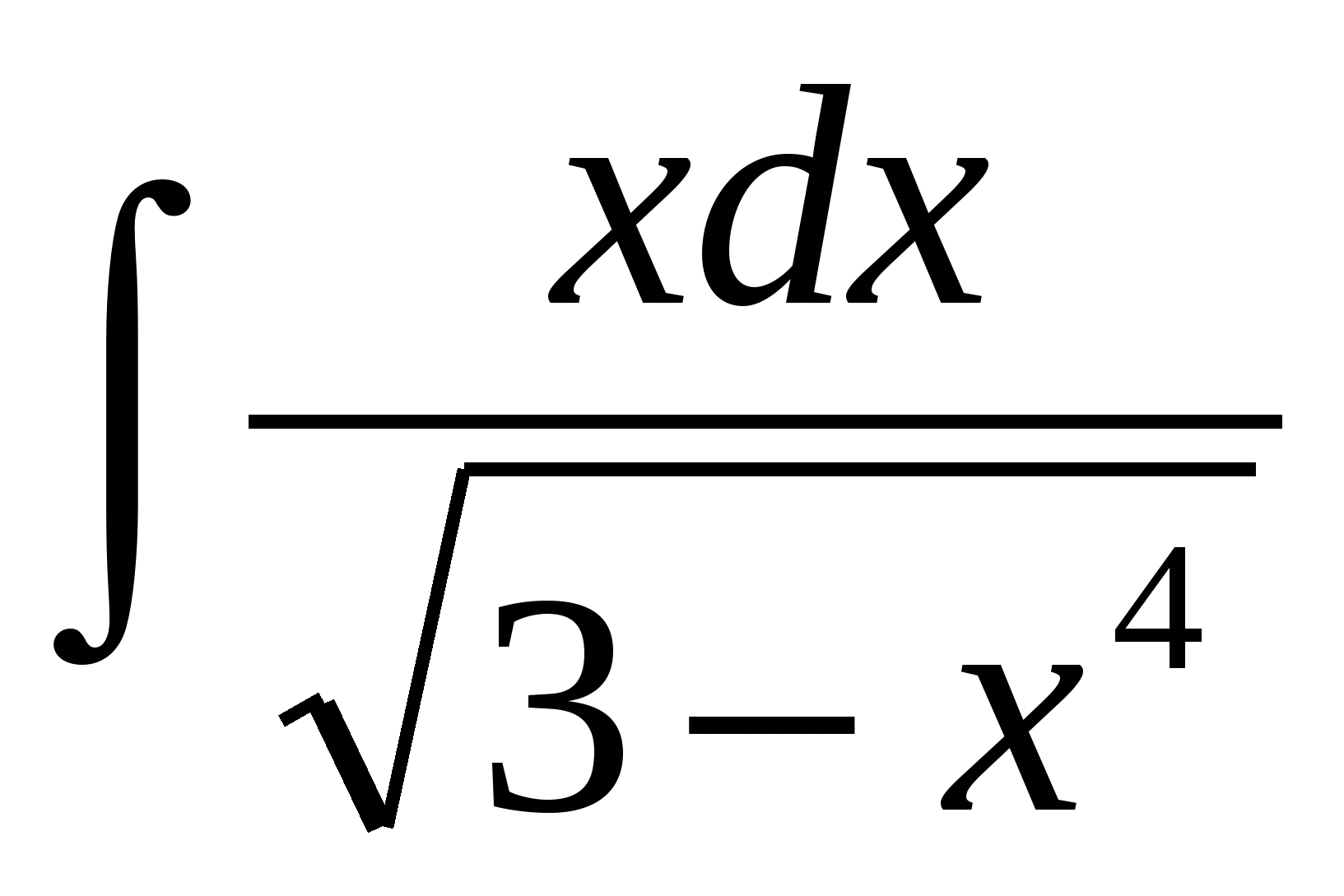
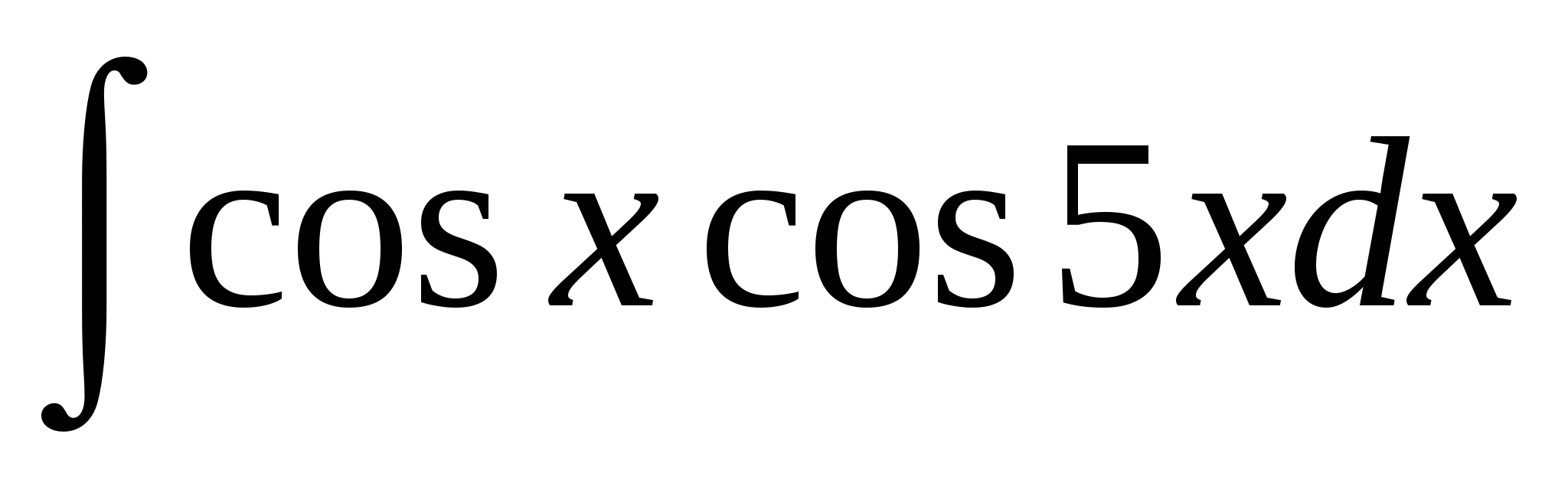
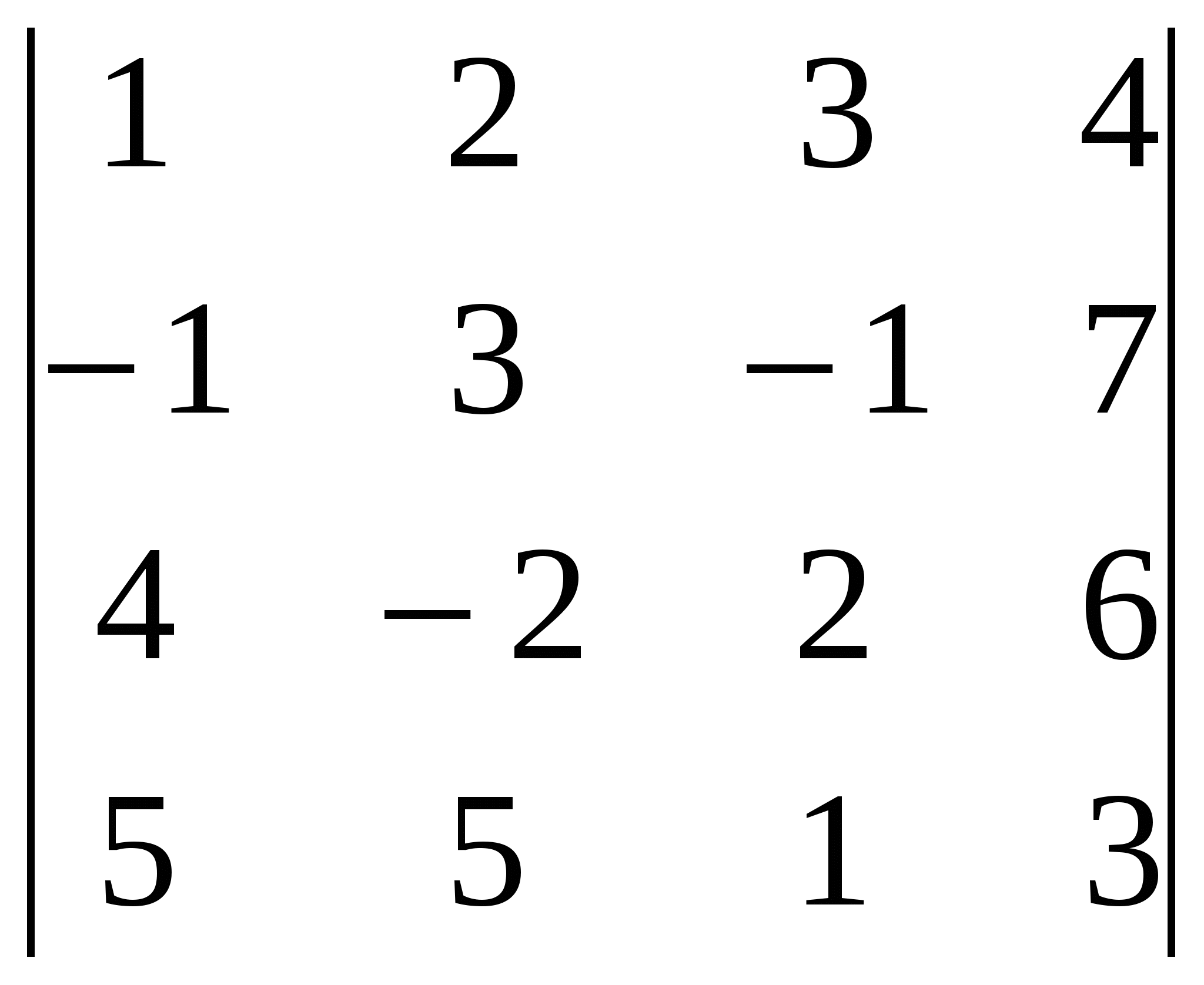
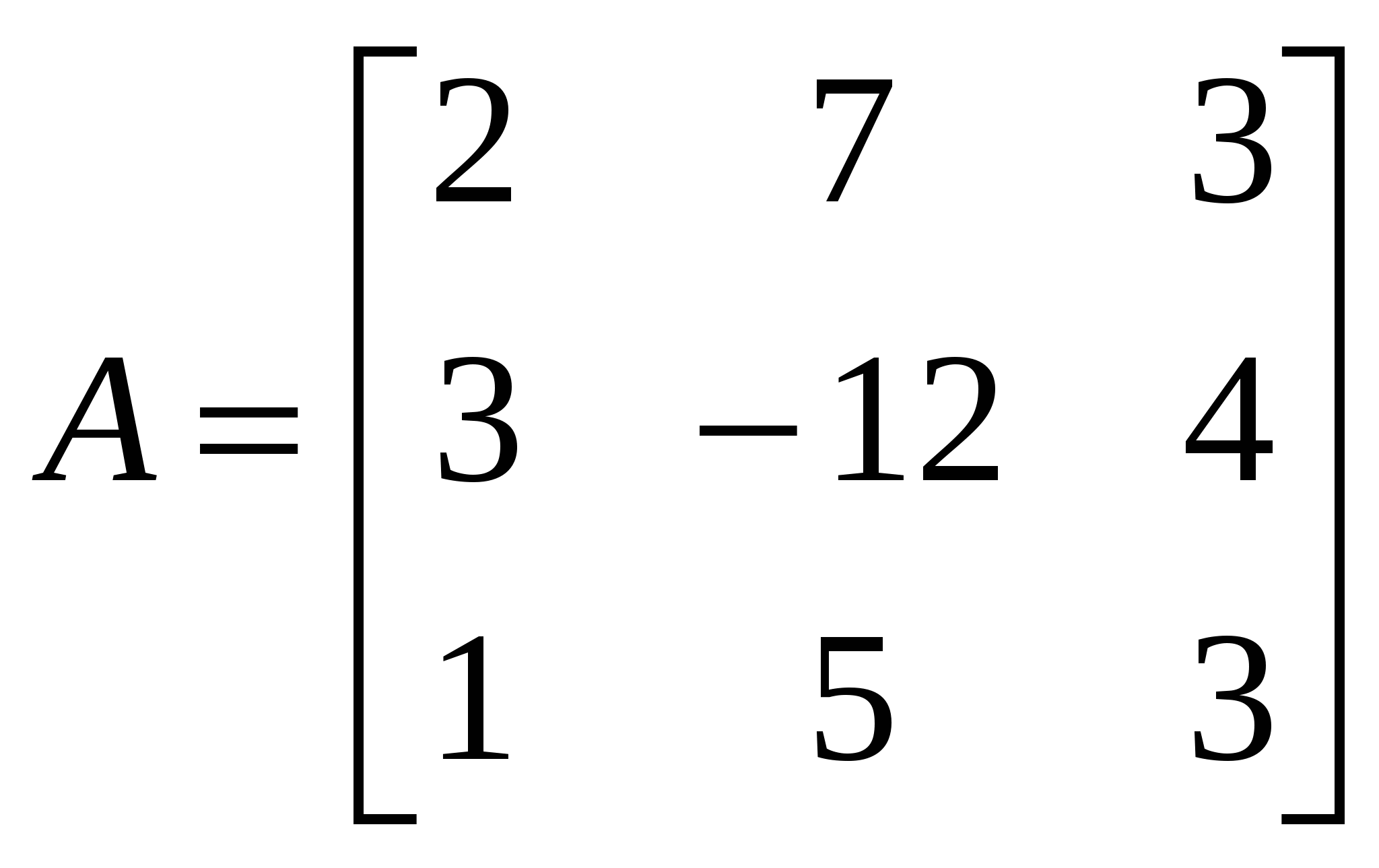
  
  
Задание 5. Наберите формулу вычисления консолидированного платежа   
  


Задание 6. Наберите текст решения уравнения



Задания для самостоятельной работы.

* 1. , b) , c) .d)

1. a), b) , c) .
3. .
4. .

**Практическое занятие № 23. Подготовка текстов и демонстрационных материалов**

Использование готовых шаблонов и создание собственных.

***Упражнение 1. Создать бланк по образцу***

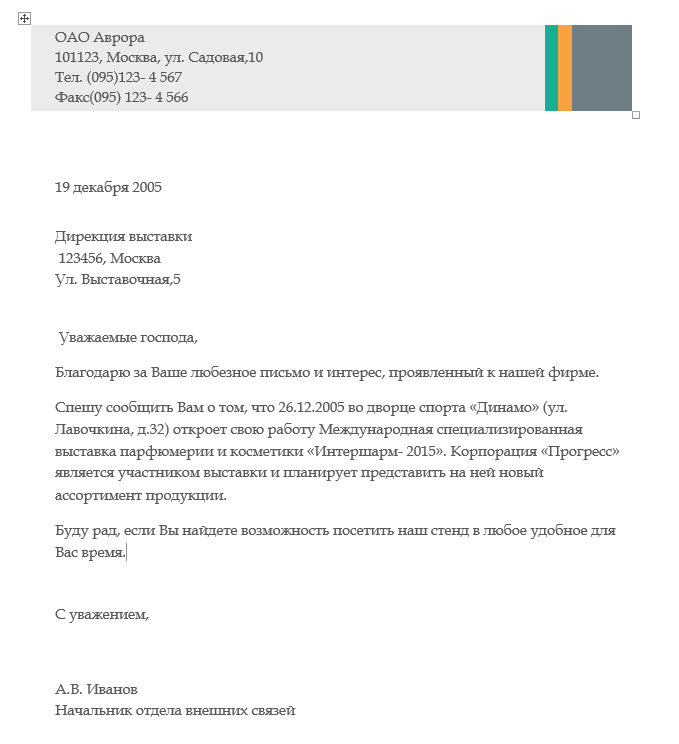
1. **Создать документ** *Письмо*.

1.1. В меню *Файл-Создать* найти вкладку шаблонов писем и просмотреть ее состав.

1.2. Выбрать стандартный шаблон *Деловое письмо* для создания документа на его основе.

1.3. Создать на основе шаблона экземпляр письма, введя необходимый текст и заполнив поля шаблона.

1.4. Сохранить созданный документ под именем *Письмо\_1*.



***Упражнение 2. Создать бланк по образцу***

**1. Создание собственных шаблонов пользователя.**

1.1. Для создания пользовательского шаблона *Приказа* вызвать меню *Файл-Создать*.

Пользовательский шаблон создавать на основе шаблона *Обычный,* в соответствии с рис. 1

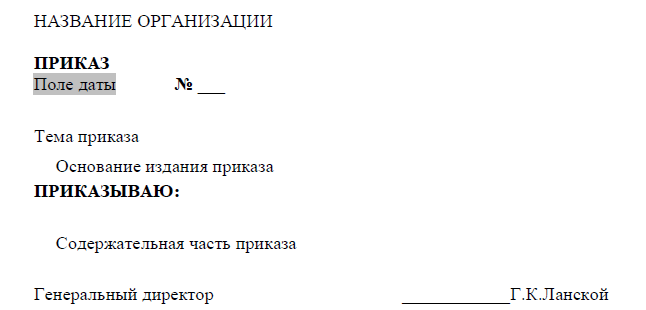


Рис. 1

1.2. .Сохранить шаблон под именем *Шаблон\_2* в своей папке в предварительно созданной папке *Шаблоны.*

**Примечание.** Указать путь к шаблону, пользуясь командой *Сервис-Параметры-Расположение.*

1.3. Создать документ, на основе пользовательского шаблона *Шаблон\_2*, оформленный, как показано на рис. 2, включив в него все необходимые атрибуты приказа.

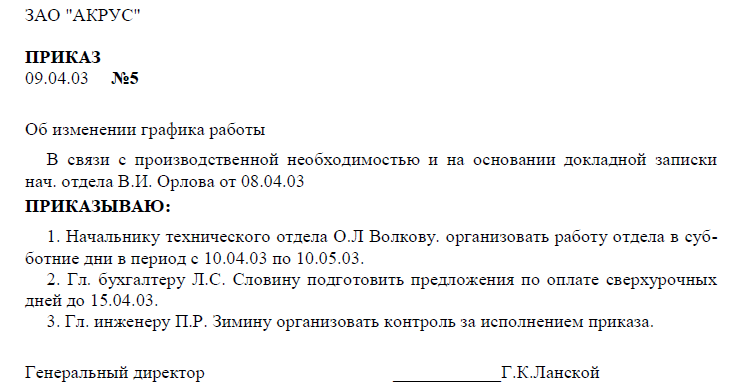
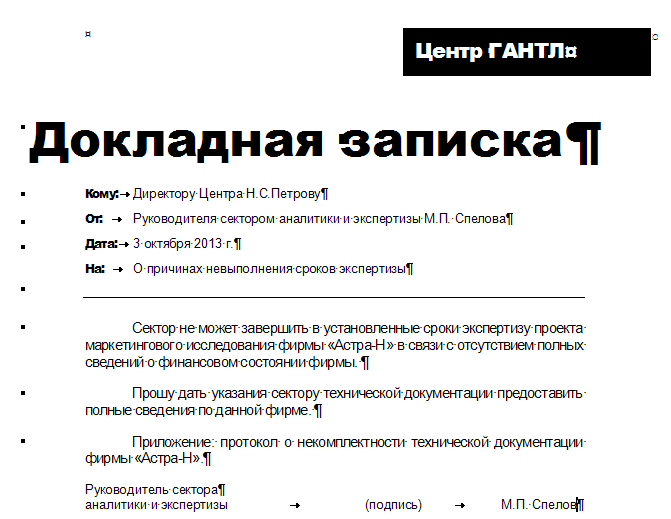


Рис. 2

1.4. Сохранить его как *документ под именем Приказ* в своей папке*.*

***Упражнение 3. Создать Докладную записку на основе шаблона Служебная записка****.*



Сохранить его как документ под именем *Докладная записка* в своей папке*.*

**Практическое занятие № 24. Подготовка текстов и демонстрационных материалов**

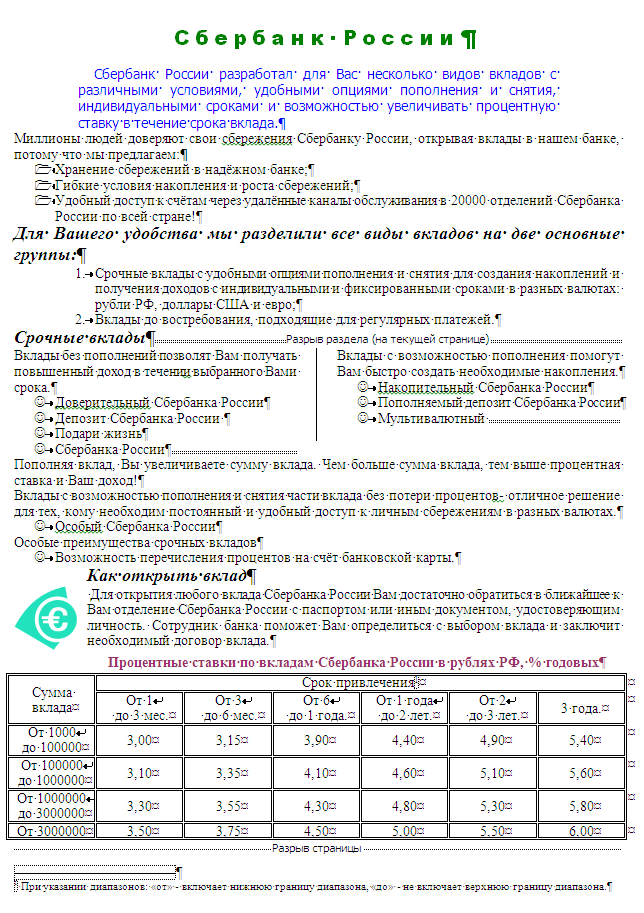
Средства поиска и замены. Системы проверки орфографии и грамматики. Нумерация страниц. Разработка гипертекстового документа: определение структуры документа, автоматическое формирование списка иллюстраций, сносок и цитат, списка используемой литературы и таблиц. Библиографическое описание документов. Коллективная работа с документами. Рецензирование текста.

Цель: Подготовка документа к печати.

**Практические задания**

Параметры формирования:

1. Создайте документ с полями правое, верхнее и нижнее поля-1 см., левое поле-2 см.
2. Оформите заголовок документа: шрифт: Arial Black, полужирный, размер шрифта 18, цвет текста зелёный, , выравнивание по центру, интервал разреженный на 5пт. Интервал перед и после по 12пт.
3. Параметры следующего абзаца: шрифт Tahoma, размер 12, цвет шрифта – синий, выравнивание по ширине. Отступ слева и справа по 2 см., первая строка – отступ 0,5см.
4. Основной текст оформите шрифтом Times New Roman, размер 12, Выравнивание по ширине.
5. Для заголовков 2-го уровня используйте шрифт Times New Roman, размер 16, начертание – полужирный курсив.
6. Оформите списки и колонки в соответствии с образцом.
7. Заголовок таблицы выровнен по центру, начертание полужирное, размер шрифта 12,цвет символов вишневый, интервал перед 5пт, после 3 пт..
8. Текст в таблице выровнен по центру, границы оформите синей двойной линией.
9. Из коллекции картинок категории Бизнес вставьте картинку.
10. Установите формат рисунка: высота – 2 см, положение – вокруг рамки.



**Практическое занятие № 25. Подготовка текстов и демонстрационных материалов**

Технические средства ввода текста. Распознавание текста. *Распознавание устной речи.* *Компьютерная верстка текста. Настольно-издательские системы.*

Цель: создание комплексного документа.

Практические задания

Создайте документ, при сохранении документа укажите Ваше имя и фамилию.

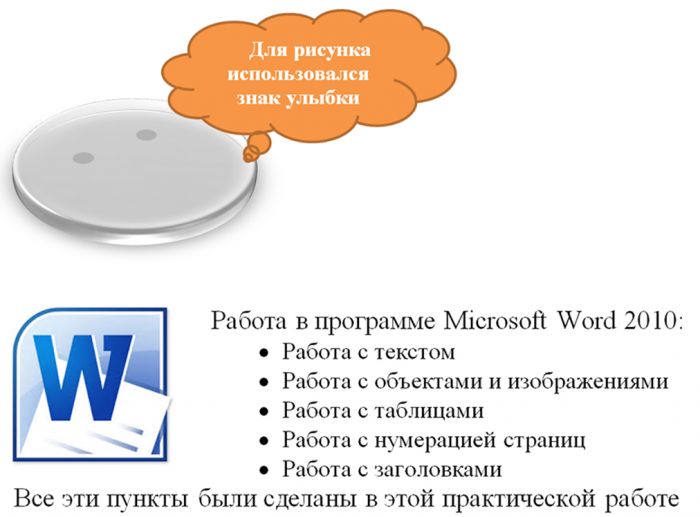
1. Задайте разметку страницы следующим образом
   * Сверху = 1,27 см
   * Снизу = 1,27 см
   * Справа = 1,27 см
   * Слева = 1,27 см
2. Создайте верхний колонтитул и укажите по центру название учебного заведения, а справа дату и время.
3. Создайте нижний колонтитул и укажите слева свое имя и фамилию, по центру группу и справа Ваш электронный адрес.
4. На первой странице оформите содержание документа
5. На второй странице будет находиться “Задание №2” со следующими параметрами:
   * Заголовок 1, шрифт Times New Roman 24 bold
6. Заголовок для третьей страницы будет “Таблица” с параметрами:
   * Заголовок 1, шрифт Times New Roman 24 bold
7. Так же на третьей странице будет находиться таблица. Со следующими параметрами:
   * Шрифт TimesNewRoman 12
   * Ширина таблицы 13см.
   * Раскрасьте ячейки таблице как показано ниже на рисунке 1

  
*Рисунок 1 Пример таблицы*

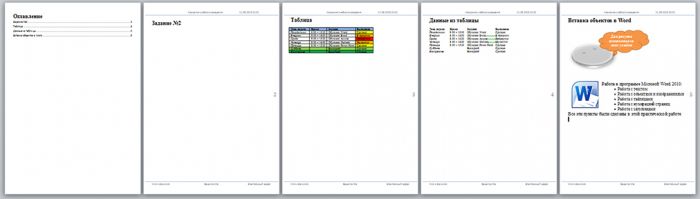
1. На четвертой странице сделайте заголовок “Данные из таблицы”. Со следующими параметрами:
   * Заголовок 1, шрифт Times New Roman 24 bold
2. Преобразуйте текст из таблицы, как показано на рисунке 2

  
*Рисунок 2 Текст из таблицы*

1. На пятой странице, сделайте заголовок “Вставка объектов в Word”. Со следующими параметрами:
   * Заголовок 1, шрифт Times New Roman 24 bold
2. Вставьте и отформатируйте объекты, как показано на рисунке 3.
3. Найдите логотип Word в интернете и сделайте ему обтекание текстом, как показано на рисунке 3.

[](http://digipo.eu/images/microsoft/word/tablica3.jpg)  
*Рисунок 3 Работа с объектами*

1. Пронумеруйте страницы со следующими параметрами:
2. Вставьте нумерацию по центру справа
3. На первой странице номер и колонтитул не ставить.

[](http://digipo.eu/images/microsoft/word/zadanie2.jpg)

*Рисунок 4. Пример готовой работы*

**Практическое занятие № 26. Работа с аудиовизуальными данными**

Технические средства ввода графических изображений. Кадрирование изображений. Цветовые модели. Коррекция изображений. Работа с многослойными изображениями. Работа с векторными графическими объектами. Группировка и трансформация объектов. Технологии ввода и обработки звуковой и видеоинформации. *Технологии цифрового моделирования и проектирования новых изделий. Системы автоматизированного проектирования. Разработка простейших чертежей деталей и узлов с использованием примитивов системы автоматизированного проектирования. Аддитивные технологии (3D-печать).*

Цель: обрабатывать аудио, визуальный контент и медиа-файлы средствами звуковых, графических и видео-редакторов;

**Программное****обеспечение***: звуковой редактор*Audacity

**Вопросы для самопроверки:**

1. Какая комбинация клавиш используется для открытия нужного файла в программе Audacity? Открытия нового окна?
2. Для чего используются эффекты: плавное нарастание и плавное затухание?
3. Каково назначение эффектов: «дилей», «эхо», «реверберация», нормализация, тремало?

**Задание №1.**

Создание рингтона

1. Открыть программу Audacity.
2. Для того, чтобы открыть нужный файл используем команду *Открыть*из меню *Файл* или комбинацию клавиш CTRL+O. В рабочей области программы откроется выбранная вами песня.
3. Нажав Play (пробел) можно прослушать всю песню, запоминая примерное расположение того участка композиции, который вы желали бы услышать в виде рингтона. Текущее время воспроизведения видно в нижней части окна программы.
4. Определив значение, вносим его в поле «начало выделения» и «конец». При этом в основной части программы более темным цветом будет показан определенный нами участок композиции, то есть будущий наш рингтон.
5. Пункт 4 можно выполнить простым выделением мышью необходимого нам участка песни.
6. Скопируем выделенный участок с помощью меню Правка – Копировать или CTRL+C.
7. Создайте новое окно CTRL+N.
8. Вставьте с помощью меню Правка – Вставить или CTRL+V.
9. Придадим нашему рингтону более законченное звучание. Для чего можно использовать в начале и конце файла Эффект *Плавное нарастание*: выделите участок длительностью в пару секунд – меню *Эффекты* – *Плавное нарастание (Плавное затухание)*.
10. При желании можно использовать другие эффекты.
11. Для создания громкого рингтона необходимо полностью выделить композицию (Ctrl+A) – меню *Эффекты – Усиление сигнала*. Передвигая ползунок, установите нужный уровень сигнала. Установите флажок «*Разрешить перезагрузку сигнала*».
12. Сохраним файл с помощью меню *Файл – Сохранить как*. Но сохранение файла происходит в формате .aup, который необходимо конвертировать в другие форматы. Для того, чтобы сохранить в необходимом вам формате выберите команду меню Файл – Экспортировать. Audacity поддерживает следующие форматы: .mp3, .aiff, .wav, .ogg, .mp4a, .amr и другие/

**Задание 2**

***С****оздать рингтоны «трезвучие: ля-до\_диез-ми» с использование эффектов «плавное нарастание», «плавное затухание», «нормализация», «дилей», «эхо», «реверберация».*

**Этапы выполнения работы:**

1. сгенерировать три следующих друг за другом тональных импульса с частотами 440, 550 и 660 Гц; длительность импульсов 0,25 с, пауза между импульсами 0,5 с; общая длительность сигнала 3 с (после третьего импульса следует пауза длительностью 0,75 с);
2. обеспечить плавное нарастание и затухание фронтов каждого импульса;
3. нормализовать уровень сигнала к -8 дБ;
4. сохранить на диске полученный «базовый» сигнал в виде проекта;
5. поочередно применить эффекты «дилей», «эхо», «реверберация» - результаты записать на диск в форматах wav;
6. конвертировать полученные файлы из формата wav в форматы mp3или mid.

**Задание 3**

*Создать рингтон с использованием эффекта «тремоло».*

**Этапы выполнения работы:**

1. сгенерировать тональный импульс частотой 575 Гц, амплитудой 0,11 и длительностью 3 с; во втором треке сгенерировать тональный импульс частотой 1500 Гц, амплитудой 0,33 и длительностью 3 с; в третьем треке сгенерировать тональный импульс частотой 2800 Гц, амплитудой 0,56 и длительностью 3 с;
2. произвести микширование трех полученных треков (Дорожки - Свести в последнюю дорожку);
3. применить эффект «тремоло» с параметрами**waveform type = inverse sawtooth; wetness level = 100; frequency = 20 Гц;**
4. сохранить на диске полученный «базовый» сигнал в виде проекта;
5. создать паузы длительностью 0,2 с через каждые 0,4 с и вновь записать проект под другим именем;
6. удалить возможные щелчки в конце каждого импульса с помощью плавного затухания заднего фронта на интервале одного зубца «пилы»;
7. подобрать комфортную высоту тона (Эффекты - Смена высота тона);
8. с помощью кнопки «Изменение огибающей» создайте эффект постепенного нарастания громкости звука;
9. сохранить на диске полученный сигнал в виде проекта;
10. конвертировать полученные файлы из формата wav в форматы шрЗ или mid;
11. скопировать полученные файлы в память мобильного телефона.

**Самостоятельная работа**

*Создать рингтон с использованием различных эффектов для мобильного телефона.*

**Задание для отчета:**

**Оформление результатов работы:**

1. Отчет о работе со скриншотами;
2. Демонстрация звуковых файлов.

**Задание на дом:** Доклад «Оцифровка звука с аудиокассет»

**Практическое занятие № 27. Электронные (динамические) таблицы**

Технология обработки числовой информации. Ввод и редактирование данных. Автозаполнение. Форматирование ячеек. Стандартные функции.

Цель: Изучить интерфейс приложения

Теория

**Excel** – программа, предназначенная для обработки и анализа числовой информации, хранимой в электронных таблицах в среде Microsoft Windows.

Microsoft Excel позволяет:

* создавать широкий спектр документов табличного вида; ёмкость электронной таблицы достаточную для реализации большинства документов;
* группировать документы в рабочие книги;
* представлять табличную информацию в графической форме (в виде диаграмм различного типа);
* выполнять сортировки и фильтрации данных в таблицах;
* создавать связанные и сводные таблицы;
* проводить аналитические исследования с помощью:
  + формул и связей;
  + диаграмм;
  + программы «поиск решения»;
* проводить статистический анализ данных и т.д.

Список расширений файлов, которые будут формироваться при работе с Excel:

**.xls** – файл таблицы (электронная таблица)

.**xlw** – расширение для определенного пользователем объединения нескольких файлов, которые должны обслуживаться под одним общим именем (рабочая папка);

.**xlt** –файл шаблона ( шаблоны – это файлы, содержащие заголовки для часто используемого бланка).

Итак. Из всех выше перечисленных возможностей Excel рассмотрим самую первую, или, другими словами, попробуем создать небольшой табличный документ.

**Пример 1.** Создание простейшего документа в Excel .

**Основные изучаемые операции:**

* запуск Microsoft Excel;
* создание рабочей книги;
* определение параметров страницы;
* выполнение простейших операций по вводу и редактированию данных;
* создание автоматически вычисляемых значений (итогов);
* использование стандартных функций;
* оформление заголовков документов.

Пусть требуется создать документ о результатах продаж соответствующих товаров по месяцам за полугодие:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| месяц | январь | февраль | март | апрель | май | июнь | итого |
| продукция |
| наименование  продукции 1 | 1 | 1 | 1 | 1 | 1 | 1 | 6 |
| наименование  продукции 2 | 2 | 2 | 2 | 2 | 2 | 2 | 12 |
| наименование  продукции 3 | 3 | 3 | 3 | 3 | 3 | 3 | 18 |
| Всего | 6 | 6 | 6 | 6 | 6 | 6 | 36 |

Для того чтобы создать этот документ, необходимо загрузить Microsoft Excel.

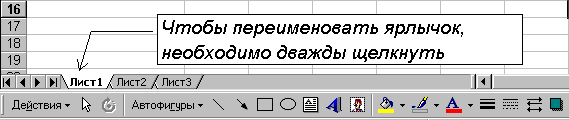
Запуск Excel:

*Вариант 1.* на рабочем столе Windows дважды щелкнуть по значку Microsoft Excel.

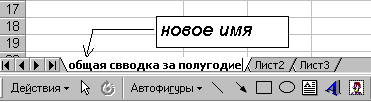
*Вариант 2.* через кнопку Пуск. Выбрав пункт меню Программы, найти Microsoft Excel.

При запуске Excel создает новую незаполненную рабочую книгу.

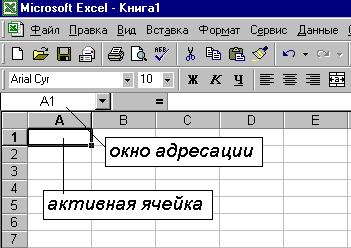
Рабочая книга – это обычный документ или тип файла в Microsoft Excel. Она представляет собой электронный эквивалент папки – скоросшивателя. Книга состоит из листов. Имя каждого листа выводится на ярлычок в нижней части рабочей книги. Листы можно перемещать и копировать из одной книги в другую, а также менять их порядок в пределах одной книги (нажав кнопкой мыши на наименование листа и не отжимая кнопку, перетащить лист).



Пусть первый активный лист будет называться «Общая сводка за полугодие». Переименуйте его.



В Excel работают в основном с рабочими листами. Рабочий лист представляет собой сетку, содержащую 256 столбцов и 65536 строк. Пересечение столбца и строки образует ячейку – основной элемент электронной таблицы для хранения данных. Каждая ячейка имеет свой уникальный адрес или ссылку. Например, ячейка, находящаяся на пересечении столбца С и строки 4, имеет адрес С4. Эти адреса используются при записи формул или ссылках на ячейки. Обычно вначале выделяется ячейка или ячейки, с которыми будут работать, а затем вводятся данные или выбирается команда. Выбранные ячейки выделяются на экране подсветкой. **Активной ячейкой** называется та, в которую в данный момент вводится информация или которая выделена толстой рамкой.



Ввод данных в таблицу.

Прежде чем создавать таблицу, необходимо проанализировать характер вводимых данных.

В выполняемом документе можно выделить такие группы:

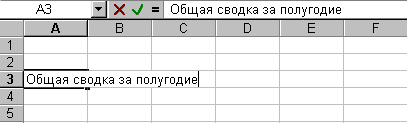
* заголовок документа;
* наименование строк и столбцов таблицы («шапка» таблицы);
* текстовые данные (наименование продукции);
* числовые данные (результат продаж за месяц – задаваемые значения);
* итоговые данные (суммы по строкам и столбцам – вычисляемые значения).

Каждая из выделенных групп данных требует определенных знаний и приемов при вводе и оформлении.

**Заголовок таблицы.**

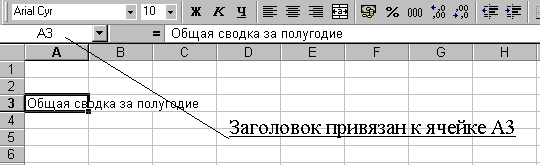
Неизвестно какой ширины будет таблица, поэтому заголовок следует разместить в первой колонке и пока не выравнивать по центру страницы.

Определите, в какой строке будет располагаться заголовок, например в 3-й строке. Активизируйте ячейку А3 (щёлкнуть кнопкой мыши по ячейке А3). Наберите требуемый текст: «Общая сводка за полугодие»



Для того чтобы отредактировать введенный текст, необходимо активизировать ячейку, к которой он привязан, и затем либо дважды щёлкнуть кнопкой мыши по тексту, либо перейти в строку ввода текста и формул.

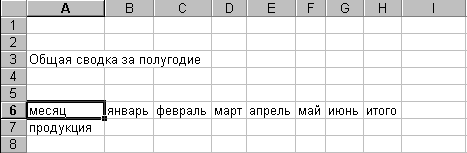
Обратите внимание, что, хотя текст и занимает на экране несколько ячеек, он всегда привязан (адресуется) только к одной ячейке.



**Создание наименований столбцов таблицы.**

В формируемом документе наименование столбцов таблицы занимает две строки.

Заполните соответствующие ячейки названиями столбцов, начиная с 6-й строки столбца А.

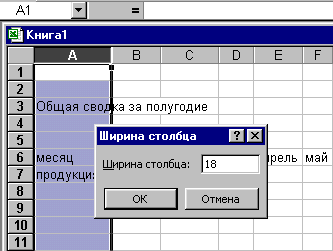


По умолчанию ширина всех столбцов одинаковая. Для создаваемого документа ширина первого столбца, содержащего наименование продукции, безусловно, должна быть больше, чем ширина остальных столбцов, которые заполняются числовыми данными.

**Изменение ширины столбца**

*Вариант 1.* Установить курсор на разделительную линию между столбцами А и В и, когда курсор мыши примет вид черной двунаправленной стрелки, не отжимая кнопку мыши, переместить границу на требуемое расстояние (ширина столбца указывается в окне подсказки).

*Вариант 2.* Выделить столбец А, нажав кнопкой мыши на наименование столбца, а затем через меню Формат – Столбец – Ширина указать точное значение ширины



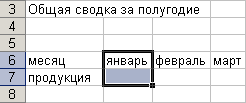
**Форматирование ячеек.**

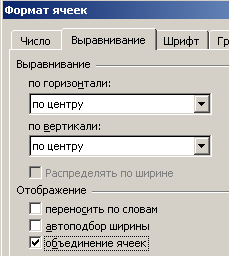
Для выравнивания текста по правому краю следует активизировать ячейку А6 и нажать на пиктограмму . Пиктограммы выравнивания текста действуют только для форматирования данных внутри ячеек таблицы. 

**Объединение ячеек**

Наименование столбцов с названиями месяцев требуется расположить в центре объединенных ячеек 6-й и 7-й строк.

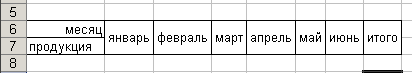
Выделить ячейки В6-В7. через меню Формат – Ячейки перейти на закладку Выравнивание и отметить параметр Объединение ячеек или нажать на кнопку на панели инструментов Форматирование . Операции объединения выполнить для остальных заголовков столбцов.





**Центрирование данных.**

Выделить ячейки, в которых требуется отцентрировать данные. Через меню Формат – Ячейки перейти на закладку Выравнивание и установить параметры: по горизонтали – по центру, по вертикали – по центру.



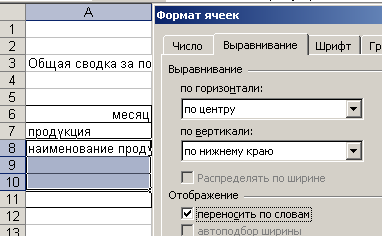
**Перенос текста внутри ячеек.**

Ячейки с наименованиями продукции (А8 – А10) содержат текст, который необходимо расположить с переносом в несколько строк и отцентрировать.

Для переноса текста внутри ячеек необходимо:

* выделить ячейки, в которых требуется перенос текста;
* через меню Формат – Ячейки перейти на закладку Выравнивание;
* отменить параметр переноса по словам;
* задать параметры Выравнивания: по горизонтали – по центру, по вертикали – по нижнему краю.

**Копирование данных.**

Текст в ячейках А8 – А10 повторяется за исключением цифр, поэтому целесообразно скопировать в две нижележащие ячейки и затем отредактировать. Для выполнения копирования требуется:

* активизировать ячейку, текст будет копироваться;
* через меню Правка – Копировать сохранить выделенную информацию в буфере временной памяти;
* выделить ячейки А9 – А10 и, указав меню Правка – Вставить, заполнить ячейки текстом.

Выполните редактирование данных в ячейках А9 – А10, заменив в них номер продукции.

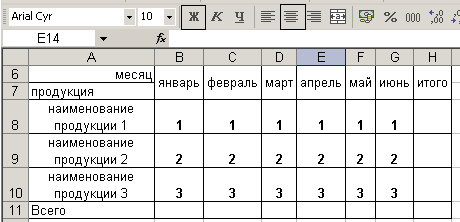
В ячейку А11 введите Всего.

**Ввод и форматирование числовых данных.**

В ячейки В8 – В10 введите соответственно числа 1-3.

Скопируйте значения из ячеек В8 – В10 в соответствующие ячейки по месяцам.

Отформатируйте значения в интервале ячеек В8 – G10: выравнивание - по центру ячеек, начертание – жирным.



**Ввод формул.**

**Формула –** это комбинация констант, вычислительных операторов и функций, которая вводится в ячейку для получения нового результирующего значения.

Вид формул может изменяться в зависимости от решаемых задач за счёт использования ссылок, функций, текста и имён.

**Арифметические операторы :**

+ сложение

- вычитание

/ деление

\* умножение

% процент

^ возведение в степень.

**Операторы сравнения:**

= равно

> больше

< меньше

>= больше или равно

<= меньше или равно

<> не равно

**Оператор текста.**

***&*** соединяет или сцепляет два или более текстовых значения в единое текстовое значение.

**Порядок вычисления операторов.**

% процент

^ возведение в степень

\* и / умножение и деление

+ и – сложение и вычитание

& присоединение текста

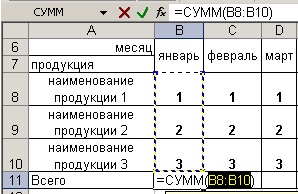
=<> <=> = <> сравнения.

Круговые скобки изменяют порядок вычисления: в первую очередь выполняются действия, объединённые в формуле круглыми скобками.

В создаваемом документе требуется подсчитать сумму по строкам и столбцам графах “Итого” и “Всего”.

Для того чтобы подсчитать сумму за январь (сумма по столбцу), необходимо:

* установить курсор в ячейку (активизировать ячейку), где будет размещена сумма (ячейка В11);
* нажать на верхней панели на пиктограмму  автосуммирования;
* завершить формирование суммы нажатием Enter.
* скопируёте содержимое ячейки В11 в ячейки С11 – G11.



Обратите внимание, как изменились ссылки на суммируемые ячейки. Ссылки позволяют использовать в одной формуле данные из различных областей, а так - же значения из одной ячейки в нескольких формулах.

**Типы ссылок.**

**Относительные ссылки** подобны определению навигации от текущего положения, т.е. такая ссылка определяет, как найти требуемую ячейку, стартуя из ячейки, содержащей формулу. Например, А6, F17.

**Абсолютные ссылки** определяют, как найти ячейку, отсчитывая ее местоположение от начала электронной таблицы. В абсолютных ссылках перед именем столбца и номером строки ставится знак $ (доллар). Например, A$6$, F$17$.

**Смешанные ссылки** содержат комбинации относительных и абсолютных ссылок. Например, A6$, F$17.

Различие между относительными и абсолютными ссылками проявляется, когда формулы копируются или пересылаются из одной ячейки в другую.

**Типы операторов ссылок**

**Оператор интервала** (двоеточие) определяет одну ссылку на все ячейки, расположенные между двумя ссылками включая последние (например: А2:С2. Результат: выделены ячейки А2, В2, С2).

**Оператор объединения** (точка с запятой) определяет одну ссылку, объединяя ячейки, на которые указывают две исходные ссылки (например: А2; С2. Результат: выделены ячейкиА2 и С2).

**Оператор пересечения** (пробел) определяет одну ссылку на ячейки, общие для двух ссылок (например: А5:D5 C4:C9. Результат: выделена ячейка С5).

**Изменение типов ссылки**

Чтобы поменять тип ссылок, который используется в формулах, необходимо:



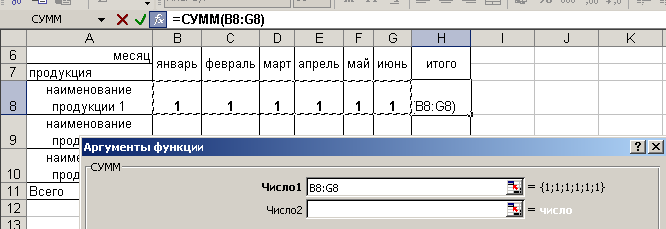
* Выделить ячейку или интервал ячеек, содержащих формулы;
* В строке формул выделить всю формулу либо только ту ее часть, где необходимо поменять тип ссылки;
* Нажатием клавиши F4, циклически изменяя типы ссылок, выбрать нужный тип;
* Нажатием на Enter завершить изменение ссылки.

**Создание формул с использованием встроенных функций**

Осталось ввести формулы для вычисления сумм по строкам в столбец «Итого» (ячейки Н8-Н11). Для этого требуется:

* активизировать ячейку Н8;
* перейти в строку ввода формул;
* ввести знак = (ввод любой формулы начинается со знака равно);
* выбрать через меню Вставка - Функция вычисление суммы;
* указать интервал ячеек, значения которых требуется просуммировать (переместить мышкой диалоговое окно функции и выделить требуемый интервал ячеек, который будет размещен в поле функции для суммирования);
* для завершения создания формулы нажать Enter.

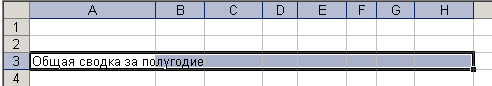
Cкопируйте созданную формулу в ячейки Н9-Н11. Отформатируйте значения столбца «Итого»: выравнивание - по центру ячеек, начертание - жирным.

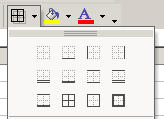


**Центрирование заголовка**

После того как таблица готова, можно оформить заголовок:

* выделить ячейки в строке заголовка (3-я строка) до границы листа;
* через меню Формат - Ячейки перейти на закладку Выравнивание и поставить опцию Объединение ячеек, затем установить параметры Выравнивание по горизонтали - по центру, Выравнивание по вертикали - по центру.



Документ готов. Нажмите на клавишу просмотра документа и оцените свои труды. Очевидно, что не хватает обрамления табличной части документа.

**Обрамление таблиц**

Для этого требуется:

* Выделить ячейки, входящие в табличную часть документа (А6-Н11);
* Раскрыть пиктограмму обрамления и выбрать и выбрать тип обрамления для создания границ таблицы.

Нажмите на клавишу для просмотра документа.

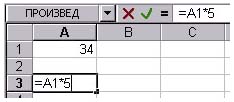
**Практическое занятие № 28. Электронные (динамические) таблицы**

Виды ссылок в формулах.

Цель: Изучить различные типы адресации ячеек

## Теория. Адресация ячеек.

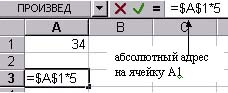
В Excel существует два типа адресации ячеек (ссылки на ячейку). (**Ссылка** - это формализованное обращение к другой ячейке.) Относительный адрес показывает, как расположена та или иная ячейка относительно заданной. Например, формула содержащая адрес **А1** находится в ячейки **А3**.



Это значит, что речь в формуле идет о ячейки, которая находится через одну ячейку над ячейкой с формулой. И при копировании (например, в ячейку **В5**) или переносе адрес автоматически изменится таким образом, чтобы в новом положении он показывал на ячейку, которая расположена точно так же.



**Абсолютный адрес** всегда указывает на точный адрес ячейки, вне зависимости от того, где располагается формула, использующая эту ячейку. **Внешним признаком** абсолютного адреса является наличие знака **$** перед значением координаты в адресе ячейки. Знак **$** может стоять в адресе, как перед обозначением столбца, так и перед номером строки. Например, адрес **$А$1** - абсолютный адрес, при любом копировании и перемещении он останется неизменным и всегда будет указывать на ячейку, находящуюся на пересечении столбца **А** и строки **1**.





В адресах: **$F3** и **B$2** сочетаются абсолютная и относительная адресация (смешанная адресация). В первом случае (**$F3**) абсолютная адресация используется для задания столбца, а относительная - для строки, во втором случае (**B$2**) - наоборот.

Обычно в формулах используется относительная адресация, так как в этом случае при копировании и перемещении формул полностью сохраняются взаимосвязи между ячейками таблицы. Абсолютная адресация используется при включении в формулу неизменных величин, которые всегда расположены в одних и тех же ячейках, независимо от положения ссылающихся на них формул.

**Пример 1. Расчеты в таблицах**

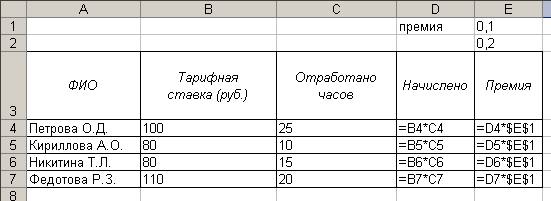
Рассчитать величину начисленных денег и величину выдаваемой премии для каждого сотрудника отдела по формулам:

Начислено = Отработано часов\*Тарифная ставка.

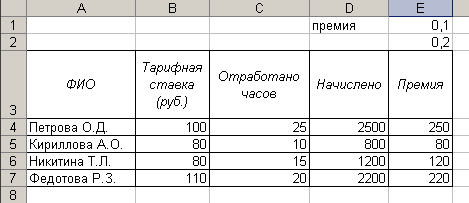
Премия = Начислено \*Процент премии.

Величина Процент премии – 10% - хранится в отдельной ячейке Е1.

Формулы, по которым рассчитываются соответствующие таблицы, приведены ниже



Результаты вычисления по формулам:

****

**Пример 2. Абсолютные ссылки**

*Задание*. Составьте таблицу распределения дивидендов (выплаты доходов акционерам по итогам работы) в акционерном обществе в соответствии с количеством акций каждого акционера.

1. Сначала нужно определить исходные данные задачи: величину распределяемой суммы дохода (число), фамилии работников (текст), количество акций у каждого акционера (целое число). В качестве рассчитываемых данных в таблице будет суммарное значение числа всех акций (число) и сумма выплаты каждому акционеру (число).
2. Запустите Excel, переименуйте Лист1 в Дивиденды, опишите на нем структуру таблицы и введите исходные данные следующим образом
3. В ячейку В8 введите формулу суммы диапазона В4:В7; для этого, выделив диапазон В4:В7, щелкните кнопку Автосумма на панели инструментов Стандартная.

|  |  |  |  |
| --- | --- | --- | --- |
|  | А | В | С |
| 1 | Распределение дивидендов в акционерном обществе | | |
| 2 | Сумма доходов |  |  |
| 3 | Фамилия | Количество акций | Выплата |
| 4 | Иванов | 5 |  |
| 5 | Петров | 6 |  |
| 6 | Сидоров | 8 |  |
| 7 | Костин | 4 |  |
| 8 | Итого: |  |  |

1. Введите в ячейку В2 величину распределяемого дохода, например, 10000.
2. Введите в ячейку С4 формулу расчета выплаты =$В$2/$В$8\*В4. (Выплата каждому акционеру равна частному от всей суммы доходов акционерного общества и количества всех акций, умноженному на количество акций данного акционера.) Обратите внимание, что адреса ячеек В2 и В8 записаны в абсолютной форме, т.е. они не изменятся при копировании формулы из ячейки С4 в С5:С7.

*Примечание*. Для создания абсолютной ссылки на ячейки В2, В8 нужно при вводе формулы щелкнуть на данной ссылке клавишей **F4**.

1. Задайте финансовый формат для значений данных в ячейках С4:С7, для чего, выделив этот диапазон, выберите в меню **Формат** опцию ячейки, а затем на вкладке **Число** окна *Формат ячеек* выберите вариант числового формата: **Финансовый**, после чего укажите число десятичных знаков 2, а в поле **Обозначение** выберите **р.** и щелкните кнопку «ОК».
2. Измените ширину столбца С, чтобы в нем правильно отображались выплаты в финансовом формате.
3. Проверьте действие таблицы, изменяя данные в ячейках В2,В4:В7.
4. Сохраните таблицу с диаграммой под именем **Дивиденды** и закройте окно программы Excel.

**Упражнения:**

Уровень 1

1. Создайте таблицу:

* При заполнении используйте: автозаполнение, копирование содержимого ячеек;
* Заголовки отцентрируйте, установите размер шрифта 14, полужирный;
* Задайте денежный формат для столбцов: Цена и Сумма;
* Используйте автосуммирование для нахождения итоговых значений по столбцам (ячейки строки Итого);
* Найдите значения в ячейках столбца Сумма по следующей формуле:   
  Сумма = Цена \* Кол-во за год

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № | Наименование | Ед.изм. | Цена, р. | 1 кв | 2 кв | 3 кв | 4 кв | Кол. за год | Сумма |
| 1 | Тетрадь | шт. | 25 | 10 | 15 | 8 | 14 |  |  |
| 2 | Ручка | шт. | 28 | 9 | 12 | 11 | 10 |  |  |
| 3 | Карандаш | шт. | 15 | 15 | 10 | 12 | 16 |  |  |
| 4 | Блокнот | шт. | 60 | 4 | 7 | 9 | 6 |  |  |
| 5 | Маркер | шт. | 40 | 7 | 5 | 8 | 6 |  |  |
| Итого | |  |  |  |  |  |  |  |  |

1. Постройте и заполните таблицу начисления зарплаты, если размер премии составляет 20% оклада.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | **ФИО** | **Оклад** | **Премия** | **Зарплата** |
| 1 | Арсентьева В.А. | 30000 | ? | ? |
| 2 | Петров М.Н. | 25 000 | ? | ? |
| 3 | Романов К.Н. | 20 000 | ? | ? |
| 4 | Семенова Л.П. | 15000 | ? | ? |
| 5 | Яковлева С.Т. | 10000 | ? | ? |
| Итого: | | |  | ? |

Уровень 2

1. Постройте и заполните таблицу.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № | Наименование товара | Ед. изм | Кол-во | Цена | Сумма | Ставка НДС | Сумма НДС | Всего с НДС |
| 1 | Кресло | шт. | 6 | 5000 | ? | 20% | ? | ? |
| 2 | Диван | шт. | 4 | 20 500 | ? | 20% | ? | ? |
| 3 | Набор м/мебели | шт. | 2 | 25000 | ? | 20% | ? | ? |
| 4 | Стул | шт. | 200 | 1000 | ? | 20% | ? | ? |
| ВСЕГО К ОПЛATE | | | | | ? |  | ? | ? |
|  | | | | |  |  |  |  |

1. Рассчитать процент успеваемости и процент качества знаний студентов курса.

Процент успеваемости = («отличники»+ «на 4 и 5»+ «троечники») / кол-во учеников\*100

Качество знаний = («отличники»+ «на 4 и 5»)/ кол-во учеников\*100



Уровень 3

1. Постройте и заполните таблицу. Отформатируйте таблицу средствами Автоформата (Цветной 2)

НАЛОГ НА ПРИБЫЛЬ

Ставка налога на прибыль = 35%

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Наименование предприятия | Выручка от реализованной товарной продукции | Себестоимость реализации и товарной продукции | Коммерческие расходы | Управленческие расходы | Прибыль от  реализации | Сумма налога  на прибыль |
| ООО  "Сигма" | 8 580 | 2 340 | 0 | 1 840 | ? | ? |
| ОАО  "Поиск" | 12 620 | 4 560 | 2 360 | 4 210 | ? | ? |
| ЗАО  "Пилот" | 9 430 | 0 | 3 780 | 2 950 | ? | ? |
| ООО  "Стрела" | 7 780 | 0 | 5 200 | 0 | ? | ? |
| ОАО  "Искра" | 14 540 | 5 790 | 4 310 | 3 780 | ? | ? |
| ИТОГО: | ? | ? | ? | ? | ? | ? |

**Практическое занятие № 29. Электронные (динамические) таблицы**

Фильтрация и сортировка данных в диапазоне или таблице. Коллективная работа с данными. *Подключение к внешним данным и их импорт.*

Цель: Использование стандартных функций. Мастер функций. Использование статистических функций в решении задач.

*студент должен знать:*

* правила использования и синтаксис основных статистических функций;
* принципы работы мастера функций;

*студент должен уметь:*

* производить вычисления с использованием категории статистических функций.

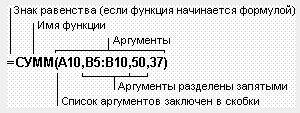
**Теория** Функции

Функции — заранее определенные формулы, которые выполняют вычисления по заданным величинам, называемым аргументами, и в указанном порядке. Эти функции позволяют выполнять как простые, так и сложные вычисления.

В Microsoft Excel используется более 100 функций, объединенных по категориям, например:

* **Функции работы с датой и временем** позволяют анализировать и работать со значениями даты и времени в формулах. Например, если требуется использовать в формуле текущую дату, воспользуйтесь функцией **СЕГОДНЯ**, возвращающей текущую дату по системным часам.
* **Логические функции** предназначены для проверки выполнения условия или для проверки нескольких условий. Так, функция **ЕСЛИ** позволяет определить, выполняется ли указанное условие, и возвращает одно значение, если условие истинно, и другое, — если оно ложно.
* **Математические функции** позволяют производить простые и сложные математические вычисления, например вычисление суммы диапазона ячеек, вычисление суммы ячеек диапазона, удовлетворяющих указанному условию, округление чисел и прочее.
* **Статистические функции** позволяют выполнять статистический анализ диапазонов данных.

Функции задаются с помощью формул, которые выполняют вычисления по заданным величинам, называемым аргументами, и в указанном порядке, называемом синтаксисом. Список аргументов может состоять из чисел, текста, логических величин (ИСТИНА или ЛОЖЬ), массивов, значений ошибок (#Н/Д) или ссылок. Необходимо следить за соответствием типов аргументов. Кроме того, аргументы могут быть как константами, так и формулами. Эти формулы, в свою очередь, могут содержать другие функции.



Написание функции начинается с указания имени функции, затем вводится открывающая скобка, указываются аргументы, отделяющиеся запятыми, а затем — закрывающая скобка. Вставлять функцию в формулу рекомендуется с помощью ***панели формул***. В панели формул отображается имя функции и ее аргументы, описание функции и аргументов, а также возвращаемое функцией и формулой значение.

Рис. 2. Мастер функций

При появлении сообщения **Ошибка в формуле**:

* Проверьте, одинаково ли количество открывающих и закрывающих скобок.
* Проверьте правильность использования оператора диапазона при ссылке на группу ячеек.
* Проверьте, все ли необходимые аргументы введены для функций.
* Если первый символ в имени книги или листа не является буквой, необходимо заключить имя в одинарные кавычки.
* Проверьте, в каждой ли внешней ссылке указано имя книги и полный путь к ней.
* Не изменяйте формат чисел, введенных в формулы. Например, даже если в формулу необходимо ввести 1000 р., то введите число 1000.

**Ошибка #####.** Ошибка появляется, когда вводимое числовое значение или результат выполнения формулы не умещается в ячейке. В этом случае можно увеличить ширину столбца путем перемещения границы, расположенной между заголовками столбцов. Кроме того, можно изменить формат числа ячейки.

**Ошибка #ДЕЛ/0!.** Ошибка появляется, когда в формуле делается попытка деления на ноль. Например, в качестве делителя используется ссылка на ячейку, содержащую нулевое или пустое значение (если операнд является пустой ячейкой, то ее содержимое интерпретируется как ноль), или в формуле содержится явное деление на ноль.

**Ошибка #Н/Д.** Значение ошибки **#Н/Д** является сокращением термина “Неопределенные Данные**”.** Это значение помогает предотвратить использование ссылки на пустую ячейку. Введите в ячейки листа значение **#Н/Д**, если они должны содержать данные, но в настоящий момент эти данные отсутствуют. Формулы, ссылающиеся на эти ячейки, тоже будут возвращать значение **#Н/Д** вместо того, чтобы пытаться производить вычисления. Ошибка может возникнуть, если не заданы один или несколько аргументов стандартной или пользовательской функции, а также задан недопустимый аргумент.

**Ошибка #ИМЯ?.** Ошибка #ИМЯ? появляется, когда Excel не может распознать имя, используемое в формуле. Возможная причина:

* Используемое имя было удалено или не было определено.
* Имеется ошибка в написании имени.
* Имеется ошибка в написании имени функции.
* В формулу введен текст, не заключенный в двойные кавычки.
* В ссылке на диапазон ячеек пропущен знак двоеточия (:).

**Ошибка #ПУСТО!.** Ошибка #ПУСТО! появляется, когда задано пересечение двух областей, которые в действительности не имеют общих ячеек.

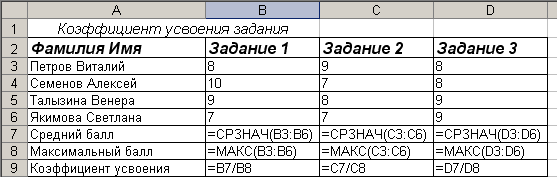
**Ошибка #ССЫЛКА!.** Ошибка #ССЫЛКА! появляется, когда используется недопустимая ссылка на ячейку. Возможно, ячейки, на которые ссылаются формулы, были удалены или в эти ячейки было помещено содержимое других скопированных ячеек.

**Ошибка #ЗНАЧ!.** Ошибка #ЗНАЧ! появляется, когда используется недопустимый тип аргумента или операнда. Например, вместо числового или логического значения введен текст, и Microsoft Excel не может преобразовать его к нужному типу данных.

**Пример:** Рассчитать коэффициент усвоения задания (К).

Коэффициент усвоения задания (К) вычисляется по формуле:

К = Средний балл за данное задание / максимальный балл за данное задание

****

**Упражнения:**

**Уровень 1**

1. Создайте таблицу. Выполните необходимые вычисления.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Месяц | 2014 | 2015 | 2016 | За три года |
| Январь | 37,2 | 34,5 | 8 | ? |
| Февраль | 11,4 | 51,3 | 1,2 | ? |
| Март | 16,5 | 20,5 | 3,8 | ? |
| Апрель | 19,5 | 26,9 | 11,9 | ? |
| Май | 11,7 | 45,5 | 66,3 | ? |
| Июнь | 129,1 | 71,5 | 60 | ? |
| Июль | 57,1 | 152,9 | 50,6 | ? |
| Август | 43,8 | 96,6 | 145,2 | ? |
| Сентябрь | 85,7 | 74,8 | 79,9 | ? |
| Октябрь | 86 | 14,5 | 74,9 | ? |
| Ноябрь | 12,5 | 21 | 56,6 | ? |
| Декабрь | 21,2 | 22,3 | 9,4 | ? |
| Суммарно | ? | ? | ? | ? |
| Максимум | ? | ? | ? | ? |
| Минимум | ? | ? | ? | ? |
| Среднемесячное | ? | ? | ? | ? |

Уровень 2

1. Постройте таблицу: «Крупные водохранилища России», по которой будет видно: суммарная площадь водохранилищ, средний объем водохранилищ, максимальную глубину и минимальный напор водохранилищ.

Средняя глубина Камского водохранилища - 6.5 м. Площадь Горьковского водохранилища – 1400 кв.км. Объем Рыбинского водохранилища – 25 куб.км. Напор Цимлянского водохранилища – 26 м. Площадь Братского водохранилища – 5300 кв.км. Средняя глубина Куйбышевского водохранилища - 10,4 м. Объем Цимлянского водохранилища – 24 куб.км. Площадь Рыбинского водохранилища – 180 куб.км. Площадь Камского водохранилища – 1 700 кв.км. напор Куйбышевского водохранилища – 28 м. Средняя глубина Цимлянского водохранилища - 9,2 м. Напор Камского водохранилища - 21 м. Площадь Куйбышевского водохранилища – 5 000 кв.км. Напор Рыбинского водохранилища – 25 м. Средняя глубина братского водохранилища – 34 м. Объем Куйбышевского водохранилища – 52 куб.км. Напор Горьковского водохранилища – 18 м. Средняя глубина Рыбинского водохранилища – 5,5 м. Объем Камского водохранилища - 11 куб.км. Напор Братского водохранилища – 104 м. Площадь Цимлянского водохранилища – 2600 кв.км.

1. Постройте таблицу «Крупнейшие промышленные корпорации», по которой будет видно: суммарный и средний оборот всех компаний, максимальное и минимальное количество работников.

Компания «Дженирал Моторс» находится в США. Она занимается производством автомобилей. Компания в начале 90-х гг. имела оборот в 102млрд. долларов и 811000 работников. Компания «Тойота мотор» находится в Японии. Она занимается производством автомобилей. Компания в начале 90-х гг. имела оборот в 42 млрд. долларов и 84207 работников. Компания «Ройял Датч-Шелл» занимается производством нефтепродуктов. Компания в начале 90-х гг. имела оборот в 78 млрд. долларов и 133 000 работников. Компания «Тексако» находится в США. Она занимается производством нефтепродуктов. Компания в начале 90-х гг. имела 54481 работника. Компания «Эксон» находится в США. Она занимается производством нефтепродуктов. Компания в начале 90-х гг. имела оборот в 76млрд. долларов и 146000 работников. Компания «Форд Мотор» находится в США. Она занимается производством автомобилей. Компания в начале 90-х гг. имела оборот в 72млрд. долларов и 369300 работников. Копания «Интернешионал бизнес мэшинс» находится в США. Она занимается производством вычислительной техники. Компания в начале 90-х гг. имела оборот в 54млрд. долларов и 403508 работников. Компания «Мобил» находится в США. Она занимается производством нефтепродуктов. Компания в начале 90-х гг. имела оборот в 52млрд. долларов. Компания «Бритиш петролиум» находится в Великобритании. Она занимается производством нефтепродуктов. Компания в начале 90-х гг. имела оборот в 45 млрд. долларов и 126020 работников. Компания «ИРИ» находится в Италии. Она занимается производством металлов. Компания в начале 90-х гг. имела оборот в 41 млрд. долларов и 422000 работников.

**Практическое занятие № 30. Электронные (динамические) таблицы**

Решение вычислительных задач из различных предметных областей.

Цель: Математические функции. Способы представления данных. Виды диаграмм. Решение задач аналитической геометрии средствами Excel. Возможности визуализации различных уравнений. Построение кривых на плоскости.

*студент должен знать:*

* способы обращения к категории математических функций;
* правила использования и синтаксис основных математических функций;
* принципы работы мастера функций; этапы работы мастера диаграмм.

*студент должен уметь:*

* создавать, форматировать и видоизменять диаграммы;
* использовать функции категории математические при решении вычислительных задач.

**Математические функции**

**Функция ОКРУГЛ**

Функция ОКРУГЛ округляет число, задаваемое ее аргументом, до указанного количества десятичных разрядов и имеет следующий синтаксис: =ОКРУГЛ(число;количество\_цифр)

Аргумент число может быть числом, ссылкой на ячейку, в которой содержится число, или формулой, возвращающей числовое значение. Аргумент количество цифр, который может быть любым положительным или отрицательным целым числом, определяет, сколько цифр будет округляться. Задание отрицательного аргумента количество цифр округляет до указанного количества разрядов слева от десятичной запятой, а задание аргумента количество цифр равным 0 округляет до ближайшего целого числа. Excel округляет цифры, которые меньше 5, с недостатком (вниз), а цифры, которые больше или равны 5, с избытком (вверх).

Не смешивайте функцию ОКРУГЛ с фиксированными форматами, например 0 и 0,00, которые имеются на вкладке Число окна диалога Формат ячеек. При использовании этих форматов содержимое в ячейке округляется до заданного числа десятичных разрядов, но при этом изменяется только отображение числа в ячейке, а не само значение. При выполнении вычислений Excel всегда использует хранимое, а не отображаемое значение.

**Функция ПРОИЗВЕД**

Функция ПРОИЗВЕД перемножает все числа, задаваемые ее аргументами, и имеет следующий синтаксис: =ПРОИЗВЕД(число1; число2;...). Функция ПРОИЗВЕД может иметь до 30 аргументов. Excel игнорирует любые пустые ячейки, текстовые и логические значения.

**Функция КОРЕНЬ**

Функция КОРЕНЬ возвращает положительный квадратный корень числа и имеет следующий синтаксис: =КОРЕНЬ(число)

Аргумент число должен быть положительным числом. Например, следующая функция возвращает значение 2: =КОРЕНЬ(4)

Если число отрицательное, КОРЕНЬ возвращает ошибочное значение: =#ЧИСЛО!.

**Функция LOG**

Функция LOG возвращает логарифм положительного числа по заданному основанию. Эта функция имеет следующий синтаксис: =LOG(число; основание')

Например, следующая формула возвращает значение 2,321928095, то есть логарифм 5 по основанию 2: =LOG(5;2) Если вы не укажете аргумент основание, Excel примет его равным 10.

**Функция LN**

Функция LN возвращает натуральный (по основанию е) логарифм положительного числа, указанного в качестве аргумента. Эта функция имеет следующий синтаксис: =LN(число). Например, следующая формула возвращает значение 0,693147181: =LN(2)

**Функция ПИ**

Функция ПИ возвращает значение константы ( с точностью до 14 десятичных знаков: 3,14159265358979. Эта функция имеет следующий синтаксис: =ПИ(). Функция ПИ не имеет аргументов, но мы все равно должны вводить после имени функции круглые скобки.

Обычно функция ПИ используется в других функциях или формулах.

**Функции РАДИАНЫ и ГРАДУСЫ**

Тригонометрические функции используют углы, выраженные в радианах, a не в градусах. Excel предоставляет две функции, РАДИАНЫ и ГРАДУСЫ, чтобы облегчить работу с тригонометрическими функциями. Вы можете преобразовать радианы в градусы, используя функцию ГРАДУСЫ, которая имеет следующий синтаксис: =ГРАДУСЫ(угол). Здесь угол — это число, представляющее собой угол, измеренный в радианах. Для преобразования градусов в радианы используется функция РАДИАНЫ которая имеет следующий синтаксис: =РАДИАНЫ(угол). Здесь угол — это число, представляющее собой угол, измеренный в градусах. Например, следующая формула возвращает значение 180: =ГРАДУСЫ(3,1415927). В то же время следующая формула возвращает значение 3,1415927: =РАДИАНЫ(180)

**Функция SIN**

Функция SIN возвращает синус угла и имеет следующий синтаксис: =SIN(число) Здесь число — это угол в радианах. Например, следующая формула возвращает значение 0,997494987: =SIN(1,5)

**Функция COS**

Функция COS вычисляет косинус угла и имеет следующий синтаксис: =COS(число)

Здесь число — это угол в радианах. Например, следующая формула возвращает значение 0,070737202: =COS(1,5)

Функция TAN

Функция TAN вычисляет тангенс угла и имеет следующий синтаксис: =TAN(число). Здесь число — это угол в радианах. Например, следующая формула возвращает тангенс угла 1,5 радиана, равный 14,10141995: =TAN(1,5)

**Пример. Использование математических функций**

*Задание*. Создать таблицу с использованием математических функций, которая рассчитывает значения функции ***y=sin(*2x/ 3*)\*cos(*x/2*)*** на интервале значений ***х*** от -π до +2 с шагом 0,1, вычисляет максимальное и минимальное значения функции на данном интервале области определения, а также строит график данной функции.

1. Запустите программу Excel и для получения подсказки о синтаксисе тригонометрических функций в окне справки Excel на вкладке **Содержание** выберите раздел *Справка по функциям*, тема *Математические функции*. Для просмотра информации щелкните по ссылкам SIN, COS и ПИ. Для возврата к предыдущему окну справки щелкайте кнопку «Назад».

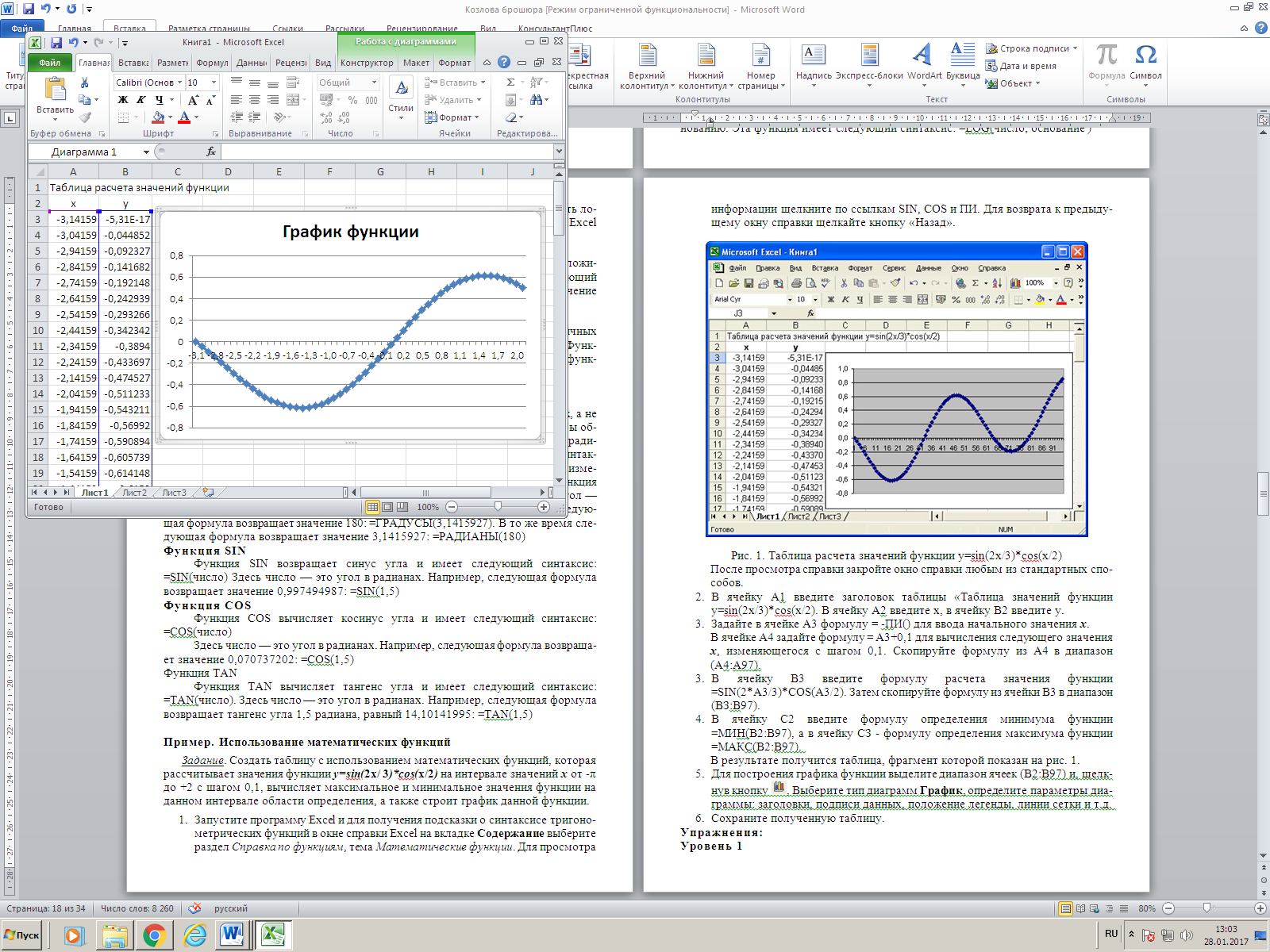


Рис. 3. Таблица расчета значений функции y=sin(2x/3)\*cos(x/2)

После просмотра справки закройте окно справки любым из стандартных способов.

1. В ячейку А1 введите заголовок таблицы «Таблица значений функции y=sin(2x/3)\*cos(x/2). В ячейку А2 введите х, в ячейку В2 введите у.
2. Задайте в ячейке А3 формулу = -ПИ() для ввода начального значения ***х***.

В ячейке A4 задайте формулу = А3+0,1 для вычисления следующего значения ***х***, изменяющегося с шагом 0,1. Скопируйте формулу из A4 в диапазон (А4:А97).

1. В ячейку В3 введите формулу расчета значения функции =SIN(2\*A3/3)\*COS(A3/2). Затем скопируйте формулу из ячейки В3 в диапазон (ВЗ:В97).
2. В ячейку С2 введите формулу определения минимума функции =МИН(В2:В97), а в ячейку СЗ - формулу определения максимума функции =МАКС(В2:В97).

В результате получится таблица, фрагмент которой показан на рис. 1.

1. Для построения графика функции выделите диапазон ячеек (В2:В97) и, щелкнув кнопку p42206, Выберите тип диаграмм **График**, определите параметры диаграммы: заголовки, подписи данных, положение легенды, линии сетки и т.д.
2. Сохраните полученную таблицу.

**Упражнения:**

**Уровень 1**

1. В сельскохозяйственном кооперативе работают 5 сезонных рабочих. Норма сбора овощей составляет N кг. Оплата труда производится по количеству собранных овощей: k рублей за 1 кг. Составить таблицу, содержащую сведения о количестве собранных овощей каждым рабочим и об оплате труда каждого рабочего. Известно, что 1-й рабочий собрал в 3 раза больше нормы; 2-й на 50 кг меньше первого; 3-й - в 1,5 раза больше нормы; 4-й на 75 кг больше третьего; 5-й - на 10 кг больше первого.

Постройте круговую диаграмму, демонстрирующую распределение оплаты труда каждому работнику.

1. Оформите решение квадратного уравнения ax2+bx+c=0, используя ЭТ. Напишите заголовок. Выполните решение для следующих наборов коэффициентов:

a) а = 2, b= - 3, с = - 2;

b) а = 2, b= - 4, с = - 6;

c) а = - 3, b = 1, с = 1.

1. Найдите значение выражения для

а) х = 1, у=1; b) х = 1, у = 0; с) х = 2, у = 1.

1)  2)  3) 

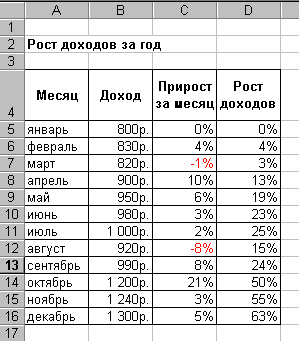
**Практическое занятие № 31. Электронные (динамические) таблицы**

Компьютерные средства представления и анализа данных. Визуализация данных.

Цель: Создание документа, содержащего элементы графического анализа в виде диаграмм и графиков, содержит операции:

* Изменение параметров (ширины) строк и столбцов;
* Работа со списками;
* Использование абсолютных ссылок;
* Построение диаграмм;
* Редактирование диаграмм;
* Построение смешанных диаграмм и графиков;
* Создание комментариев.

Например, требуется создать документ о росте доходов фирмы за год.



Перечислим действия, которые требуется выполнить самостоятельно:

1. Создать новый файл в Excel с именем Доход.
2. Переименовать Лист 1 на Рост доходов.
3. Определить границы листа формата А4.
4. Во второй строке написать заголовок «Рост доходов за год».
5. В четвертой строке сформировать «шапку» документа из четырех столбцов: разместить надписи столбцов с переносом в несколько строк и отцентрировать их по высоте и ширине ячеек

**Заполнение столбца доходы.**

Выделить ячейки В5 – В16 и назначить им финансовый рублевый формат целого типа.

Заполнить эти ячейки соответствующими значениями из документа..

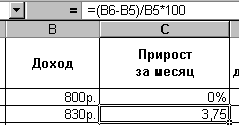
**Заполнение столбца прирост за месяц**

Прирост за месяц – это процент прироста дохода по сравнению с предыдущим месяцем.

За январь прирост будет составлять 0%.

Активизируйте ячейку С6 и составьте формулу для вычисления прироста за месяц в процентном соотношении.

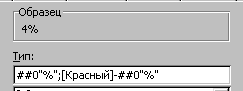
Скопируйте эту формулу для всех последующих месяцев.



*Требование*

Значения округлить до целых, написать знак процента (%) и отрицательные значения отобразить красным цветом.

Реализация указанного требования выполняется составлением формата ячеек, например, следующим образом:



тип: ##0”%”;[Красный]-##0”%”

Назначьте этот формат для всех ячеек интервала С5-С16

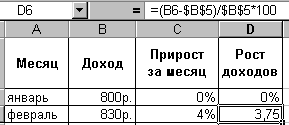
##### Абсолютная адресация

Рост доходов – это процент прироста дохода по сравнению с началом года (январем). За январь рост доходов будет составлять 0%.

Особенность при составлении формулы для столбца Рост доходов заключается в том, что процент роста вычисляется во всех ячейках относительно одного значения - января. Поэтому, чтобы при копировании адрес ячейки В5 (январь) не смещался, необходимо использовать абсолютную адресацию.

Для составления формулы с абсолютной адресацией требуется:

* установить курсор в ячейку D6 и составить формулу вычисления роста доходов относительно начала года (января) в процентах;
* скопировать формулу на остальные ячейки этого столбца;
* назначить такой же формат ячеек, как и для столбца Прирост, за месяц.



В результате выполнения всех действий получится документ, изображенный в начале статьи.

###### Построение диаграммы

Диаграмма - графическое представление данных.

Значение из ячеек рабочего листа, или точки данных, отображаются на экране в виде строк, линий, столбцов, секторов или других компонентов диаграммы.

Точки данных группируются в ряды данных, которые отличаются друг от друга цветом, формой, заполнением.

Построение диаграммы «Изменение величины дохода по месяцам» включает следующие действия:

-выделить ячейки, относящиеся к столбцам Месяц и Доход (А4-В16);

-нажать на пиктограмму Мастер диаграмм  и выполнить определенные шаги действий.

Шаг 1-й. На закладке Стандартные выбрать диаграммы: Гистограмма.

Шаг 2-й. Подтвердить диапазон данных, а ряд данных – в столбцах.

Шаг 3-й. На закладке Заголовки указать название диаграммы Рост доходов за год.

Шаг 4-й. Определить место размещения диаграммы: на том же листе (это так называемые внедренные диаграммы) или на отдельном листе (лист диаграмм). Выбрать на том же листе.

**Важно**! *Независимо от того, какая диаграмма создается – внедренная или лист диаграммы, - она автоматически связывается с тем рабочим листом, на основе которого создана. При изменении исходных данных диаграмма автоматически обновляется.*

**Перемещение и изменение размеров области диаграммы.**

Выделить диаграмму, щелкнув мышкой по краям диаграммы, чтобы появились черные засечки на линиях обрамления. И, не отжимая левой кнопки мыши, перетащить диаграмму к ячейке А18.

Для изменения размеров области диаграммы установить курсор мыши на требуемую засечку и, не отжимая левой кнопки мыши, потянуть в нужном направлении.

**Редактирование диаграммы**.

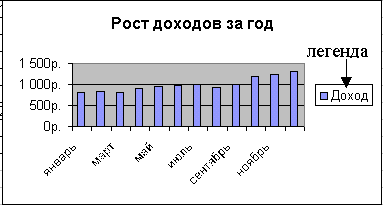
Автоматически сформированная диаграмма явно требует редактирования.

***Правило****:* чтобы отредактировать элемент диаграммы, достаточно дважды щелкнуть по нему мышкой.

**Удаление элемента диаграммы.**

На диаграмме, состоящей из одного ряда данных, область легенды является излишней. Для удаления области легенды необходимо:

* выделить Легенду, нажав мышкой один раз (Окно Легенды должно быть выделено засечками);
* нажать клавишу Del.



**Надписи осей.**

Для формирования надписи для Y требуется:

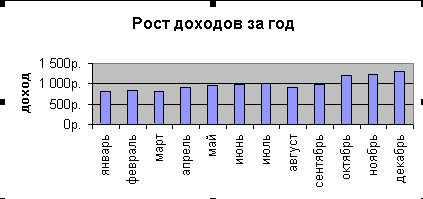
* установить курсор на область Диаграммы и нажать правую кнопку мыши;
* через меню Параметры диаграммы открыть закладку Заголовки и для оси Y задать надпись «Доход».

**Редактирование надписи оси.**

Чтобы все наименования месяцев разместились в области диаграммы, необходимо развернуть их вертикально. Для изменения надписи оси надо:

* установить курсор на любое наименование месяца на диаграмме (например, январь) и дважды щелкнуть левой кнопкой мыши;
* через меню Формат оси перейти на закладку Выравнивание и установить Ориентацию на 90 градусов;
* установить курсор на красную точку, которой заканчивается слово «надпись», и, не отжимая левой кнопки мыши, перетащить на требуемый угол поворота.

После выполнения операций редактирования диаграмма имеет следующий вид:



**Изменение шага значений оси.**

Шаг значений по оси Y равен 500. Требуется уменьшить шаг до 300. Для этого необходимо:

* установить курсор на любое из значений оси Y и дважды щелкнуть по нему левой кнопкой мыши;
* через меню Формат оси выбрать закладку Шкала и в опции Цена основных делений установить значение 300;
* перейти на закладку Шрифт и установить размер цифр – 10.

**Изменение формата рядов данных.**

Для того чтобы ряду данных (столбцам) придать объемный вид и сменить цветовую палитру, требуется:

* нажать курсором по центру одного из столбцов данных, чтобы во всех столбцах диаграммы в центре появились точки;
* через меню Формат выбрать Формат ряда данных или щелкнуть по выделенным столбцам правой кнопкой мыши;
* перейти на закладку Вид и установить параметры: Граница – с тенью, Заливка – голубая.

**Подписи данных.**

Диаграмма станет более наглядной, если каждый столбец будет содержать отображаемые числовые значения. Для этого необходимо:

* выделить весь ряд данных (во всех столбцах диаграммы должны быть точки);
* через меню Формат выбрать Формат ряда данных, затем перейти на закладку Подписи данных и отметь опцию Значение;
* для оформления подписей данных щелкнуть по ним левой кнопкой мыши, при этом обратить внимание, чтобы были выделены все подписи, и через меню Формат выбрать Формат подписей данных;
* перейти на закладку Шрифт и установить параметры: шрифт – Arial Cyr, начертание – полужирный, высота символов – 8;
* перейти на закладку Выравнивание и установить параметры: Ориентация - на 90 градусов, Положение подписи – центр.

**Изменение ширины столбцов диаграммы.**

Чтобы подписи данных размещались в столбцах, необходимо увеличить ширину столбцов:

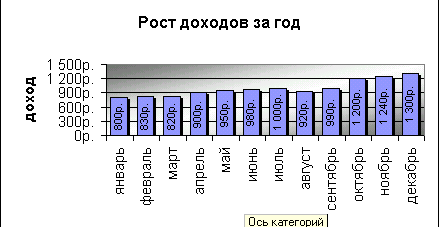
* выделить весь ряд данных (точки должны быть во всех столбцах диаграммы);
* через меню Формат – Выделенный ряд перейти на закладку Параметры и установить ширину зазора – 50.

**Изменение фона построения диаграммы.**

Следующая задача – изменить серый фон, на котором построена диаграмма. Для этого надо:

* выделить область построения диаграммы;
* через меню Формат – Выделенная область построения нажать на пункт Способ заливки, затем на закладке Градиентная указать: один цвет; цвет 1 – белый; тип штриховки – диагональная 1; тип – 4.

В результате диаграмма принимает следующий вид:



Усложним задачу: требуется построить диаграмму « Прирост доходов по месяцам» по данным из двух столбцов – Месяц и Прирост за месяц.

Изменение несмежных интервалов

**Важно!** Чтобы выделить несмежные интервалы, необходимо сначала выделить ячейку в первой строке или столбце, а затем нажать клавишу CTRL и, не отжимая её, выделить остальные ячейки.

Для построения Диаграммы «Прирост доходов по месяцам» необходимо выполнить следующие действия:

* выделить интервалы ячеек А4-А16 и С4-С16
* нажать на пиктограмму Мастер диаграмм.

Шаг 1-й. Выбрать Тип-График, Вид-График с маркерами.

Шаг 2-й. На закладке Диапазон данных - установить курсор в поле Диапазон и выделить мышкой еще раз столбец Прирост за месяц (С4-С16). Перейти на закладку Ряд. Установить курсор в окно Подпись оси Х и выделить мышкой столбец Месяц (А5-А16).

Шаг 3-й. на закладке Заголовки в поле ось Y указать Проценты прироста.

Шаг 4-й. Создать внедренную диаграмму.



Разместить диаграмму под документом, а предыдущую диаграмму перенести вправа.

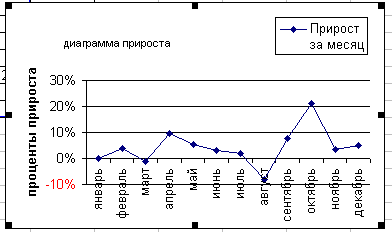
**Требования:**

Сделать надписи месяцев вертикальными.

Изменить заголовок самой диаграммы на заголовок «Диаграмма прироста».

Легенду расположить в верхнем правом углу. Высота текста должна быть 8.

Убрать фон под графиком.



**Добавление новых рядов данных**

На диаграмму прироста необходимо добавить новый ряд данных, соответствующих столбцу Рост доходов.

Вариант 1-й. Выделить вместе с заголовком ячейки, которые образуют новый ряд данных (D6-D16), и, установив курсор на нижнюю границу выделенной области, перетащить на область диаграммы.

Вариант 2-й. Выделить вместе с заголовком ячейки, которые образуют новый ряд данных, скопировать их в буфер обмена (меню Правка – Копировать), затем перейти в область диаграммы и выполнить вставку из буфера (меню Правка – Вставить).

Вариант 3.Выделить область диаграммы, нажать правую кнопку мыши и через меню Исходные данные перейти на закладку Ряд.

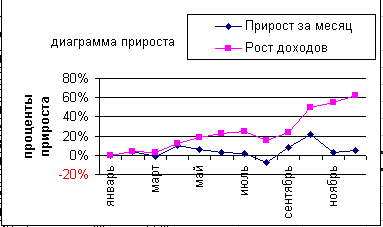
В Окне Ряд нажать кнопку Добавить.

Установить курсор в окно Имя и мышкой указать ячейку, которая является заголовком нового ряда (ячейка D4).

Установить курсор в окно Значения и мышкой указать интервал ячеек, которые являются элементами нового ряда (D5-D16).



После добавления нового ряда диаграмма включает два графика.



##### Сочетание нескольких типов диаграмм в одной диаграмме

Необходимо добавить в эту же диаграмму новый ряд данных из столбца Доход.

Особенность добавления заключается в том, что доход измеряется в других единицах ( в рублях, а не в процентах ) и интервал значений лежит в других пределах (проценты в интервале от 20 до 70, а рубли- от 800 до 1300).

Для получения комбинированной диаграммы необходимо выполнить следующие действия:

* выделить ячейки, относящиеся к столбцу Доход (В4-В16);
* скопировать выделенную область в буфер;
* перейти в область построения диаграммы (область окна диаграммы должна быть выделена засечками);
* выполнить вставку из буфера;
* выделить на диаграмме новый ряд данных, щелкнув мышкой по кривой;
* установить курсор на выделенный ряд, нажать правую кнопку мыши и выбрать в меню Тип диаграммы – с областями.

##### Изменение надписи оси

Для изменения надписи оси необходимо:

* выделить область построения диаграммы;
* нажать правую кнопку мыши, затем через меню Параметры диаграммы перейти на закладку Заголовки и в окне оси Y указать Доход.

##### Добавление вспомогательной оси

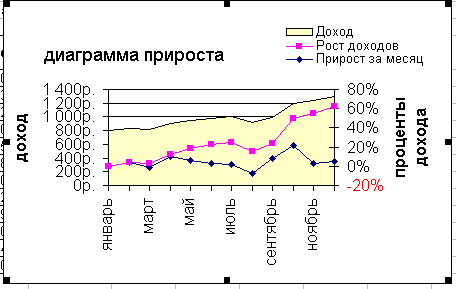
Ось Y отражает теперь величину дохода, поэтому требуется ввести дополнительную ось Y для отображения значений процентов. Для этого следует:

* выделить область построения диаграммы;
* нажать правую кнопку мыши, через меню Параметры диаграммы открыть закладку Оси и в группе Вспомогательная ось отметить ось Y.

Для создания надписи вспомогательной оси надо:

* выделить область построения диаграммы;
* нажать правую кнопку мыши, через меню Параметры диаграммы открыть закладку Заголовки и в окне Вторая ось Y указать Проценты дохода.

Самостоятельно увеличить заголовок диаграммы до 12 пунктов и убрать обрамление вокруг легенды.



##### Панель инструментов диаграммы

Диаграммы можно создавать и редактировать не только с помощью Мастера диаграмм и стандартного меню.

Excel имеет специальную панель инструментов для работы с диаграммами.

Выделить область построения диаграмм на рабочем листе и выполнить операции редактирования, используя панель инструментов диаграммы. Для этого следует:

* заменить основные линии сетки на пунктирные линии;
* изменить шкалу значений по оси Y с шагом 400;
* оформить ряд Доход градиентной заливкой;
* оформить текст Легенды курсивом;
* поместить заголовок диаграммы в рамку с тенью, а текст заголовка написать синим цветом.

**Практическое занятие № 32. Базы данных**

Понятие и назначение базы данных (далее – БД). Классификация БД. Системы управления БД (СУБД).

Цель: изучить Интерфейс. Организация баз данных.

Студент должен знать:

Основные определения и понятия: БД, реляционные БД, основные структурные компоненты, поле, ключевое поле, запись, типы данных, сортировку.

Студент должен уметь:

Создавать таблицы в режиме конструктора,

### Теория. Общие сведения

Access - это *система управления базами данных* (СУБД). Под *системой управления* понимается комплекс программ, который позволяет не только хранить большие массивы данных в определенном формате, но и обрабатывать их, представляя в удобном для пользователей виде. Access дает возможность также автоматизировать часто выполняемые операции (например, расчет заработной платы, учет материальных ценностей и т.п.). С помощью Access можно не только разрабатывать удобные формы ввода и просмотра данных, но и составлять сложные отчеты.

Access является приложением Windows, а поскольку и Windows и Access разработаны одной фирмой (Microsoft), они очень хорошо взаимодействуют друг с другом. СУБД Access работает под управлением Windows; таким образом, все преимущества Windows доступны в Access, например, вы можете вырезать, копировать и вставлять данные из любого приложения Windows в приложение Access и наоборот.

Access - это реляционная СУБД. Это означает, что с ее помощью можно работать одновременно с несколькими таблицами базы данных. Применение реляционной СУБД помогает упростить структуру данных и таким образом облегчить выполнение работы. Таблицу Access можно связать с данными, хранящимися на другом компьютере или на сервере, а также использовать таблицу, созданную в СУБД Paradox или Dbase. Данные Access очень просто комбинировать с данными Excel.

В СУБД Access предусмотрено много дополнительных сервисных возможностей. *Мастера* помогут вам создать таблицы, формы или отчеты из имеющихся заготовок, сделав за вас основную черновую работу. *Выражения* используются в Access, например, для проверки допустимости введенного значения. *Макросы* позволяют автоматизировать многие процессы без программирования, тогда как встроенный в Access язык *VBA* (Visual Basic for Applications) - специально разработанный компанией Microsoft диалект языка Basic для использования в приложениях Microsoft Office - дает возможность опытному пользователю программировать сложные процедуры обработки данных. Просматривая свою форму или отчет, вы сможете представить, как они будут выглядеть в распечатанном виде. И наконец, используя такие возможности *языка программирования С,* как функции и обращения к Windows API (Application Programming Interface - интерфейс прикладных программ Windows), можно написать подпрограмму для взаимодействия Access с другими приложениями - источниками данных.

В Microsoft Access добавлено множество новых средств, разработанных для облегчения работы в Интернет и создания приложений для Web. Для доступа к сети Интернет и использования преимуществ новых средств необходимы средства просмотра Web, например Microsoft Internet Explorer, а также модем. Пользователь имеет возможность непосредственно подключаться к узлам Microsoft Web из программ Office (в том числе и из Access) с помощью команды **Microsoft на Web** из пункта меню **?**. При этом можно, например, получить доступ к техническим ресурсам и загрузить общедоступные программы, не прерывая работу с Access.

Система Access содержит набор инструментов для управления базами данных, включающий конструкторы таблиц, форм, запросов и отчетов. Кроме того, Access можно рассматривать и как среду для разработки приложений. Используя макросы для автоматизации задач, вы можете создавать такие же мощные, ориентированные на пользователя приложения, как и приложения, созданные с помощью "полноценных" языков программирования, дополнять их кнопками, меню и диалоговыми окнами. Программируя на VBA, можно создавать программы, по мощности не уступающие самой Access. Более того, многие средства Access, например *мастера* и *конструкторы,* написаны на VBA. Мощность и гибкость системы Access делают ее сегодня одной из лучших программ для управления базами данных.

### Работа с мастерами

*Мастер* (Wizard) - специальная программа, помогающая в решении какой-то задачи или создании объекта определенного типа. Эта программа поможет вам за несколько минут выполнить рутинную работу, на которую без применения этой программы может уйти несколько часов. Программа-мастер задает вопросы о содержании, стиле и формате объекта, а затем создает этот объект без какого-либо вмешательства с вашей стороны. В Access имеется около сотни *мастеров,* предназначенных для проектирования баз данных, приложений, таблиц, форм, отчетов, графиков, почтовых наклеек, элементов управления и свойств.

### Информация для пользователей электронных таблиц

Те, кто знакомы с Excel, заметят, что Access во многом похож на Excel. Прежде всего, обе программы являются продуктами для Windows, следовательно, можно использовать свой опыт применения специфичных для Windows соглашений. Данные таблицы или запроса Access отображаются в виде электронной таблицы, которую принято называть *таблицей данных.* Вы обнаружите, что размер строк и столбцов таблицы данных можно изменять так же, как в рабочих таблицах Excel. Фактически режим ввода данных Access ничем не отличается от аналогичного режима Excel. Основное различие между таблицей базы данных (БД) и электронной таблицей - в системе адресации; в электронной таблице адресуется каждая ячейка, а в таблице БД - только поля текущей записи. В электронной таблице каждая ячейка обрабатывается индивидуально, а в таблице БД обработка идет по записям, причем записи обрабатываются однотипным образом. Эти упрощения для БД позволяют повысить скорость обработки и количество обслуживаемой информации.

### Контекстно-зависимая справка и Помощник

Справочная система фирмы Microsoft является, наверное, лучшей среди аналогичных программ как для новичков, так и для опытных пользователей. Access дает возможность использовать контекстно-зависимую справку, для получения которой достаточно нажать правую клавишу мыши. Какие бы вы ни испытывали затруднения при работе с системой, вам поможет появляющаяся на экране справка по интересующей вас теме. Помимо этого справочная система Access имеет удобные и простые в использовании содержание, предметный указатель, систему поиска, журнал хронологии и закладки. В локализованной версии Access (как и во всем Microsoft Office ) компания Microsoft добавила новое средство - *Помощник.* Помощник отвечает на вопросы, выдает советы и справки об особенностях используемой программы.

### Структура таблицы и типы данных

Все составляющие базы данных, такие, как таблицы, отчеты, запросы, формы и объекты, в Access хранятся в едином дисковом файле. Основным структурным компонентом базы данных является таблица. В таблицах хранятся вводимые нами данные. Внешне каждая таблица Access похожа на таблицы, с которыми мы привыкли работать на бумаге, - она состоит из столбцов, называемых *полями,* и строк, называемых *записями.* Каждая запись таблицы содержит всю необходимую информацию об отдельном элементе базы данных. Например, запись о преподавателе может содержать фамилию, имя, отчество, дату рождения, должность и т.п.

При разработке структуры таблицы, прежде всего, необходимо определить названия полей, из которых она должна состоять, типы полей и их размеры. Каждому полю таблицы присваивается уникальное имя, которое не может содержать более 64 символов. Имя желательно делать таким, чтобы функция поля узнавалась по его имени. Далее надо решить, данные какого типа будут содержаться в каждом поле. В Access можно выбирать любые из основных типов данных. Один из этих типов данных должен быть присвоен каждому полю. Значение типа поля может быть задано только в режиме конструктора. Ниже представлены типы данных Access и их описание.

|  |  |
| --- | --- |
| **Тип данных** | **Описание** |
| Текстовый (Значение по умолчанию) | Текст или числа, не требующие проведения расчетов, например номера телефонов (до 255 знаков) |
| Числовой | Числовые данные различных форматов, используемые для проведения расчетов |
| Дата/время | Для хранения информации о дате и времени с 100 по 9999 год включительно |
| Денежный | Денежные значения и числовые данные, используемые в математических расчетах, проводящихся с точностью до 15 знаков в целой и до 4 знаков в дробной части |
| Поле MEMO | Для хранения комментариев; до 65535 символов |
| Счетчик | Специальное числовое поле, в котором Access автоматически присваивает уникальный порядковый номер каждой записи. Значения полей типа счетчика обновлять нельзя |
| Логический | Может иметь только одно из двух возможных значений (TrueFalse, Да/Нет) |
| Поле объекта OLE | Объект (например, электронная таблица Microsoft Excel, документ Microsoft Word, рисунок, звукозапись или другие данные в двоичном формате), связанный или внедренный в таблицу Access |
| Гиперссылка | Строка, состоящая из букв и цифр и представляющая адрес гиперссылки. Адрес гиперссылки может состоять максимум из трех частей: текст, выводимый в поле или в элементе управления; путь к файлу (в формате пути UNC) или к странице (адрес URL). Чтобы вставить адрес гиперссылки в поле или в элемент управления, выполните команду Вставка, Гиперссылка |
| Мастер подстановок | Создает поле, в котором предлагается выбор значений из списка или из поля со списком, содержащего набор постоянных значений или значений из другой таблицы. Это в действительности не тип поля, а способ хранения поля |

В Access существует четыре способа создания пустой таблицы:

* использование мастера баз данных для создания всей базы данных, содержащей все требуемые отчеты, таблицы и формы, за одну операцию. Мастер баз данных создает новую базу данных, его нельзя использовать для добавления новых таблиц, форм, отчетов в уже существующую базу данных;
* мастер таблиц позволяет выбрать поля для данной таблицы из множества определенных ранее таблиц, таких, как деловые контакты, список личного имущества или рецепты;
* ввод данных непосредственно в пустую таблицу в режиме таблицы. При сохранении новой таблицы в Access данные анализируются, и каждому полю присваивается необходимый тип данных и формат;
* определение всех параметров макета таблицы в режиме конструктора.

Независимо от метода, примененного для создания таблицы, всегда имеется возможность использовать режим конструктора для дальнейшего изменения макета таблицы, например для добавления новых полей, установки значений по умолчанию или для создания масок ввода. Однако только четвертый метод позволяет сразу задать ту структуру таблицы, которая вам нужна, и поэтому далее рассмотрим именно этот метод.

### Ввод данных в ячейки таблицы

Ввод данных в ячейки таблицы производится обычным образом - вы переводите курсор в нужную ячейку, т.е. делаете ее активной, и вводите в нее данные (вводить данные в ячейки для поля с типом данных *Счетчик* не требуется, так как значения в этом поле появляются автоматически при вводе данных в любую другую ячейку данной записи). Особенности ввода следующие: при нажатии клавиши <Del> ячейка очищается; если ввод данных в ячейку прервать, нажав клавишу <Esc>, то восстанавливается старое значение, а если нажать клавиши <Enter> или <Таb>, то в ячейку заносится новое значение. Можно редактировать текущее значение, меняя в нем отдельные символы. Для этого надо в ячейке таблицы дважды щелкнуть мышью или нажать клавишу <F2>. Однако есть одна важная особенность при вводе данных - для некоторых типов данных (числовой, денежный, дата/время, логический) Access автоматически проверяет правильность их ввода. Например, если вы введете букву в ячейку с числовым типом, то Access выдаст сообщение о неправильно введенном значении и не позволит перейти к другой ячейке, пока не будут введены правильные данные.

Для всех типов полей (кроме типов *Счетчик* и поле объекта OLE) можно самостоятельно задавать ограничения для вводимых данных. Для этого в режиме конструктора надо выбрать вкладку *Общие,* перевести курсор в поле с именем "Условия на значение" и ввести ограничение на данные. Например, стаж работы вряд ли может быть более 70 лет, и ограничение будет следующее: <70. Ограничение можно вводить и на текстовое поле, Обычно в этом случае задаются слова, которые могут присутствовать в данном поле.

***Примечание****.* Вводить ограничения можно не только вручную, но и с помощью *Построителя выражений.* При появлении курсора в строке параметра *Условия на значение* справа от нее появится кнопка с тремя точками. При щелчке по этой кнопке появляется окно построителя выражений (рис. 1.1). Обычно им пользуются для построения сложных выражений, но разобраться в его работе довольно просто при помощи справки, расположенной в этом же окне.

Можно использовать еще один удобный инструмент при вводе данных - параметр *Значение по умолчанию* (находится также на вкладке *Общие).* Здесь можно задать данные, которые Access будет вводить по умолчанию при заполнении таблицы. Это удобно использовать, когда большинство значений данного поля одинаковы и лишь некоторые отличаются. Например, должность большинства преподавателей - доцент. Если ввести это слово в строку параметра *Значение по умолчанию,* то все ячейки поля "Должность" примут значение *Доцент* и надо будет лишь изменить значения ячеек для преподавателей с другими должностями. Данные можно вводить и копированием их из одной ячейки в другую стандартными средствами Windows.

Для каждого типа данных (кроме типа данных *Счетчик)* предусмотрено пустое (нулевое) значение. Различают два типа пустых значений: пустые (Null) значения и пустые строки, Если пустое поле имеет пустое (Null) значение, то это означает, что данные для него существуют, но пока не известны. Если же введена пустая строка (два знака прямых кавычек (" ")), то это означает, что данные не существуют вовсе. Access позволяет обрабатывать такие пустые значения.

### Редактирование данных

Редактирование данных производится очень просто - курсор переводится в нужную ячейку, старые данные удаляются (клавишами <Del> или <Backspasе>) и вводятся новые данные. Удалять можно не только данные в ячейках, но и целиком строки, предварительно их выделив. Но если таблица большая, то редактируемые данные надо сначала найти. Это делается следующим образом; переводят курсор в любую ячейку поля, начиная с которой будет вестись поиск, затем открывают пункт меню **Правка** и выполняют команду **Найти.** В появившемся окне вводят образец искомых данных и щелкают по кнопке <Найти>. Если значение найдено, курсор перейдет в эту ячейку. Иногда требуется большое количество одинаковых данных заменить на другое значение. Для этого надо открыть пункт меню **Правка** и выполнить команду **Заменить**. В появившемся окне ввести образцы того, что надо найти и на что заменить. Затем надо решить, будете ли вы просматривать каждое заменяемое значение или нет. Если будете, то надо щелкнуть по кнопке <3аменить>, а если нет - один раз щелкнуть по кнопке <3аменить все>.

### Сортировка данных

Дня удобства просмотра можно сортировать записи в таблице в определенной последовательности, например, в таблице *Преподаватели* записи можно отсортировать в порядке убывания стажа преподавателей. Кнопки сортировки на панели инструментов (или команды меню **Записи, Сортировка, Сортировка по возрастанию (сортировка по убыванию) )** позволяют сортировать столбцы по возрастанию или по убыванию. Прежде чем щелкнуть по кнопке сортировки, следует выбрать поля, используемые для сортировки. Для выбора поля достаточно поместить курсор в любую его запись. После этого щелкните по кнопке сортировки - и данные отобразятся в отсортированном порядке, В режиме таблицы можно выделить сразу два или несколько соседних столбцов, а затем выполнить по ним сортировку. По умолчанию в Access сортировка записей начинается с крайнего левого выделенного столбца. При этом записи таблицы будут отсортированы сначала по крайнему левому выделенному столбцу, затем (для одинаковых значений в первом сортируемом столбце) - по второму и т.д. Если вы хотите восстановить порядок отображения записей, используйте команду **Записи, Удалить фильтр.**

***Примечание****.* Современные СУБД такие, как Access , никогда не сортируют таблицы физически, как это делалось раньше. Средства сортировки данных (а также фильтрации, поиска и замены) реализованы в Access как автоматически создаваемые запросы. Записи таблицы всегда располагаются в файле базы данных в том порядке, в котором они были добавлены в таблицу.

### Отбор данных с помощью фильтра

Фильтр - это набор условий, применяемых для отбора подмножества записей. В Access существуют фильтры четырех типов; фильтр по выделенному фрагменту, обычный фильтр, расширенный фильтр и фильтр по вводу.

*Фильтр по выделенному фрагменту -* это способ быстрого отбора записей по выделенному образцу. Например, вам нужно просмотреть в таблице записи только о доцентах, Выделите слово *Доцент* в любой из записей. Щелкните по кнопке <Фильтр по выделенному> (или выполните команду **Записи, Фильтр, Фильтр по выделенному**), и Access выберет только те записи, для которых значение в столбце Должность равно *Доцент.* Обратите внимание, что в строке состояния окна таблицы присутствует слово *Фильтр.* В дополнение к этому кнопка <Применить фильтр> (третья кнопка фильтрования, на которой изображена воронка) затенена, а это означает, что используется фильтр. При отключении этой кнопки все фильтры будут сняты. Установки фильтра не пропадут; он просто будет отключен. Фильтр по выделенному может собирать вместе критерии выбора при каждом использовании кнопки <Фильтр по выделенному>. Например, вы поместили курсор в столбец Должность в записи, в которой значение поля равно *Доцент,* и щелкнули по кнопке <Фильтр по выделенному>. Вы увидите только записи обо всех доцентах. Если затем вы поместите курсор в столбец <Должность> и выделите слово *Информатика* (конечно, если такая дисциплина присутствует в вашей таблице), а затем щелкните по кнопке <Фильтр по выделенному> появятся только записи о доцентах, которые преподают информатику. Если вы хотите просмотреть значения, которые не удовлетворяют этому критерию, например просмотреть всех преподавателей, кроме доцентов, читающих информатику, щелкните правой кнопкой (курсор мыши должен находиться внутри таблицы) и выберите команду **Исключить выделенное**. Будут выбраны все записи, кроме недавно выбранных *(инверсный*  выбор).

Фильтрование данных в Access производится с помощью кнопок <Фильтр по выделенному> или <Изменить фильтр> (команда меню **Записи, Фильтр, Изменить фильтр).** После нажатия второй кнопки от таблицы остается одна запись. Каждое поле становится полем со списком (когда в нем находится курсор), в котором можно выбрать из списка всезначения для данного поля. После щелчка по кнопке <Применить фильтр> будут выбраны записи, соответствующие измененному фильтру. Еще более сложные условия фильтрации можно задать командой меню **Записи, Фильтр, Расширенный фильтр**...

**Практическое занятие № 33. Базы данных**

Таблицы. Запись и поле. Ключевое поле. Типы данных.

Цель: Изучить возможности управления базой данных на примере однотабличного макета.

Студент должен знать:

Основные определения и понятия: БД, реляционные БД, основные структурные компоненты, поле, ключевое поле, запись, типы данных, сортировку.

Студент должен уметь:

Создавать таблицы в режиме конструктора,

### Задание 1

### Создание базы данных.

1. Создайте новую базу данных.

2. Создайте таблицу базы данных.

3. Определите поля таблицы в соответствии с табл. 1.

4. Сохраните созданную таблицу.

Таблица.1. Таблица данных *Преподаватели*

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Размер поля |
| Код преподавателя | Счетчик |  |
| Фамилия | Текстовый | 15 |
| Имя | Текстовый | 15 |
| Отчество | Текстовый | 15 |
| Дата рождения | Дата/время | Краткий |
| Должность | Текстовый | 9 |
| Дисциплина | Текстовый | 11 |
| Телефон | Текстовый | 9 |
| Зарплата | Денежный |  |

### ТЕХНОЛОГИЯ РАБОТЫ

Для создания новой базы данных:

* загрузите Access, в появившемся окне выберите пункт **Новая база данных;**
* в окне "Файл новой базы данных" задайте имя вашей базы (пункт **Имя Файла)** и выберите папку (пункт **Папка**), где ваша база данных будет находиться. По умолчанию Access предлагает вам имя базы db1, а тип файла - *Базы данные Access.* Имя задайте *Преподаватели,* а тип файла оставьте прежним, так как другие типы файлов нужны в специальных случаях;
* щелкните по кнопке <Создать>.

Для создания таблицы базы данных:

* в окне базы данных выберите вкладку *Таблицы,* а затем щелкните по кнопке <Создать>;
* в окне "Новая таблица" выберите пункт **Конструктор** и щелкните по кнопке <ОК>. В результате проделанных операций открывается окно таблицы в режим конструктора, в котором следует определить поля таблицы.

Для определения полей таблицы:

* введите в строку столбца Имя поля имя первого поля *Код преподавателя;*
* в строке столбца "Тип данных" щелкните по кнопке списка и выберите тип данных *Счетчик.* Поля вкладки *Общие* оставьте такими, как предлагает Access.

***Примечание.*** Заполнение строки столбца "Описание" необязательно и обычно используется для внесения дополнительных сведений о поле.

Для определения всех остальных полей таблицы базы данных *Преподаватели* в соответствии с табл.1 выполните действия, аналогичные указанным выше.

***Внимание!*** Обратите внимание на вкладку *Общие* в нижней части экрана. Советуем изменить данные в пункте **Размер поля, а** остальные пункты оставить по умолчанию (их функции рассмотрим далее). Например, для текстового типа данных Access предлагает по умолчанию длину 50 символов. Но вряд ли поле "Фамилия" будет содержать более 15 символов, хотя лучше точно подсчитать, сколько символов в самой длинной фамилии. Не бойтесь ошибиться - в дальнейшем можно скорректировать длину поля. Для числового типа Access предлагает *Длинное целое,* но ваши данные могут быть либо небольшие целые числа (в диапазоне от -32768 до 32767) - тогда надо выбрать *Целое,* либо дробные числа - тогда надо выбрать *С  плавающей точкой..* Для выбора необходимого параметра надо щелкнуть по полю, а затем нажать появившуюся кнопку списка и выбрать необходимые данные. В результате ваша таблица будет иметь более компактный вид, а объем базы данных уменьшится.

4. Для сохранения таблицы:

* выберите пункт меню **Файл, Сохранить;**
* в диалоговом окне "Сохранение" введите имя таблицы *Преподаватели',*
* щелкните по кнопке <ОК>.

***Примечание.*** В результате щелчка по кнопке <ОК> Access предложит вам задать ключевое поле (поле первичного ключа), т.е. поле, однозначно идентифицирующее каждую запись. Для однотабличной базы данных это не столь актуально, как для многотабличной, поэтому щелкните по кнопке <Нет>.

### ЗАДАНИЕ 2

### Заполнение базы данных.

1. Введите ограничения на данные, вводимые в поле "Должность"; должны вводиться только слова *Профессор, Доцент* или *Ассистент*.

2. Задайте текст сообщения об ошибке, который будет появляться на экране при вводе неправильных данных в поле "Должность".

3. Задайте значение по умолчанию для поля "Должность" в виде слова *Доцент.*

4. Введите ограничения на данные в поле <Код>; эти данные не должны повторяться.

5. Заполните таблицу данными в соответствии с табл.2 и проверьте реакцию системы на ввод неправильных данных в поле "Должность".

6. Измените ширину каждого поля таблицы в соответствии с шириной данных.

7. Произведите поиск в таблице преподавателя Миронова.

8. Произведите сортировку данных в поле "Дата рождения" по убыванию,

9. Произведите фильтрацию данных по полям "Должность" и "Дисциплина".

Просмотрите созданную таблицу, как она будет выглядеть на листе бумаги при печати.

**Таблица 2**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Код | Фамилия | Имя | Отчество | Дата рожд. | Должность | Дисциплина | Телефон | Зарплата |
| 1 | Истомин | Ремир | Евгеньевич | 23.10.74 | Доцент | Информатика | 10-44-68 | 22890р. |
| 2 | Миронов | Павел | Юрьевич | 25.07.60 | Профессор | Экономика | 12-21-40 | 50200р. |
| 3 | Гришин | Евгений | Сергеевич | 05.12.77 | Доцент | Математика | 60-23-65 | 25760р. |
| 4 | Сергеева | Ольга | Ивановна | 12.02.82 | Ассистент | Математика | 34-85-69 | 15450р. |
| 5 | Емец | Татьяна | Ивановна | 16.02.71 | Доцент | Экономика | 66-75-33 | 22890р. |
| 6 | Игнатьева | Татьяна | Павловна | 30.05.76 | Доцент | Информатика | 10-36-98 | 23790р. |
| 7 | Миронов | Алексей | Николаевич | 30.07.78 | Доцент | Физика | 66-75-33 | 23890р. |

### ТЕХНОЛОГИЯ РАБОТЫ

Для задания условия на значение для вводимых данных:

* войдите в режим *Конструктор* для проектируемой таблицы. Если вы находитесь в окне базы данных, то выберите вкладку *Таблицы* и щелкните по кнопке <Конструктор>. Если вы находитесь в режиме таблицы, то щелкните по кнопке на панели инструментов или выполните команду **Вид, Конструктор;**
* в верхней части окна щелкните по полю "Должность";
* в нижней части окна щелкните по строке параметра *Условие на значение;*
* щелкните по кнопке для определения условий на значение при помощи построителя выражений;
* в появившемся окне напишите слово *Профессор,* затем щелкните по кнопке (эта кнопка выполняет функцию ИЛИ), напишите *Доцент,* снова щелкните по этой же кнопке, напишите *Ассистент* и щелкните по кнопке <ОК>. Таким образом, вы ввели условие, при котором в поле "Должность" могут вводиться только указанные значения.

2. В строке *Сообщение об ошибке* введите предложение "Такой должности нет, правильно введите данные".

3. В строке *Значение по умолчанию* введите слово "Доцент".

4. Введите ограничения на данные в поле "Код". Здесь ограничения надо вводить не совсем обычным способом. Дело в том, что коды преподавателей не должны повторяться, а также должна быть обеспечена возможность их изменения (из-за последнего условия в этом поле нельзя использовать тип данных *Счетчик,* в котором данные не повторяются). Для выполнения второго условия пришлось задать в поле "Код" тип данных *Числовой,* а для выполнения первого условия сделайте следующее:

* щелкните по строке параметра *Индексированное поле*;

***Примечание.*** Индекс - это средство Access, ускоряющее поиск и сортировку данных в таблице. Ключевое поле (поле первичного ключа) таблицы индексируется автоматически. Не допускается создание индексов для полей типа *MEMO* и *Гиперссылка vim* полей объектов *OLE.* Свойство *Индексированное поле* определяет индекс, создаваемый по одному полю. Индексированное поле может содержать как уникальные, так и повторяющиеся значения. Допускается создание произвольного количества индексов.

* выберите в списке пункт **Да (совпадения не допускаются);**

перейдите в режим **Таблица,** щелкнув по кнопке на панели инструментов или выполнив команду **Вид, Режим таблицы.** На вопрос о сохранении таблицы щелкните по кнопке <Да>.

5. Введите данные в таблицу в соответствии с табл. *2.* Попробуйте в поле <Должность> любой записи ввести слово *Лаборант.* Посмотрите, что получилось. На экране должно появиться сообщение; "Такой должности нет, правильно введите данные". Введите правильное слово.

6. Для изменения ширины каждого поля таблицы в соответствии с шириной данных:

* щелкните в любой строке поля "Код";
* выполните команду **Формат, Ширина столбца;**
* в появившемся окне щелкните по кнопке <По ширине данных>. Ширина поля изменится;
* проделайте эту операцию с остальными полями.

7. Для поиска в таблице преподавателя Миронова:

* переведите курсор в первую строку поля "Фамилия";
* выполните команду **Правка, Найти;**
* в появившейся строке параметра *Образец* введите *Миронов;*
* в строке параметра *Просмотр* должно быть слово *ВСЕ* (имеется в виду искать по всем записям);
* в строке параметра *Совпадение* выберите из списка *С любой частью поля;*
* в строке параметра *Только в текущем поле* установите флажок (должна стоять галочка);
* щелкните по кнопке <Найти>. Курсор перейдет на вторую запись и выделит слово *Миронов*;
* щелкните по кнопке <Найти далее>. Курсор перейдет на седьмую запись и также выделит слово *Миронов;*
* щелкните по кнопке <3акрыть> для выхода из режима поиска.

8. Для сортировки данных в поле "Дата рождения" по убыванию:

* щелкните по любой записи поля "Дата рождения";
* щелкните по кнопке на панели управления или выполните команду **Записи, Сортировка, Сортировка** **по убыванию.** Все данные в таблице будут отсортированы в соответствии с убыванием значений в поле "Дата рождения".
* Для фильтрации данных по полям "Должность" и "Дисциплина":
* щелкните по записи *Доцент* поля "Должность";
* щелкните по кнопке или выполните команду **Записи, Фильтр,** **Фильтр по выделенному**. В таблице останутся только записи о преподавателях - доцентах;
* щелкните по записи *Информатика* поля "Дисциплина";
* щелкните по кнопке или выполните команду **Записи, Фильтр, Фильтр по** **выделенному.** В таблице останутся только записи о преподавателях - доцентах кафедры информатики;
* для отмены фильтрации щелкните по кнопке на панели инструментов или выполните команду **Записи, Удалить фильтр.** В таблице появятся все данные.

Для просмотра созданной таблицы:

* щелкните по кнопке или выполните команду **Файл, Предварительный, просмотр.** Вы увидите таблицу как бы на листе бумаги;
* закройте окно просмотра.

***Примечание.*** Если вы захотите изменить поля или ориентацию таблицы на листе бумаги, выполните команду **Файл, Параметры страницы.** В открывшемся окне можете изменять указанные параметры.

**Практическое занятие № 34-35. Базы данных**

Запрос. Типы запросов. Запросы с параметрами. Сортировка. Фильтрация. Вычисляемые поля.

Цель: Изучить способы выбора данных из таблиц с помощью запросов, создавать запросы на выборку, на обновление и удаление,

Студент должен знать:

* Основные определения и понятия: БД, реляционные БД, основные структурные компоненты, поле, ключевое поле, запись, типы данных, сортировку.
* Окно конструктора запросов.

Студент должен уметь:

* Выбирать данные из таблиц с помощью запросов, создавать запросы на выборку, на обновление и удаление,

### Формирование запросов на выборку

Запросы являются мощным средством обработки данных, хранимых в таблицах Access . С помощью запросов можно просматривать, анализировать и изменять данные из нескольких таблиц. Они также используются в качестве источника данных для форм и отчетов. Запросы позволяют вычислять итоговые значения и выводить их в компактном формате, подобном формату электронной таблицы, а также выполнять, вычисления над группами записей.

Запросы можно создавать самостоятельно и с помощью *мастеров. Мастера запросов* автоматически выполняют основные действия в зависимости от ответов пользователя на поставленные вопросы. Самостоятельно разработать запросы можно в режиме **конструктора**.

В Access можно создавать следующие типы запросов:

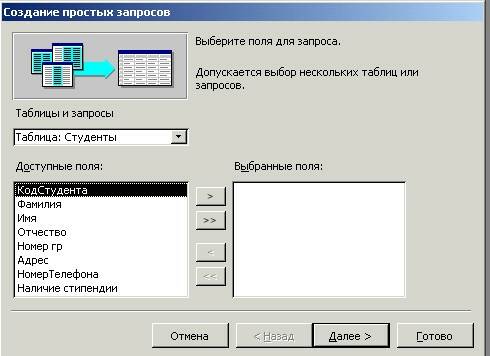
* запрос на выборку;
* запрос с параметрами;
* перекрестный запрос;
* запрос на изменение (запрос на удаление, обновление и добавление записей на создание таблицы);
* запросы SQL (запросы на объединение, запросы к серверу, управляющие запросы, подчиненные запросы)

*Запрос на выборку* используется наиболее часто. При его выполнении данные, удовлетворяющие условиям отбора, выбираются из одной или из нескольких таблиц и выводятся в определенном порядке. Например, можно вывести на экран данные о фамилиях доцентов, стаж которых более 15 лет (на основе таблицы *Преподаватели).*

***Примечание.*** Простые запросы на выборку практически не отличаются от фильтров. Более того, фильтры можно сохранять как запросы.

Можно также использовать запрос на выборку, чтобы сгруппировать записи для вычисления сумм, средних значений, пересчета и других действий. Например, используя запрос на выборку, можно получить данные о среднем стаже доцентов и профессоров (на основе таблицы *Преподаватели).*

*Запрос с параметрами* - это запрос, при выполнении которого в его диалоговом окне пользователю выдается приглашение ввести данные, на основе которых будет выполняться запрос. Например, часто требуются данные о том, какие дисциплины ведут преподаватели. Чтобы не создавать отдельные запросы по каждому преподавателю, можно создать один запрос с параметрами, где в качестве параметра будет использоваться фамилия преподавателя. При каждом вызове этого запроса вам будет предложено ввести фамилию преподавателя, а затем на экран будут выведены все поля, которые вы указали в запросе, например фамилия, имя, отчество преподавателя и читаемая им дисциплина.



**Рис.** Окно выбора вариантов построения запросов

Для создания нового запроса надо в окне базы данных выбрать вкладку *Запросы* и щелкнуть по кнопке <Создать>. Откроется окно "Новый запрос", вид которого представлен на рис. 2.8.

В окне вы должны выбрать один из пяти пунктов: Конструктор, Простой запрос, Перекрестный запрос, Повторяющиеся записи. Записи без подчиненных. *Конструктор* позволит вам самостоятельно создать любой тип запроса, но этот режим рекомендуется пользователям, уже имеющим некоторый опыт создания запросов. *Простой запрос* позволит создать с помощью *Мастера* запрос на выборку из определенных полей таблиц или других запросов. (Это наилучший способ создания запроса для начинающих пользователей.) Что такое *перекрестный запрос,* рассматривалось выше. При выборе пункта *Повторяющиеся записи* будет создан запрос на поиск повторяющихся записей в простой таблице или в запросе, а при выборе пункта *Записи без подчиненных -* запрос на поиск записей, которым не соответствует ни одна запись в подчиненной таблице. Такой запрос используется для многотабличных баз данных.

У вас может возникнуть вопрос: как создавать запросы с параметрами и запросы на изменение, если при создании запроса они явно не указаны? Следует отметить, что основой для всех этих запросов является запрос на выборку, т.е. сначала необходимо определить набор данных, с которым хотите работать. Затем для созданного запроса на выборку надо перейти в режим конструктора. Задание параметров производится в строке *Условия отбора* для соответствующих полей. Подробнее это будет рассмотрено ниже при выполнении задания. Для доступа к запросам на изменение надо открыть пункт меню **Запрос** - в открывшемся списке вы увидите все виды запросов на изменение.

При выполнении запроса на выборку Access извлекает записи из таблиц и формирует результирующий набор данных. Он выглядит, как таблица, хотя и не является ею. Результирующий набор данных является *динамическим* (или *виртуальным)* набором записей и не хранится в базе данных.

После закрытия запроса результирующий набор данных этого запроса прекращает свое существование. Хотя сам по себе динамический набор данных больше не существует, помните, что данные, которые в нем содержались, остаются в базовых таблицах.

При сохранении запроса остается только структура запроса - перечень таблиц, список полей, порядок сортировки, ограничения на записи, тип запроса и т.д.

При сохранении в базе данных запрос, по сравнению с результирующим набором данных, имеет ряд преимуществ:

на физическом носителе информации (обычно это жесткий диск) требуется меньший объем пространства;

запрос может использовать обновленные версии любых записей, измененных со времени последнего запуска запроса.

При каждом выполнении запрос обращается к базовым таблицам и снова создает результирующий набор данных. Поскольку сам по себе результирующий набор данных не сохраняется, запрос автоматически отображает любые изменения, происшедшие в базовых таблицах с момента последнего запуска этого запроса (даже в реальном времени в многопользовательской среде).

Для сохранения запроса следует выполнить следующие действия. Выполните команду **Файл, Сохранить** или щелкните по кнопке <Сохранить> на панели инструментов. Если вы впервые сохраняете запрос, введите новое имя запроса в диалоговом окне "Сохранение".

### ЗАДАНИЕ

### Формирование запросов на выборку.

1. На основе таблицы *Преподаватели* создайте простой запрос на выборку, в котором должны отображаться фамилии, имена, отчества преподавателей и их должность.

2. Данные запроса отсортируйте по должностям.

3. Сохраните запрос.

4. Создайте запрос на выборку с параметром, в котором должны отображаться фамилии, имена, отчества преподавателей и преподаваемые ими дисциплины, а в качестве параметра задайте фамилию преподавателя и выполните этот запрос для преподавателя *Гришина.*

### ТЕХНОЛОГИЯ РАБОТЫ

1. Для создания простого запроса:

* в окне базы данных откройте вкладку *Запросы;*
* в открывшемся окне щелкните по кнопке <Создать>;
* из появившихся пунктов окна "Новый запрос" выберите **Простой запрос** и щелкните по кнопке <ОК>;
* в появившемся окне в строке *Таблицы/запросы* выберите таблицу *Преподаватели* (если других таблиц или запросов не было создано, она будет одна в открывающемся списке);
* в окне "Доступные поля" переведите выделение на параметр *Фамилия,*
* щелкните по кнопке. Слово *Фамилия* перейдет в окно "Выбранные поля";
* аналогично в окно "Выбранные поля" переведите поля "Имя", "Отчество", "Должность" (порядок важен - в таком порядке данные и будут выводиться);
* щелкните по кнопке. <Далее>;
* в строке параметра *Задайте имя запроса* введите новое имя Д*олжности преподавателей;*
* щелкните по кнопке <Готово>. На экране появится таблица с результатами запроса.

2. Для сортировки данных:

* щелкните в любой строке поля "Должность";
* отсортируйте данные по убыванию. Для этого щелкните по кнопке http://www.stu.ru/inform/praktika/Pweb4/42.htm9.jpg на панели инструментов или выполните команду **Записи, Сортировка, Сортировка** **по** **убыванию.**

3. Для сохранения запроса:

* щелкните по кнопке http://www.stu.ru/inform/praktika/Pweb4/42.htm10.jpg или выполните команду **Файл, Сохранить;**
* закройте окно запроса.

4. Для создания запроса на выборку с параметром:

* создайте запрос на выборку для следующих полей таблицы *Преподаватели:* "Фамилия", "Имя", "Отчество", "Преподаваемая дисциплина". Запрос создавайте аналогично тому, как это делалось в п.1;
* задайте имя запросу *Преподаваемые дисциплины;*
* щелкните по кнопке <Готово>. На экране появится таблица с результатами запроса;
* перейдите в режиме конструктора, щелкнув по кнопке или выполнив команду **Вид, Конструктор;**
* в строке параметра *Условия отбора* для поля "Фамилия" введите фразу (скобки тоже вводить): [*Введите фамилию преподавателя*]
* выполните запрос, щелкнув по кнопке http://www.stu.ru/inform/praktika/Pweb4/42.htm11.jpg на панели инструментов или выполнив команду **Запрос, Запуск;**

***Примечание.*** Вышеописанным способом запрос выполняется только в режиме конструктора. Для того чтобы выполнить запрос из другого режима, надо открыть вкладку *Запросы,* выделить требуемый запрос и щелкнуть по кнопке <Открыть>.

* в появившемся окне введите фамилию *Гришин* и щелкните по кнопке <ОК>. На экране появится таблица с данными о преподавателе *Гришине -* его имя, отчество и преподаваемая им дисциплина;
* сохраните запрос;
* закройте окно запроса.

**Практическое занятие № 36. Базы данных**

*Формы. Отчеты.*

### ЗАДАНИЕ

### На основе таблицы *Преподаватели* создайте отчет с группированием данных по должностям.

### ТЕХНОЛОГИЯ РАБОТЫ

Для создания отчета:

* откройте вкладку *Отчеты* и щелкните по кнопке <Создать>;
* в открывшемся окне выберите пункт Мастер отчетов;
* щелкните по значку раскрывающегося списка в нижней части окна;
* выберите из появившегося списка таблицу *Преподаватели;*
* щелкните по кнопке **<**OK>, В появившемся окне выберите поля, которые будут присутствовать в отчете. В данном примере присутствовать будут все поля из таблицы, поэтому щелкните по кнопке >>;
* щелкните по кнопке <Далее>;
* в появившемся окне присутствует перечень полей. Переведите выделение на поле "Должность";
* щелкните по кнопке. Таким образом вы задаете группировку данных по должности;
* щелкните по кнопке <Далее>;
* параметры появившегося окна оставим без изменений, поэтому щелкните по кнопке <Далее>;
* в появившемся окне выберите стиль оформления отчета;
* щелкните по кнопке. <Далее>;
* в появившемся окне введите название отчета *Преподаватели;*
* щелкните по кнопке <Готово>. На экране появится сформированный отчет;
* просмотрите, а затем закройте отчет.

**Практическое занятие № 37. Базы данных**

Многотабличные БД. Связи между таблицами. *Нормализация*.

Цель: *Проектирование многотабличной базы данных. Установка связей между таблицами.*

Студент должен знать:

* Основные определения и понятия: БД, реляционные БД, основные структурные компоненты, поле, ключевое поле, запись, типы данных, сортировку.

Студент должен уметь:

* Создавать таблицы в режиме конструктора,

### Организация данных

Слово "реляционная" происходит от английского *relation -* отношение. *Отношение -* тематическое понятие, но в терминологии моделей данных отношения удобно изображать в виде таблицы. При этом строки таблицы соответствуют кортежам отношения, а столбцы - атрибутам. Ключом называют любую функцию от атрибутов кортежа, которая может быть использована для идентификации кортежа. Такая функция может быть значением одного, из атрибутов (простой ключ), задаваться алгебраическим выражением, включающим значения нескольких атрибутов (составной ключ). Это означает, что данные в строках каждого из столбцов составного ключа могут повторяться, но комбинация данных каждой строки этих столбцов является уникальной. Например, в таблице Студенты есть столбцы *Фамилии* и *Год рождения.* В каждом из столбцов есть некоторые повторяющиеся данные, т.е. одинаковые фамилии и одинаковые года рождения. Но если студенты, имеющие одинаковые фамилии, имеют разные года рождения, то эти столбцы можно использовать в качестве составного ключа. Как правило, ключ является уникальным, т.е. каждый кортеж определяется значением ключа однозначно, но иногда используют и неуникальные ключи (ключи с повторениями). В локализованной (русифицированной) версии Access вводится термин *ключевое поле,* которое можно трактовать как *первичный ключ.*

В Access можно выделить три типа ключевых полей: простой ключ, составной ключ и внешний ключ.

Одно из важнейших достоинств реляционных баз данных состоит в том, что вы можете хранить логически сгруппированные данные в разных таблицах и задавать связи между ними, объединяя их в единую базу. Для задания связи таблицы должны иметь поля с одинаковыми именами или хотя бы с одинаковыми форматами данных. Связь между таблицами устанавливает отношения между совпадающими значениями в этих полях. Такая организация данных позволяет уменьшить избыточность хранимых данных, упрощает их ввод и организацию запросов и отчетов. Поясним это на примере. Допустим, вам в базе надо хранить, данные о студентах (фамилия, изучаемая дисциплина) и преподавателях (фамилия, номер кафедры, ученая степень, преподаваемая дисциплина). Если хранить данные в одной таблице, то в строке с фамилией студента, изучающего конкретную дисциплину, будут храниться все атрибуты преподавателя, читающего эту дисциплину. Это же огромная избыточность данных. А если хранить данные о студенте в одной таблице, о преподавателе - в другой и установить связь между полями "Читаемая дисциплина" - "Изучаемая дисциплина" (фактически это одинаковые поля), то избыточность хранимых данных многократно уменьшится без ущерба для логической организации информации.

В Access можно задать три вида связей между таблицами; *Один-ко-многим,, Многие-ко-многим* и *Один-к-одному.*

Связь *Один-ко-многим -* наиболее часто используемый тип связи между таблицами. В такой связи каждой записи в таблице А может соответствовать несколько записей в таблице В (поля с этими записями называют *внешними ключами),* а запись в таблице В не может иметь более одной соответствующей ей записи в таблице А.

При связи *Многие-ко-многим* одной записи в таблице А может соответствовать несколько записей в таблице В, а одной записи в таблице В - несколько записей в таблице А. Такая схема реализуется только с помощью третьей (связующей) таблицы, ключ которой состоит по крайней мере из двух полей, одно из которых является общим с таблицей А, а другое - общим с таблицей В.

При связи *Один-к-одному* запись в таблице А может иметь не более одной связанной записи в таблице В и наоборот. Этот тип связи используют не очень часто, поскольку такие данные могут быть помещены в одну таблицу. Связь с отношением *Один-к-одному* применяют для разделения очень широких таблиц, для отделения части таблицы в целях ее защиты, а также для сохранения сведений, относящихся к подмножеству записей в главной таблице.

Тип создаваемой связи зависит от полей, для которых определяется связь:

* связь *Один-ко-многим* создается в том случае, когда только одно из полей является ключевым или имеет уникальный индекс, т.е. значения в нем не повторяются;
* связь *Один-к-одному* создается в том случае, когда оба связываемых поля являются ключевыми или имеют уникальные индексы;
* связь *Многие-ко-многим* фактически представляет две связи типа *один-ко-многим* через третью таблицу, ключ которой состоит, по крайней мере, из двух полей, общих для двух других таблиц.

### Целостность данных

Целостность данных означает систему правил, используемых в СУБД Access для поддержания связей между записями в связанных таблицах (таблиц, объединенных с помощью связи), а также обеспечивает защиту от случайного удаления или изменения связанных данных. Контролировать целостность данных можно, если выполнены следующие условия:

* связанное поле (поле, посредством которого осуществляется связь) одной таблицы является ключевым полем или имеет уникальный индекс;
* связанные поля имеют один тип данных. Здесь существует исключение. Поле счетчика может быть связано с числовым полем, если оно имеет тип *Длинное целое,*
* обе таблицы принадлежат одной базе данных Access. Если таблицы являются связанными, то они должны быть таблицами Access. Для установки целостности данных база данных, в которой находятся таблицы, должна быть открыта. Для связанных таблиц из баз данных других форматов установить целостность данных невозможно.

### ЗАДАНИЕ 1

### Создание инфологической и логической моделей базы данных.

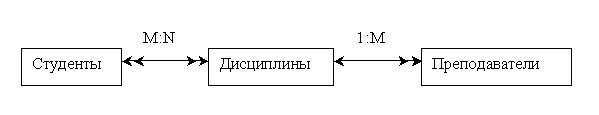
1. Разработайте информационно-логическую модель реляционной базы данных.

2. Разработайте логическую модель реляционной базы данных

### ТЕХНОЛОГИЯ РАБОТЫ

1.Перед разработкой информационно-логической модели реляционной базы данных рассмотрим, из каких информационных объектов должна состоять эта база данных. Можно выделить три объекта, которые не будут обладать избыточностью, *- Студенты, Дисциплины* и *Преподаватели.* Представим состав реквизитов этих объектов в виде "название объекта (перечень реквизитов)": *Студенты* (код студента, фамилия, имя, отчество, номер группы, дата рождения, стипендия, оценки). *Дисциплины* (код дисциплины, название дисциплины), *Преподаватели* (код преподавателя, фамилия, имя, отчество, дата рождения, телефон, заработная плата).

Рассмотрим связь между объектами *Студенты* и *Дисциплины.* Студент изучает несколько дисциплин, что соответствует многозначной связи и отражено на рис. 3.9 двойной стрелкой. Понятно, что каждая дисциплина изучается множеством студентов. Это тоже многозначная связь, обозначаемая двойной стрелкой (связь "один" обозначена одинарной стрелкой). Таким образом, связь между объектами *Студенты* и *Дисциплины - Многие-ко-многим* (М : N).



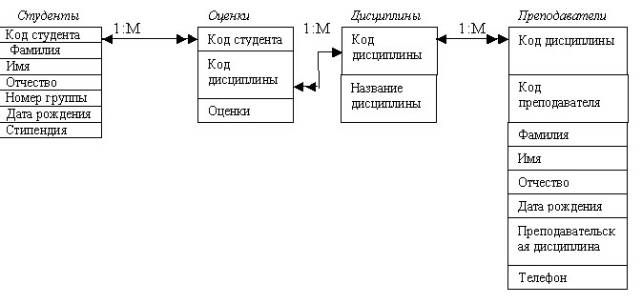
**Рис** Типы связей между объектами *Студенты, Дисциплины и Преподаватели.*

Множественные связи усложняют управление базой данных, например, в СУБД Access при множественных связях нельзя использовать механизм каскадного обновления. Поэтому использовать такие связи нежелательно и нужно строить реляционную модель, не содержащую связей типа *Многие-ко-многим.* В Access дня контроля целостности данных с возможностью каскадного обновления и удаления данных необходимо создать вспомогательный объект связи, который состоит из ключевых реквизитов связываемых объектов и который может быть дополнен описательными реквизитами. В нашем случае таким новым объектом для связи служит объект *Оценки,* реквизитами которого являются код студента, код дисциплины и оценки. Каждый студент имеет оценки по нескольким дисциплинам, поэтому связь между объектами *Студенты* и *Оценки* будет *Один-ко-многим* (1:М). Каждую дисциплину сдает множество студентов, поэтому связь между объектами *Дисциплины* и *Оценки* также будет *Один-ко-многим* (1:М). В результате получаем информационно-логическую модель базы данных, приведенную на рисунке.

## 43

**Рис.** Информационно-логическая модель реляционной базы данных

2.В реляционной базе данных в качестве объектов рассматриваются отношения, которые можно представить в виде таблиц. Таблицы между собой связываются посредством общих полей, т.е. одинаковых по форматам и, как правило, по названию, имеющихся в обеих таблицах. Рассмотрим, какие общие поля надо ввести в таблицы для обеспечения связности данных. В таблицах *Студенты* и *Оценки* таким полем будет "Код студента", в таблицах *Дисциплины* и *Оценки -* "Код дисциплины", в таблицах *Преподаватели* и *Дисциплины -* "Код дисциплины". Выбор цифровых кодов вместо фамилий или названий дисциплин обусловлен меньшим объемом информации в таких полях: например, число "2". по количеству символов значительно меньше слова "математика". В соответствии с этим логическая модель базы данных  представлена на рисунке где жирными буквами выделены ключевые поля.



**Рис.** Логическая модель базы данных

### ЗАДАНИЕ 2

### Создание реляционной базы данных.

1.Создайте базу данных *Деканат.*

2.Создайте структуру таблицы *Студенты.*

3.Создайте структуру таблицы *Дисциплины.*

4. Измените структуру таблицы *Преподаватели.*

5. Создайте структуру таблицы *Оценки.*

6.Разработайте схему данных, т.е. создайте связи между таблицами.

### ТЕХНОЛОГИЯ РАБОТЫ

1. Создайте базу *данных Деканат,* выполнив следующие действия:

* загрузите Access, в появившемся окне выберите пункт *Новая база данных,* затем щелкните по кнопке <ОК>;
* в окне <Файл новой базы данных> задайте имя (пункт *Имя файла)* и выберите папку (пункт *Папка),* где ваша база будет находиться. По умолчанию Access предлагает имя базы *db1,* а тип файла - *Базы данных Access.* Имя задайте *Деканат,* а тип файла оставьте прежним, так как другие типы файлов нужны в специальных случаях;
* щелкните по кнопке <Создать>

2. Создайте структуру таблицы *Студенты.* Для этого:

* в окне базы данных выберите вкладку *Таблицы,* а затем щелкните по кнопке <Создать>;
* в окне "Новая таблица" выберите пункт *Конструктор* и щелкните по кнопке <ОК>. В результате проделанных операций открывается окно таблицы в режиме конструктора, в котором следует определить поля таблицы;
* определите поля таблицы в соответствии с таблицей:

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Размер поля |
| Код студента | Числовой | Целое |
| Фамилия | Текстовый | 15 |
| Имя | Текстовый | 12 |
| Отчество | Текстовый | 15 |
| Номер группы | Числовой | Целое |
| Телефон | Текстовый | 9 |
| Стипендия | Логический | Да/Нет |

* в качестве ключевого поля задайте "Код студента". Для этого щелкните по полю "Код студента" и по кнопке    на панели инструментов или выполните команду **Правка, Ключевое поле**;
* закройте таблицу, задав ей имя *Студенты.*

***Примечание****.*Заполнять таблицу данными пока не требуется, это будет сделано в режиме формы.

3. Создайте структуру *таблицы Дисциплины* аналогично п. 2 в соответствии с таблицей.

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Размер поля |
| Код дисциплины | Числовой | Целое |
| Название дисциплины | Текстовый | 30 |

В качестве ключевого поля задайте "Код дисциплины". Заполняться эта таблица будет также в режиме формы.

4. Структура таблицы *Преподаватели* уже создана в работе 1 и заполнена данными, этому для работы используйте эту таблицу с одним лишь изменением - в структуру таблицы надо добавить поле "Код дисциплины" и заполнить его в соответствии с данными табл.

5. Создайте структуру таблицы *Оценки* аналогично п. 2 в соответствии с табл.

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Размер поля |
| Код студента | Числовой | Целое |
| Код дисциплины | Числовой | Целое |
| Оценки | Числовой | Байт |

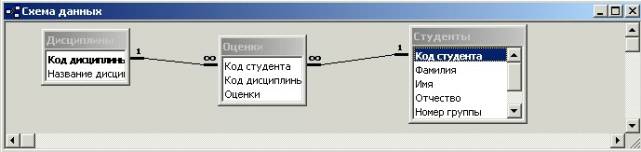
В этой таблице задавать ключевое поле не надо, так как данные во всех полях могут повторяться. Эта таблица, аналогично предыдущим, будет заполняться в режиме формы.

6. Разработайте схему данных, т.е. создайте связи между таблицами. Для этого:

* щелкните по кнопке на панели инструментов или выполните команду **Сервис, Схема данных**. На экране появится окно "Схема данных";
* щелкните по кнопке на панели инструментов или выполните команду **Связи, Добавить таблицу**;
* в появившемся окне будет выделено название одной таблицы. Щелкните по кнопке <Добавить>;
* переведите выделение на имя следующей таблицы и щелкните по кнопке <Добавить>. Аналогично добавьте оставшиеся две таблицы;
* закройте окно, щелкнув по кнопке <3акрыть>;
* создайте связь между таблицами *Дисциплины* и *Оценки.* Для этого подведите курсор мыши к полю "Код дисциплины" в таблице *Дисциплины*  щелкните левой кнопкой мыши и, не отпуская ее, перетащите курсор на поле "Код дисциплины" в таблицу *Оценки,* а затем отпустите кнопку мыши. На экране откроется окно "Связи";
* установите флажок ("галочку") в свойстве *Обеспечение целостности данных,* щелкнув по нему;
* установите флажок в свойстве *Каскадное обновление связанных полей* и *Каскадное удаление связанных записей;*

***Примечание.*** Задание каскадного обновления связанных полей и каскадного удаления связанных записей позволит вам отредактировать записи только в *таблице Дисциплины,* а в таблице *Оценки* эти действия будут со связанными записями выполняться автоматически. Например, если вы удалите из таблицы *Дисциплины* один предмет, то в таблице *Оценки* удалятся все строки, связанные с этим предметом.

* щелкните по кнопке <Создать>. Связь будет создана;
* аналогично создайте связи между полем "Код дисциплины" в *таблице Дисциплины* и полем "Код дисциплины" в таблице *Преподаватели,* а также между полем "Код студента" в таблице *Студенты* и полем "Код студента" в таблице *Оценки.* Результат представлен на рис. 12;
* закройте окно схемы данных, ответив *ДА* на вопрос о сохранении макета.



**Рис.** Структура таблицы *Студенты*

**Практическое занятие № 38 Заполнение таблиц данными с помощью построения Форм. Создание Формы в режиме Конструктор.**

Цель:Заполнение таблиц данными с помощью построения Форм. Создание Формы в режиме Конструктор.

### ЗАДАНИЕ

### Создание форм для ввода данных в таблицы.

1. Создайте форму *Студенты.*

2. Заполните данными таблицу *Студенты* посредством формы *Студенты.*

3. Создайте *форму Дисциплины.*

4. Заполните данными *таблицу Дисциплины* посредством *формы Дисциплины.*

5. Создайте форму *Оценки.*

6. Заполните данными таблицу *Оценки* посредством формы *Оценки.*

### ТЕХНОЛОГИЯ РАБОТЫ

1. Для создания формы *Студенты:*

* откройте вкладку *Формы;*
* щелкните по кнопке <Создать>;
* в открывающемся списке выберите таблицу *Студенты,*
* выберите пункт Автоформа: ленточная;
* щелкните по кнопке <ОК>. Форма для ввода данных создана;

***Примечание.*** Если вас не удовлетворяет макет, вы можете перейти в режим конструктора и изменить макет, передвигая и изменяя размеры элементов - заголовков полей и ячеек для ввода данных. Достаточно щелкнуть по элементу - он выделяется прямоугольной рамкой, и вы можете изменять размеры и двигать элемент. Если вы хотите изменить другие параметры элемента, надо, по выделенному элементу щелкнуть правой клавишей мыши, и откроется окно свойств элемента. В силу ограниченности объема раздела описать все свойства нет возможности, но их можно изучить самостоятельно по справочной системе, а многие свойства понятны уже из своего названия.

2. Заполните данными, приведенными в таблицу *Студенты* посредством формы.

Таблица

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Код студента | Фамилия | Имя | Отчество | Номер группы | Телефон | Стипендия |
| 1 | Арбузов | Николай | Николаевич | 151 | 64-45-78 | Да |
| 2 | Киршин | Петр | Валерьевич | 151 | 25-45-89 | Да |
| 3 | Кривинский | Сергей | Николаевич | 151 | 45-55-52 | Нет |
| 4 | Крылова | Елена | Петровна | 151 | 64-56-88 | Да |
| 5 | Кульчий | Григорий | Викторович | 151 | 25-45-12 | Да |
| 6 | Патрикеев | Олег | Борисович | 152 | 56-52-56 | Нет |
| 7 | Перлов | Кирилл | Николаевич | 152 | 55-55-57 | Нет |
| 8 | Соколова | Наталия | Петровна | 152 | 20-45-32 | Нет |
| 9 | Степанская | Ольга | Витальевна | 152 | 45-78-23 | Да |
| 10 | Тимофеев | Сергей | Трофимович | 152 | 56-23-89 | Да |

***Примечание.*** Переход между ячейками лучше выполнять клавишей <Таb> либо мышью. Существуют и другие варианты перехода по строкам или полям спомощью различных клавиш и их комбинаций. Обычно их используют опытные пользователи, не любящие работать с мышью.

Закройте форму, задав ей имя *Студенты.*

3. Создайте форму Дисциплины аналогично п.1.

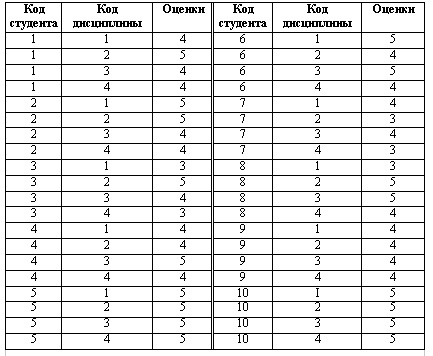
4. Заполните данными, таблицу Дисциплины посредством, формы и закройте форму, задав ей имя Дисциплины.

|  |  |
| --- | --- |
| Код дисциплины | Название |
| 1 | Информатика |
| 2 | Математика |
| 3 | Физика |
| 4 | Экономика |

5. Создайте форму Дисциплины аналогично п. 1.

6. Заполните данными таблицу Оценки. Посредством формы закройте форму, задав ей имя Оценки.

Таблица Оценки



**Практическое занятие № 39. Подготовка и выполнение исследовательского проекта**

Технология выполнения исследовательского проекта: постановка задачи, выбор методов исследования, составление проекта и плана работ, подготовка исходных данных, проведение исследования, формулировка выводов, подготовка отчета. Верификация (проверка надежности и согласованности) исходных данных и валидация (проверка достоверности) результатов исследования. Статистическая обработка данных. Обработка результатов эксперимента.

[Этапы выполнения исследовательской работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Fetapy)

І этап. Подготовка к исследовательской работе (проекту)

1. Найди проблему – то, что на твой взгляд хочешь изучить и исследовать.

2. Назови свое исследование, т.е. определи тему исследовательской работы;

3. Опиши [актуальность исследовательской работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Faktualnost), т.е. обоснуй выбор именно этой темы работы;

4. Сформулируй [цель исследовательской работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Fcel-raboty) и поэтапно распиши [задачи исследовательской работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Fzadachi);

5. Выбери оптимальный вариант решения проблемы;

6. Составь вместе с учителем план работы для реализации своего исследовательского проекта.

ІІ этап. Планирование исследовательской работы

1. Определись, где планируешь искать и найти информацию;

2. Определись со способами сбора и анализа информации, т.е. каким образом, в какой форме и кто будет собирать, выбирать и анализировать информацию;

3. Выбери способ представления результатов работы, т.е. в какой форме будет твой отчет (текстовое описание работы, присутствие диаграмм, презентации, фотографий процесса исследования или эксперимента, аудио- или видео-записи наблюдений, опытов, этапов эксперемента и конечного результата);

4. Установи критерии оценки (как будешь оценивать) хода эксперимента, исследования, полученного результата исследовательской работы (исследовательского проекта);

5. Распредели задачи и обязанности между учащимися в группе, если это групповой проект.

ІІІ этап. Исследование (процесс исследования, эксперимента)

1. Собери необходимую информацию для проведения исследования, при необходимости, проведи расчеты, замеры, подбери качественный и безопасный материал и инструменты для эксперимента и т.д.

2. Проведи то, что запланировал: интервью, опросы, наблюдения, эксперименты, опыты, необходимую работу.

ІV этап. Выводы

1. Проведи анализ полученной в ходе исследовательской работы информации;

2. Дай экономико-экологическое обоснование (затратно, экономически выгодно, экологично ли выполнение твоей исследовательской работы);

3. Сформулируй выводы (добился ли того, что ставил в цели и задачах).

V этап. Отчет и защита работы

1. Оформи и подготовь представление результатов своей работы: защиту в виде устного отчета, устный отчета с демонстрацией, письменного отчета и краткой устной защиты с презентацией;

2. Проведи защиту своей исследовательской работы (проекта) и прими участие в возможном обсуждении, давай четкие ответы на возникшие вопросы.

VІ этап. Оценка процесса и результатов работы  
1. Поучаствуй в оценке исследовательской работы путем коллективного обсуждения и самооценки.

Если Вам необходимо создать Творческий проект, то полную информацию об оформлении, требованиях и правилах написания такого вида работы Вы найдете на сайте [Творческие проекты и работы учащихся](https://infourok.ru/go.html?href=http%3A%2F%2Ftvorcheskie-proekty.ru%2F), который также как и сайт Обучонок всесторонне поможет Вам.

[План исследовательской работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Fplan)

1. [Титульный лист исследовательской работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Ftitulniy-list)

2. [Содержание исследовательской работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Fsoderjanie)

3. [Введение исследовательской работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Fvvedenie)

Во Введении исследовательской работы обосновывается актуальность выбранной темы, определяются объект, предмет исследования и основные проблемы, формулируется цель и содержание поставленных задач, сообщается, в чем состоит новизна исследования (если имеется).

В этой главе определяются методы исследования, обосновывается теоретическая и практическая значимость(если есть практическая часть) работы.

Структура Введения исследовательской работы:

* [Актуальность исследовательского проекта](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Faktualnost)
* [Объект и предмет исследования](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Fnode%2F425)
* [Цель исследовательской работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Fcel-raboty)
* [Задачи исследовательской работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Fzadachi)
* [Методы исследовательской работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Fmetody)
* [Теоретическая значимость работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Fnode%2F430)
* [Практическая значимость работы](https://infourok.ru/go.html?href=http%3A%2F%2Fobuchonok.ru%2Fznachimost)

**Практическое занятие № 40. *Системы искусственного интеллекта и машинное обучение***

*Машинное обучение – решение задач распознавания, классификации и предсказания. Искусственный интеллект. Анализ данных с применением методов машинного обучения. Экспертные и рекомендательные системы.*

*Большие данные в природе и технике* *(геномные данные, результаты физических экспериментов, интернет-данные, в частности данные социальных сетей). Технологии их обработки и хранения.*

**Теоретическая часть:**

Машинное обучение ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) machine learning, ML) — класс методов [искусственного интеллекта](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D0%BB%D0%BB%D0%B5%D0%BA%D1%82), характерной чертой которых является не прямое решение задачи, а обучение в процессе применения решений множества сходных задач. Для построения таких методов используются средства [математической статистики](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%81%D1%82%D0%B0%D1%82%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B0), [численных методов](https://ru.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D1%8B), [математического анализа](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7), [методов оптимизации](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D1%82%D0%B8%D0%BC%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), [теории вероятностей](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B2%D0%B5%D1%80%D0%BE%D1%8F%D1%82%D0%BD%D0%BE%D1%81%D1%82%D0%B5%D0%B9), [теории графов](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2), различные техники работы с [данными в цифровой форме](https://ru.wikipedia.org/wiki/%D0%94%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5_(%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D0%BA%D0%B0)).

Различают два типа обучения:

1. [Обучение по прецедентам](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BF%D0%BE_%D0%BF%D1%80%D0%B5%D1%86%D0%B5%D0%B4%D0%B5%D0%BD%D1%82%D0%B0%D0%BC), или индуктивное обучение, основано на выявлении [эмпирических закономерностей](https://ru.wikipedia.org/wiki/%D0%AD%D0%BC%D0%BF%D0%B8%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%B7%D0%B0%D0%BA%D0%BE%D0%BD%D0%BE%D0%BC%D0%B5%D1%80%D0%BD%D0%BE%D1%81%D1%82%D1%8C) в [данных](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D0%B1%D0%BE%D1%80%D0%BA%D0%B0).
2. Дедуктивное обучение предполагает формализацию знаний экспертов и их перенос в компьютер в виде [базы знаний](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B7%D0%BD%D0%B0%D0%BD%D0%B8%D0%B9).

Дедуктивное обучение принято относить к области [экспертных систем](https://ru.wikipedia.org/wiki/%D0%AD%D0%BA%D1%81%D0%BF%D0%B5%D1%80%D1%82%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), поэтому термины машинное обучение и обучение по прецедентам можно считать синонимами.

Многие методы индуктивного обучения разрабатывались как альтернатива классическим статистическим подходам. Многие методы тесно связаны с [извлечением информации](https://ru.wikipedia.org/wiki/%D0%98%D0%B7%D0%B2%D0%BB%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%B8) ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) information extraction, information retrieval), [интеллектуальным анализом данных](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%BB%D0%BB%D0%B5%D0%BA%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) ([data mining](https://ru.wikipedia.org/wiki/Data_mining)).

Раздел машинного обучения, с одной стороны, образовался в результате разделения науки о [нейросетях](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C) на методы обучения сетей и виды топологий их архитектуры, с другой стороны — вобрал в себя методы математической статистики. Указанные ниже способы машинного обучения исходят из случая использования нейросетей, хотя существуют и другие методы, использующие понятие обучающей выборки — например, дискриминантный анализ, оперирующий обобщённой дисперсией и ковариацией наблюдаемой статистики, или байесовские классификаторы. Базовые виды нейросетей, такие как [перцептрон](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%86%D0%B5%D0%BF%D1%82%D1%80%D0%BE%D0%BD) и [многослойный перцептрон](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D1%81%D0%BB%D0%BE%D0%B9%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D1%86%D0%B5%D0%BF%D1%82%D1%80%D0%BE%D0%BD_%D0%A0%D0%BE%D0%B7%D0%B5%D0%BD%D0%B1%D0%BB%D0%B0%D1%82%D1%82%D0%B0) (а также их модификации), могут обучаться как с учителем, так и без учителя, с подкреплением и самоорганизацией. Но некоторые нейросети и большинство статистических методов можно отнести только к одному из способов обучения. Поэтому, если нужно классифицировать методы машинного обучения в зависимости от способа обучения, будет некорректным относить нейросети к определенному виду, правильнее было бы типизировать алгоритмы обучения нейронных сетей.

* Обучение с учителем — для каждого прецедента задаётся пара «ситуация, требуемое решение»:

1. Искусственная нейронная сеть
   1. Глубокое обучение
2. Метод коррекции ошибки
3. Метод обратного распространения ошибки
4. Метод опорных векторов

* Обучение без учителя — для каждого прецедента задаётся только «ситуация», требуется сгруппировать объекты в кластеры, используя данные о попарном сходстве объектов, и/или понизить размерность данных:

1. Альфа-система подкрепления
2. Гамма-система подкрепления
3. Метод ближайших соседей

* Обучение с подкреплением — для каждого прецедента имеется пара «ситуация, принятое решение»:

1. Генетический алгоритм.

* Активное обучение — отличается тем, что обучаемый алгоритм имеет возможность самостоятельно назначать следующую исследуемую ситуацию, на которой станет известен верный ответ:
* *Обучение с частичным привлечением учителя* (англ. *semi-supervised learning*) — для части прецедентов задается пара «ситуация, требуемое решение», а для части — только «ситуация»
* Трансдуктивное обучение — обучение с частичным привлечением учителя, когда прогноз предполагается делать только для прецедентов из тестовой выборки
* *Многозадачное обучение* (англ. *multi-task learning*) — одновременное обучение группе взаимосвязанных задач, для каждой из которых задаются свои пары «ситуация, требуемое решение»
* *Многовариантное обучение* (англ. *multiple-instance learning*) — обучение, когда прецеденты могут быть объединены в группы, в каждой из которых для всех прецедентов имеется «ситуация», но только для одного из них (причем, неизвестно какого) имеется пара «ситуация, требуемое решение»
* Бустинг (англ. *boosting* — улучшение) — это процедура последовательного построения композиции алгоритмов машинного обучения, когда каждый следующий алгоритм стремится компенсировать недостатки композиции всех предыдущих алгоритмов.
* Байесовская сеть

Целью машинного обучения является частичная или полная автоматизация решения сложных профессиональных задач в самых разных областях человеческой деятельности.

Машинное обучение имеет широкий спектр приложений:

* Распознавание речи
* Распознавание жестов
* Распознавание рукописного ввода
* Распознавание образов
* Техническая диагностика
* Медицинская диагностика
* Прогнозирование временных рядов
* Биоинформатика
* Обнаружение мошенничества
* Обнаружение спама
* Категоризация документов
* Биржевой технический анализ
* Финансовый надзор
* Кредитный скоринг
* Прогнозирование ухода клиентов
* Хемоинформатика
* Обучение ранжированию в информационном поиске

Сфера применений машинного обучения постоянно расширяется. Повсеместная информатизация приводит к накоплению огромных объёмов данных в науке, производстве, бизнесе, транспорте, здравоохранении. Возникающие при этом задачи прогнозирования, управления и принятия решений часто сводятся к обучению по прецедентам. Раньше, когда таких данных не было, эти задачи либо вообще не ставились, либо решались совершенно другими методами.

### Пример 1. Диагностируем заболевания

Пациенты — это объекты, признаки — это симптомы, результаты анализов, анамнез, история болезни. Порядковый признак — оценка тяжести состояния, количественные признаки в данном случае являются объектами, а признаками – объём лекарства, уровень гемоглобина, параметры давления и пульса, вес и возраст.

Таким образом, собирается информация о состоянии человека и загружается в компьютер. В результате мы: • выполняем диагностику болезни; • выбираем оптимальное лечение; • прогнозируем развитие болезни, длительность, исход; • прогнозируем осложнения; • выявляем сопутствующие синдромы.

Что может дать машинное обучение здесь? Прежде всего, выигрыш в скорости, ведь даже опытный доктор не сможет мгновенно обработать всю информацию по каждому пациенту, обобщив вдобавок другие истории, а потом выдав результат.

### Пример 2. Ищем полезные ископаемые

Признаки — данные геологоразведки. Для обучающей выборки берут два типа прецедентов: места, где 100 % есть месторождения, и места со схожими параметрами. Да, у месторождений тоже есть свои «синдромы».

Здесь машинное обучение может дать выигрыш в точности обнаружения ископаемых, поэтому можно говорить о большом научном интересе со стороны как геофизиков, так и геологов.

### Пример 3. Оцениваем надёжность и платёжеспособность кредитополучателей

Что может дать ML здесь, долго думать не надо — экономится не только время, но и реальные деньги финансовых учреждений. Стоит вспомнить Сбербанк, который уже давно уволил много сотрудников, которые этим занимались. Да, машин становится больше, а автоматизация этого процесса — обычное явление.

В этом примере машинного обучения кандидаты на получение кредита — это объекты, а признаки формируются из анкеты клиента (признаки будут другими, если в банк обращается юрицо).

С помощью машинного обучения делается выборка, включающая в себя «хорошие» кредитные истории и «плохие». В итоге клиенты делятся на классы, и принимается решение о выдаче или отказе.

Среди усложнённых алгоритмов машинного обучения — тот же кредитный скоринг (каждый клиент получает баллы за некоторые признаки). Есть и алгоритм и на основе прецедентов.

**Практическое занятие № 41. Компьютерные сети**

Принципы построения компьютерных сетей. *Аппаратные компоненты компьютерных сетей. Проводные и беспроводные телекоммуникационные каналы.* Сетевые протоколы. Принципы межсетевого взаимодействия. Сетевые операционные системы. *Задачи системного администрирования компьютеров и компьютерных сетей.*

Интернет. Адресация в сети Интернет (IP-адреса, маски подсети). Система доменных имен.

Технология WWW. Браузеры. Веб-сайт. Страница. Взаимодействие веб-страницы с сервером. Язык HTML. Динамические страницы. Разработка веб-сайтов. Язык HTML, каскадные таблицы стилей (CSS). *Динамический HTML. Размещение веб-сайтов. Использование сценариев на языке Javascript. Формы. Понятие о серверных языках программирования.* Сетевое хранение данных. Облачные сервисы.

Цель: получить практические навыки работы с электронной коммерцией в сети Интернет; научиться осуществлять заказы в Интернет – магазине, искать информацию в электронных СМИ и электронной библиотеке, пользоваться услугами Интернет – турагентств.

**Теория**

Развитие компьютерных информационных систем и телекоммуникационных технологий привело к формированию нового вида экономической деятельности – электронного бизнеса.

**Электронный бизнес** – это любая деловая активность, использующая возможности глобальных информационных сетей для преобразования внутренних и внешних связей с целью создания прибыли.

**Электронная коммерция** является важнейшим составным элементом электронного бизнеса. Под электронной коммерцией (e-commerce) подразумеваются любые формы деловых сделок, при которых взаимодействие сторон осуществляется электронным способом вместо физического обмена или непосредственного физического контакта, и в результате которого право собственности или право пользования товаром или услугой передается от одного лица другому.

**Киберсантинг (кибер-коммерция)** — это часть электронной коммерции, которая занимается продажей в Сети исключительно цифровых (информационных) продуктов.

**Формы электронной коммерции:**

**Интернет – магазин.** В российском Интернете существуют сотни магазинов, в которых можно купить все: компьютеры и программы, книги и диски, продукты питания и пр. Покупатель имеет возможность ознакомиться с товаром (техническими характеристиками, внешним видом), а также его ценой. Выбрав товар, потребитель может сделать непосредственно в Интернете заказ на его покупку, в котором указывается форма оплаты, время и место доставки. Оплата производится либо наличными деньгами после доставки товара, либо по кредитным карточкам.

**Интернет - Библиотеки.**Электронные библиотеки в Интернете содержат электронные копии печатных книг, диссертаций и других документов. Наиболее часто используется формат Web-страниц (HTML), однако иногда используются текстовые форматы TXT, RTF и DOC.

*Научная электронная библиотека* [*http://www.elibrary.ru*](https://infourok.ru/go.html?href=http%3A%2F%2Fwww.elibrary.ru)

*Энциклопедии и словари.* Универсальные энциклопедии содержат сведения о природе и обществе, а также по всем отраслям науки и техники.

*Электронные энциклопедии* [*http:*//www.dic.academic.ru](https://infourok.ru/go.html?href=http%3A%2F%2Fwww.dic.academic.ru)

*Википедия* - это проект свободной многоязычной энциклопедии, в которой каждый может изменить или дополнить любую статью или создать новую.

**Интернет – Турагентство.** Тураге́нт — организация, занимающаяся продажей сформированных туроператором туров. Турагент приобретает туры у туроператора и реализует туристский продукт покупателю, либо выступает посредником между туристом и туроператором за комиссионное вознаграждение, предоставляемое туроператором.

**Практические задания**

**Задание 1.** Осуществить заказ в Интернет - магазине комплектующих компьютера. Открыть в браузере сайт Интернет – магазина [http://key.ru/](https://infourok.ru/go.html?href=http%3A%2F%2Fkey.ru%2F) . Перейдите по ссылке компьютеры, из предложенного списка подберите необходимые комплектующие для полной сборки компьютера по приемлемой для вас цене.

**Задание 2.** Проанализируйте рейтинг Интернет – СМИ на сайте [http://www.mlg.ru/](https://infourok.ru/go.html?href=http%3A%2F%2Fwww.mlg.ru%2F) . Выпишите топ-5 самых цитируемых информационных агентств, цитируемых газет, журналов, интернет- ресурсов ТВ- каналов и радиостанций.

**Задание 3.** Осуществить поиск информации о направлениях отдыха на сайте Интернет – турагентства, которую оформить в виде таблицы:

Количество дней/ночей

Стоимость отеля и перелета

Стоимость тура

Вид отдыха

Дополнительные услуги

**Практическое занятие № 42**. **Деятельность в сети Интернет**

Расширенный поиск информации в сети Интернет. Использование языков построения запросов.

Другие виды деятельности в сети Интернет. Сервисы Интернета. Геолокационные сервисы реального времени (локация мобильных телефонов, определение загруженности автомагистралей и т.п.); интернет-торговля; бронирование билетов и гостиниц и т.п. Облачные версии прикладных программных систем. Новые возможности и перспективы развития Интернета: мобильность, облачные технологии, виртуализация, социальные сервисы, доступность. *Технологии «Интернета вещей». Развитие технологий распределенных вычислений.*

**Цель** получить практические навыки работы осуществления поиска информации или информационного объекта в тексте, файловых структурах, базах данных, сети Интернет.

**Задания**

1. Найдите ответы на вопросы, используя возможности любого поискового сервера. Ответы сохраните в документе с именем Интернет папки Z:\Самостоятельные работы.
2. Ответу в документе сохраните в виде таблицы:

|  |  |  |
| --- | --- | --- |
| **№ вопроса** | **Ответ** | **URL адрес** |
|  |  |  |

1. Вопросы:
2. Где растет анчар, о котором говорил Пушкин? Так ли оно ядовито?
3. Самые низкие (рекордно низкие) температуры на континентах (Антарктида, Азия, Северная Америка, Европа, Южная Америка, Африка)? Уточните географическое расположение достигнутой температуры.
4. Какой земной хищник самый крупный из ныне существующих?
5. Опишите происхождение слова «гиппопотам».
6. Сколько лет человечество использует культуру – пшеница?
7. Какой высоты главные пирамиды в Гизе? Кому они поставлены?
8. Крест и рыба – первые символы древнего христианства. Зачем древние христиане изображали рыбу?
9. Кто и когда изобрел спички, подъемный кран, ксерокопировальный аппарат?
10. Зачем жених и невеста обмениваются кольцами?
11. Как звали первых собак отправленные в космос СССР?
12. Почему необходимо внести изменения в рецепт пирога, если его придется печь на высоте более 1000 м над уровнем моря? В этом случае, как написано на коробке с полуфабрикатом, нужно добавить больше муки и воды и выпекать пирог при более высокой температуре.
13. Почему электронные часы называют кварцевыми?
14. Как появился хоккей? Какая игра являлась «прародительницей»? Откуда? В какое время?
15. Почему у пожилых людей волосатые уши?
16. Откуда произошло название одной из самых знаменитых марок автомобилей – Мерседес?

**Практическое занятие № 43**. **Социальная информатика**

Социальные сети – организация коллективного взаимодействия и обмена данными. Проблема подлинности полученной информации. *Государственные электронные сервисы и услуги.* Мобильные приложения. Открытые образовательные ресурсы. Информационная культура. Информационные пространства коллективного взаимодействия. Сетевой этикет: правила поведения в киберпространстве. *Стандартизация и стандарты в сфере информатики и ИКТ докомпьютерной эры (запись чисел, алфавитов национальных языков, библиотечного и издательского дела и др.) и компьютерной эры (языки программирования).*

**Цель:** получить практические навыки работы с электронной почтой.

**Теоретические сведения**

Электронная почта - сервис Интернета, позволяющий обмениваться электронными сообщениями через компьютерную сеть.

Широкую популярность электронная почта завоевала потому, что имеет несколько преимуществ перед обычной почтой:

* Это скорость пересылки сообщений.
* Надежность.
* Рассылка сообщений сразу нескольким абонентам.
* К тому же обойдется электронное сообщение дешевле, чем обычное.

Для отправки письма необходимо указать адрес, на который оно будет отправлено.

Адрес электронной почты однозначно идентифицирует почтовый ящик, в который следует доставить сообщение электронной почты; образуется при создании почтового ящика на любом из серверов. Адрес электронной почты записывается по определенной форме и состоит из 2-ух частей, разделенных символом @. Первая часть почтового адреса носит название "имя пользователя". Имеет произвольный характер и задается самим пользователем при регистрации. В качестве него может быть фамилия пользователя, его имя, псевдоним, написанные латинскими буквами, цифрами, разделенные точкой, тире, символом подчеркивания. Всего не более 31 символов. Вторя часть адреса - имя почтового сервера, на котором пользователь зарегистрировал свой почтовый ящик. Имя почтового сервера имеет крайнюю группа букв, которая обозначает зону ресурсов информационной сети, выделенную владельцу, например, какой-либо стране, международной организации. Например, группа букв ru определяет принадлежность данного сервера России.

Для отправки письма - обычного или электронного - нам необходим почтовый ящик. Почтовые ящики впервые появились в России на улицах Санкт-Петербурга и Москвы в 1848 году. В настоящее время встречаются почтовые ящики различных форм - в виде животных, различных предметов и даже в виде компьютера. Первый электронный ящик появился в 1971 году, разработанный инженером Рэем Томлинсоном, и представлял собой обычный текстовый файл, в конец которого любой из имевших доступ к системе мог дописывать свое сообщение. Сегодня почтовый электронный ящик представляет собой часть дискового пространства на сервере с определенным адресом, где может храниться почтовая информация для пользователя сети Интернет. Для отправки письма через электронную почту необходимо создать свой электронный почтовый ящик, пройдя соответствующую регистрацию. Любой пользователь Интернета может зарегистрировать почтовый ящик на одном из серверов Интернета, предоставляющем почтовые услуги. Среди серверов бесплатных почтовых служб наиболее известны: Rambler, Yahoo!, Яндекс, Google, Mail. Для рассмотрения создания электронного почтового ящика используем в качестве примера почтовую службу Mail.ru.

алгоритмом отправки электронного письма.

**Практические задания**

**Упражнение по отправке электронного письма с прикрепленным файлом**

1. В почтовом боксе Mail.ru нажать ссылку **Написать письмо.**
2. Заполнить поля:
   1. **Кому – записать адрес электронной почты kozlova.chetk@mail.ru**
   2. **Тема – Ответы на вопросы**
3. Написать текст письма: **Ответы на вопросы, используя возможности любого поискового** сервера Нажать кнопку **Прикрепить файлы.**
4. В своей сетевой папке выбрать файл **Интернет.**
5. Нажать кнопку **Открыть**.
6. Нажать кнопку **Отправить**

**Практическое занятие № 44**. **Информационная безопасность**

Средства защиты информации в автоматизированных информационных системах (АИС), компьютерных сетях и компьютерах. Общие проблемы защиты информации и информационной безопасности АИС. Компьютерные вирусы и вредоносные программы. Использование антивирусных средств. Электронная подпись, сертифицированные сайты и документы. Правовые нормы использования компьютерных программ и работы в Интернете. Законодательство РФ в области программного обеспечения. Техногенные и экономические угрозы, связанные с использованием ИКТ. Правовое обеспечение информационной безопасности.

*Цель*: закрепление основных понятий: компьютерный вирус, основные типы компьютерных вирусов, этапы действия вируса, защита от компьютерных вирусов

Теория

Компьютерный вирус – это программный код, встроенный в другую программу, или в документ, или в определенные области носителя данных и предназначенный для несанкционированных действий на компьютере.

1. Основные типы компьютерных вирусов:

|  |  |  |
| --- | --- | --- |
| Программные | Загрузочные | Макровирусы |
| Это блоки программного кода, внедренные внутрь других прикладных программ.  Вирусный код запускается при запуске программы. | Поражают системные области магнитных носителей (гибких и жестких дисков). Заражение происходит при загрузке ПК с зараженного носителя. | Поражают документы, выполненные в некоторых прикладных программах (например, Word).  Заражение происходит при открытии файла документа в окне программы, если в ней не отключена возможность исполнения макрокоманд (макросов). |

2. Этапы действия вируса:

* Размножение – вирусный код может воспроизводить себя в теле других программ.
* Вирусная атака – после создания достаточного числа копий программный вирус начинает осуществлять разрушение: нарушение работы программ и ОС, удаление информации на жестком диске, самые разрушительные вирусы вызывают форматирование жесткого диска. Некоторые вирусы могут уничтожать данные, в этом случае требуется замена микросхемы (хотя считается, что никакой вирус не в состоянии вывести из строя аппаратное обеспечение ПК).

3. Защита от компьютерных вирусов

Существуют три рубежа защиты:

* предотвращение поступления вирусов;
* предотвращение вирусной атаки, если вирус поступил на ПК;
* предотвращение разрушительных последствий, если атака произошла.

4. Методы реализации защиты

* Программные
* Аппаратные
* Организационные

5. Средства антивирусной защиты:

* Основное средство – резервное копирование наиболее ценных данных. В случае утраты информации жесткие диски форматируют, устанавливают ОС с дистрибутивного CD-диска и все необходимые программы, а данные – с резервного носителя (который должен храниться отдельно от ПК). Все регистрационные и парольные данные для доступа в Интернет рекомендуется хранить не на ПК, а в служебном дневнике в сейфе.
* Вспомогательные средства – это антивирусные программы и аппаратные средства.
  + Аппаратное средство: отключение перемычки на материнской плате не позволит осуществить стирание микросхемы BIOS ни вирусу, ни злоумышленнику, ни неаккуратному пользователю.
  + Антивирусная программа сравнивает коды программ с известными ей вирусами, которые хранятся в ее базе данных. Обновление базы – 2 раза в месяц (не реже 1 раза в 3 месяца).

6. Антивирусные программы

Для обнаружения, удаления и защиты от компьютерных вирусов разработаны специальные программы, которые позволяют обнаруживать и уничтожать вирусы. Такие программы называются антивирусными.  
Антивирусная программа сравнивает коды программ с известными ей вирусами, которые хранятся в ее базе данных. Обновление базы – 2 раза в месяц (не реже 1 раза в 3 месяца).

* Norton AntiVirus4.0 и 5.0 *(производитель: “Symantec”)*  
  Один из наиболее известных и популярных антивирусов. Процент распознавания вирусов очень высокий (близок к 100 %). В программе используется механизм, который позволяет распознавать новые неизвестные вирусы.
* Dr.Web *(производитель: “Диалог Наука”)*  
  Популярный отечественный антивирус. Хорошо распознает вирусы, но в его базе их гораздо меньше, чем у других антивирусных программ.
* Antiviral Toolkit Pro *(производитель: “Лаборатория Касперского”)*.  
  Это антивирус признан во всем мире, как один из самых надежных. Несмотря на простоту в использовании он обладает всем необходимым арсеналом для борьбы с вирусами. Эвристический механизм, избыточное сканирование, сканирование архивов и упакованных файлов - это далеко не полный перечень его возможностей.

Вопросы теста:

1. Что такое компьютерный вирус?

1) Прикладная программа.  
2) Системная программа.  
3) Программа, выполняющая на компьютере несанкционированные действия.  
4) База данных.

2. Основные типы компьютерных вирусов:

1) Аппаратные, программные, загрузочные .  
2) Программные, загрузочные, макровирусы.  
3) Файловые, программные, макровирусы.

3. Этапы действия программного вируса:

1) Размножение, вирусная атака.  
2) Запись в файл, размножение.  
3) Запись в файл, размножение, уничтожение программы.

4. В чем заключается размножение программного вируса?

1) Программа-вирус один раз копируется в теле другой программы.  
2) Вирусный код неоднократно копируется в теле другой программы.

5. Что называется вирусной атакой?

1) Неоднократное копирование кода вируса в код программы.  
2) Отключение компьютера в результате попадания вируса.  
3) Нарушение работы программы, уничтожение данных, форматирование жесткого диска.

6. Какие существуют методы реализации антивирусной защиты?

1) Аппаратные и программные.  
2) Программные, аппаратные и организационные.  
3) Только программные.

7. Какие существуют основные средства защиты?

1) Резервное копирование наиболее ценных данных.  
2) Аппаратные средства.  
3) Программные средства.

8. Какие существуют вспомогательные средства защиты?

1) Аппаратные средства.  
2) Программные средства.  
3) Аппаратные средства и антивирусные программы.

9. На чем основано действие антивирусной программы?

1) На ожидании начала вирусной атаки.  
2) На сравнении программных кодов с известными вирусами.  
3) На удалении зараженных файлов.

10. Какие программы относятся к антивирусным

1) AVP, DrWeb, Norton AntiVirus.  
2) MS-DOS, MS Word, AVP.  
3) MS Word, MS Excel, Norton Commander.

**ЛИТЕРАТУРА**

1. Цветкова М.С. Информатика: учебник / М.С. Цветкова, И.Ю. Хлобыстова. - 6-е изд.,стер. - М.: ИЦ Академия, 2020. - 352 с.,ил. - (Профессиональное образование).
2. Цветкова М.С. Информатика. Практикум для профессий и специальностей естественно - научного и гуманитарного профилей: учеб. пособие / М.С. Цветкова, И.Ю. Хлобыстова. - 3- е изд., стер. - М.: ИЦ Академия, 2019. - 240 с. - (Профессиональное образование)
3. Гребенюк Е.И. Технические средства информатизации: учебник / Е.И. Гребенюк, Н.А. Гребенюк. - 10-е изд. стер. - М.: ИЦ Академия, 2016. - 352 с. - (Профессиональное образование).