ГУ « Комплекс «Музыкальный колледж - музыкальная школа-интернат для одаренных детей»

Буканова Айжан Айтбаевна

Учитель Информатики

**Методическое пособие по информатике для 7 класса**

Павлодар

2025

Методическое пособие

Подготовила Буканова Айжан Айтбаевна

Преподаватель «информатики»

Рассмотрено на заседании отделения протокол №\_\_\_ от «\_\_»\_\_\_\_\_\_202\_г.

Преподавателем -исследователем Абуова С.А. подпись\_\_\_\_\_\_\_\_\_\_\_

Одобрено методическим советом школы  протокол № 2 от «\_\_»\_\_\_\_\_\_202\_г.

Председатель методического совета

Кунязова А.С. подпись\_\_\_\_\_\_\_\_\_\_\_\_\_

**Содержание**

Пояснительная записка

1.Содержание курса……………………………………………...4

1.1 Компьютер как универсальное устройство обработки информации..6

1.2 Основные компоненты ПК и их функции……………………………….. 7

1.3 Операционные системы и программное обеспечение…………………...9

1.4 Основы информационной безопасности………………………………...10

2.1 Основные понятия алгоритмов…………………………………………..12

2.2 Ветвления и циклы в алгоритмах………………………………………..13

2.3 Блок-схемы и их применение…………………………………………….15

2.4 Введение в программирование (Scratch, Python)……………………….16

2.5 Разработка простых программ…………………………………………...18

3.1 Виды сетей и их топология………………………………………………21

3.2 Основные протоколы и способы передачи данных…………………….24

3.3 Основы кибербезопасности………………………………………………26

3.4 Работа с облачными сервисами………………………………………….28

4.1 Форматы файлов и их особенности……………………………………...31

4.2 Основы работы с графикой и мультимедиа…………………………….35

4.3 Создание и редактирование презентаций……………………………….38

5. Разработка и защита мини-проекта……………………………………….42

 Заключение…………………………………………………………………..46

**Методическое пособие по информатике для 7 класса**

**1. Пояснительная записка,** Данная программа предназначена для изучения информатики в 7 классе и направлена на развитие алгоритмического мышления, освоение основ программирования и цифровой грамотности. Она соответствует современным требованиям к образовательному процессу и способствует формированию ключевых компетенций учащихся.

**2. Цели и задачи** Цель курса – формирование у учащихся базовых знаний и умений в области алгоритмизации, программирования, работы с цифровыми технологиями и информационной безопасностью.

Задачи:

* Развитие логического и алгоритмического мышления.
* Освоение основ программирования в среде (Scratch, Python).
* Изучение принципов работы компьютерных сетей и интернета.
* Формирование навыков безопасного поведения в цифровой среде.
* Развитие творческих способностей через проектную деятельность.

**3. Содержание курса**

**Раздел 1. Информационные технологии и цифровая грамотность** (6 часов)

* Компьютер как универсальное устройство обработки информации.
* Основные компоненты ПК и их функции.
* Операционные системы и программное обеспечение.
* Основы информационной безопасности.

**Раздел 2. Алгоритмизация и программирование** (18 часов)

* Основные понятия алгоритмов.
* Ветвления и циклы в алгоритмах.
* Блок-схемы и их применение.
* Введение в программирование (Scratch, Python).
* Разработка простых программ.

**Раздел 3. Компьютерные сети и интернет** (8 часов)

* Виды сетей и их топология.
* Основные протоколы и способы передачи данных.
* Основы кибербезопасности.
* Работа с облачными сервисами.

**Раздел 4. Работа с цифровыми данными** (8 часов)

* Форматы файлов и их особенности.
* Основы работы с графикой и мультимедиа.
* Создание и редактирование презентаций.

**Раздел 5. Проектная деятельность** (5 часов)

* Разработка и защита мини-проекта.
* Работа в командах.
* Представление проектов.

**4. Методические рекомендации** Для достижения наилучших результатов рекомендуется сочетать объяснительно-иллюстративный метод, практическую деятельность, работу в группах и проектное обучение. Использование интерактивных технологий и сервисов (Google Classroom, Scratch, Python IDE) поможет повысить интерес учащихся к предмету.

**5. Формы контроля и оценивания**

* Текущий контроль: тестирование, выполнение практических заданий.
* Итоговый контроль: защита проектов, проверочные работы.
* Оценивание проводится по критериям: правильность выполнения заданий, креативность, самостоятельность.

**6. Ожидаемые результаты** В результате изучения курса учащиеся должны:

* Понимать принципы работы компьютерных технологий.
* Владеть базовыми навыками программирования.
* Уметь безопасно работать в цифровой среде.
* Разрабатывать и представлять собственные проекты.

Программа рассчитана на 45 часов и ориентирована на формирование у учащихся интереса к информатике и дальнейшему развитию в сфере IT.

**1.1 Компьютер как универсальное устройство обработки информации**

Компьютер — это электронное устройство, предназначенное для обработки, хранения и передачи информации. Он используется в различных сферах деятельности благодаря своей универсальности и высокой скорости выполнения вычислений.

**Основные функции компьютера:**

1. **Ввод данных** – получение информации из внешних источников (клавиатура, мышь, сканер, микрофон).
2. **Обработка данных** – выполнение вычислений, преобразование информации по заданному алгоритму.
3. **Хранение данных** – сохранение информации на жёстком диске, SSD, флеш-накопителях и в облачных хранилищах.
4. **Передача данных** – обмен информацией через сети передачи данных (локальные сети, интернет).
5. **Вывод данных** – представление обработанной информации в удобном виде (монитор, принтер, динамики).

**Основные компоненты компьютера:**

* **Процессор (ЦП)** – выполняет вычисления и управляет работой компьютера.
* **Оперативная память (ОЗУ)** – временно хранит данные, с которыми работает процессор.
* **Накопители (HDD, SSD)** – предназначены для долгосрочного хранения данных.
* **Материнская плата** – соединяет все компоненты и обеспечивает их взаимодействие.
* **Видеокарта** – обрабатывает графическую информацию и выводит изображение на экран.
* **Блок питания** – обеспечивает энергией все компоненты компьютера.

**Применение компьютеров:**

Компьютеры используются во всех сферах жизни:

* В образовании – для дистанционного обучения и проведения исследований.
* В медицине – для диагностики, хранения медицинских данных, работы с изображениями.
* В бизнесе – для автоматизации процессов, бухгалтерии и анализа данных.
* В развлечениях – для игр, создания и обработки мультимедиа-контента.
* В науке – для сложных расчетов, моделирования и анализа больших данных.

**Вывод:**

Компьютер является универсальным устройством обработки информации, выполняя множество функций в различных сферах деятельности. Благодаря высокой скорости обработки данных и многофункциональности, он стал незаменимым инструментом современного общества.

### ****1.2 Основные компоненты ПК и их функции****

Компьютер состоит из множества компонентов, каждый из которых выполняет определённые функции. Основные аппаратные части ПК включают в себя:

#### ****1. Центральный процессор (ЦП, CPU – Central Processing Unit)****

Функция:

* Обрабатывает команды и управляет работой всех компонентов компьютера.
* Выполняет вычисления и логические операции.
* Определяет скорость работы системы.

#### ****2. Оперативная память (ОЗУ, RAM – Random Access Memory)****

Функция:

* Временное хранение данных, с которыми работает процессор.
* Позволяет быстро получать доступ к необходимой информации.
* После выключения компьютера данные в ОЗУ стираются.

#### ****3. Жесткий диск (HDD) и твердотельный накопитель (SSD)****

Функция:

* Долговременное хранение операционной системы, программ и пользовательских данных.
* SSD быстрее, надёжнее и энергоэффективнее, чем HDD.

#### ****4. Материнская плата (Motherboard)****

Функция:

* Соединяет все компоненты ПК и обеспечивает их взаимодействие.
* Обеспечивает питание и передачу данных между компонентами.
* Включает в себя слоты для процессора, оперативной памяти, видеокарты, жестких дисков и других устройств.

#### ****5. Видеокарта (GPU – Graphics Processing Unit)****

Функция:

* Обрабатывает графическую информацию и отвечает за вывод изображения на экран.
* Может быть интегрированной (встроенной в процессор) или дискретной (отдельной).
* Важна для работы с 3D-графикой, видеообработкой и компьютерными играми.

#### ****6. Блок питания (PSU – Power Supply Unit)****

Функция:

* Обеспечивает энергией все компоненты компьютера.
* Преобразует напряжение из сети в необходимое для работы компонентов.

#### ****7. Устройства ввода****

Функция:

* Позволяют вводить информацию в компьютер.
* Клавиатура – для текстового ввода.
* Мышь – для управления курсором и выбора объектов.
* Сканер, микрофон, графический планшет – для ввода графической и звуковой информации.

#### ****8. Устройства вывода****

Функция:

* Отображают информацию пользователю.
* Монитор – показывает изображение.
* Принтер – выводит информацию на бумагу.
* Акустическая система (динамики, наушники) – воспроизводит звук.

### ****Вывод****

Все компоненты ПК работают совместно, обеспечивая эффективную обработку, хранение и вывод информации. От их характеристик зависит производительность компьютера, его возможности и скорость работы.

### ****1.3 Операционные системы и программное обеспечение****

Компьютер без программного обеспечения не может выполнять никакие задачи. Вся работа ПК организована операционной системой, которая управляет ресурсами и взаимодействием пользователя с устройством.

### ****1. Операционные системы (ОС)****

Операционная система (ОС) – это основная программа, управляющая всеми процессами компьютера и обеспечивающая выполнение других программ.

#### ****Функции операционной системы:****

* Управление аппаратными ресурсами (процессором, памятью, устройствами ввода/вывода).
* Обеспечение взаимодействия между программами и пользователем.
* Организация хранения и обработки данных на носителях.
* Запуск и управление приложениями.

#### ****Основные виды операционных систем:****

1. **Windows** – популярная коммерческая ОС от Microsoft, с удобным графическим интерфейсом.
2. **Linux** – свободно распространяемая ОС, широко используемая в серверах и разработке.
3. **macOS** – операционная система от Apple, работающая на компьютерах Mac.
4. **Android и iOS** – мобильные операционные системы для смартфонов и планшетов.

### ****2. Программное обеспечение****

Программное обеспечение (ПО) – это набор программ, используемых для выполнения различных задач на компьютере.

#### ****Виды программного обеспечения:****

##### **1) Системное ПО**

Программы, которые обеспечивают работу компьютера:

* **Операционные системы (Windows, Linux, macOS)**
* **Драйверы устройств** – программы для взаимодействия ОС с аппаратными компонентами.
* **Утилиты** – вспомогательные программы (архиваторы, антивирусы, дефрагментаторы).

##### **2) Прикладное ПО**

Программы, предназначенные для решения конкретных задач пользователя:

* **Офисные приложения** (Microsoft Office, Google Docs).
* **Графические редакторы** (Adobe Photoshop, GIMP).
* **Веб-браузеры** (Google Chrome, Mozilla Firefox).
* **Мультимедийные программы** (VLC Media Player, Adobe Premiere).
* **Игры и образовательные приложения**.

##### **3) Системы программирования**

Программы для создания и разработки других программ:

* **Языки программирования** (Python, C++, Java).
* **Среды разработки** (PyCharm, Visual Studio, Eclipse).

### ****Вывод****

Операционные системы являются основой работы компьютера, а программное обеспечение делает его полезным для пользователя. Выбор операционной системы и прикладных программ зависит от задач, которые необходимо решать.

### ****1.4 Основы информационной безопасности****

Информационная безопасность – это защита данных и систем от несанкционированного доступа, потери, изменения или уничтожения.

### ****1. Основные угрозы информационной безопасности****

1. **Вирусы и вредоносное ПО** – программы, которые могут нанести вред компьютеру (трояны, шпионские программы, черви).
2. **Фишинг** – мошеннические сайты и письма, предназначенные для кражи личных данных.
3. **Взлом аккаунтов и паролей** – атаки на учетные записи пользователей.
4. **Несанкционированный доступ** – проникновение злоумышленников в компьютерные системы.
5. **Социальная инженерия** – манипуляции, заставляющие пользователя самостоятельно передавать конфиденциальные данные.

### ****2. Способы защиты информации****

#### ****1) Защита компьютера****

* Использование **антивирусных программ** (Kaspersky, Avast, Windows Defender).
* Регулярное **обновление программ и ОС**.
* Настройка **межсетевого экрана (файрвола)**.

#### ****2) Безопасность паролей****

* Использование **сложных паролей** (не менее 8 символов, цифры, буквы, спецсимволы).
* Неиспользование **одинаковых паролей** для разных сервисов.
* Регулярная смена паролей.

#### ****3) Защита персональных данных****

* Не передавать личные данные посторонним лицам.
* Не вводить данные на подозрительных сайтах.
* Использование **двухфакторной аутентификации**.

#### ****4) Безопасность в интернете****

* Проверка **адреса сайтов** перед вводом данных.
* Не скачивать файлы из неизвестных источников.
* Осторожность при открытии **подозрительных email-сообщений**.

### ****3. Основные принципы информационной безопасности****

* **Конфиденциальность** – информация доступна только тем, кому она предназначена.
* **Целостность** – данные остаются неизменными без разрешения владельца.
* **Доступность** – информация доступна только авторизованным пользователям.

### ****Вывод****

Информационная безопасность – это комплекс мер, направленный на защиту данных. Соблюдение правил безопасности позволяет минимизировать риски и избежать угроз в цифровой среде.

**2.1 Основные понятия алгоритмов**

**Алгоритм** – это последовательность шагов (инструкций), которые необходимо выполнить для решения определенной задачи.

**1. Свойства алгоритмов**

1. **Дискретность** – алгоритм состоит из отдельных шагов.
2. **Определенность (детерминированность)** – каждый шаг должен быть четко сформулирован.
3. **Конечность** – алгоритм должен завершаться за конечное число шагов.
4. **Результативность** – выполнение алгоритма должно приводить к получению результата.
5. **Массовость** – алгоритм должен быть применим к разным входным данным.

**2. Способы представления алгоритмов**

1. **Словесное описание** – представление алгоритма в виде текста.
2. **Блок-схема** – графическое изображение алгоритма с использованием стандартных фигур.
3. **Псевдокод** – запись алгоритма с использованием условных операторов.
4. **Программный код** – реализация алгоритма на языке программирования.

**3. Основные структуры алгоритмов**

1. **Линейные алгоритмы** – команды выполняются последовательно.
2. **Разветвляющиеся (ветвления)** – выполнение зависит от условия (если ... то ... иначе).
3. **Циклические алгоритмы** – повторение действий (пока или для).

**4. Примеры алгоритмов**

* **Линейный**: сложение двух чисел.
* **Разветвляющийся**: проверка, является ли число четным.
* **Циклический**: нахождение суммы чисел от 1 до N.

**Вывод**

Алгоритмы лежат в основе программирования и компьютерных вычислений. Умение составлять алгоритмы помогает решать задачи эффективно и правильно.

**2.2 Ветвления и циклы в алгоритмах**

Алгоритмы могут содержать конструкции, которые позволяют выполнять определенные действия в зависимости от условий или повторять действия многократно. К таким конструкциям относятся **ветвления** и **циклы**.

**1. Ветвления в алгоритмах**

**Ветвление** — это конструкция, которая выполняет разные действия в зависимости от выполнения определенного условия.

**Типы ветвлений:**

1. **Полное ветвление (if-else)**

if x > 0:

 print("Число положительное")

else:

 print("Число отрицательное или равно нулю")

Если условие выполняется (x > 0), выполняется первая команда, иначе — вторая.

1. **Неполное ветвление (if без else)**

if x > 0:

 print("Число положительное")

Если условие ложно, код просто пропускает этот блок.

1. **Множественный выбор (if-elif-else)**

if x > 0:

 print("Число положительное")

elif x < 0:

 print("Число отрицательное")

else:

 print("Число равно нулю")

Позволяет обработать несколько условий.

**2. Циклы в алгоритмах**

**Цикл** — это конструкция, которая повторяет определенные действия до выполнения условия.

**Типы циклов:**

1. **Цикл с известным числом повторений (for)**

for i in range(5):

 print(i) # Выведет числа 0, 1, 2, 3, 4

Используется, когда количество повторений известно заранее.

1. **Цикл с неизвестным числом повторений (while)**

x = 5

while x > 0:

 print(x)

 x -= 1

Выполняется, пока выполняется условие (x > 0).

**3. Операторы управления циклами**

1. **break** — прерывает выполнение цикла.

for i in range(10):

 if i == 5:

 break

 print(i) # Выведет числа 0, 1, 2, 3, 4

1. **continue** — пропускает текущую итерацию и переходит к следующей.

for i in range(5):

 if i == 2:

 continue

 print(i) # Пропустит число 2

**Вывод**

* Ветвления позволяют алгоритму выбирать различные пути выполнения.
* Циклы позволяют многократно повторять действия.
* Использование ветвлений и циклов делает алгоритмы более гибкими и эффективными.

**2.3 Блок-схемы и их применение**

**Блок-схема** — это графическое представление алгоритма, используемое для наглядного описания последовательности выполнения действий. Блок-схемы помогают анализировать алгоритмы, выявлять ошибки и упрощать программирование.

**1. Основные элементы блок-схем**

| **Фигура** | **Название** | **Функция** |
| --- | --- | --- |
| 🔹 Овал | **Начало / Конец** | Определяет начало и конец алгоритма |
| 🔷 Параллелограмм | **Ввод / Вывод** | Используется для ввода данных и вывода результатов |
| 🔲 Прямоугольник | **Процесс** | Выполнение вычислений или действий |
| 🔺 Ромб | **Условие (ветвление)** | Определяет выбор дальнейшего пути в зависимости от условия |
| ➡️ Стрелка | **Поток управления** | Определяет направление выполнения команд |

**2. Пример блок-схемы**

Рассмотрим алгоритм, который запрашивает у пользователя число и проверяет, является ли оно положительным, отрицательным или нулём.

**Алгоритм:**

1. Начало
2. Ввод числа
3. Если число > 0 → вывести "Положительное"
4. Иначе если число < 0 → вывести "Отрицательное"
5. Иначе → вывести "Ноль"
6. Конец

**Блок-схема:**

 **⬤ Начало**

 **↓**

 **🔷 Ввод числа**

 **↓**

 **🔺 Число > 0?**

 **/ \**

**Да Нет**

 **↓ 🔺 Число < 0?**

 **🔷 Вывести "Положительное" / \**

 **↓ Да Нет**

 **⬤ Конец ↓ 🔷 Вывести "Ноль"**

 **🔷 Вывести "Отрицательное"**

 **↓**

 **⬤ Конец**

**3. Применение блок-схем**

* **В программировании:**
* Используются для планирования алгоритмов перед написанием кода.
* Помогают визуализировать структуру программы.
* **В инженерии и бизнесе:**
* Применяются для описания технологических процессов.
* Используются в моделировании работы систем.
* **В образовании:**
* Упрощают объяснение алгоритмов.
* Помогают ученикам понимать логику программирования.

**Вывод**

Блок-схемы — это мощный инструмент для визуального представления алгоритмов, который делает их понятными и удобными для анализа.

### 2.4 Введение в программирование

Программирование — это процесс создания программ, которые управляют компьютерами и автоматизируют выполнение различных задач. Оно является ключевым навыком в мире технологий и используется в самых разных сферах: от разработки игр и мобильных приложений до управления роботами и анализирования данных.

#### ****Что такое программа?****

Программа — это набор инструкций, которые компьютер выполняет для достижения определённой цели. Эти инструкции пишутся на языках программирования, которые понятны как человеку, так и компьютеру.

#### ****Языки программирования****

Существует множество языков программирования, каждый из которых подходит для различных задач. Вот несколько популярных языков:

* **Python** — удобен для начинающих, широко используется в науке, разработке веб-приложений и искусственном интеллекте.
* **JavaScript** — применяется для создания веб-сайтов и интерактивных веб-приложений.
* **C++** — подходит для разработки игр, операционных систем и высокопроизводительных приложений.
* **Scratch** — визуальный язык программирования, который используется для обучения детей и начинающих основам программирования.

#### ****Алгоритмы и логика****

Программирование основано на алгоритмах — чётко определённых последовательностях шагов, которые решают задачу. Например, если представить алгоритм приготовления чая, он будет включать шаги:

1. Вскипятить воду.
2. Положить чайный пакетик в чашку.
3. Залить горячую воду.
4. Дождаться заваривания.
5. Добавить сахар или молоко по желанию.

В программировании алгоритмы записываются в виде кода, который выполняет компьютер.

#### ****Типы программирования****

* **Процедурное программирование** — программа состоит из последовательности команд (например, в C, Pascal).
* **Объектно-ориентированное программирование (ООП)** — программа строится на основе взаимодействия объектов (Java, Python, C++).
* **Функциональное программирование** — программа использует функции как основные строительные блоки (Haskell, Lisp).

#### ****Как начать программировать?****

1. Выбрать язык программирования. Scratch — отличный выбор для новичков!
2. Изучить основы: переменные, циклы, условия, функции.
3. Практиковаться, создавая небольшие проекты (например, анимации в Scratch или простые программы в Python).
4. Читать книги, смотреть обучающие видео и участвовать в онлайн-курсах.
5. Решать задачи на кодинг-платформах, таких как CodeWars, LeetCode и HackerRank.

Программирование — это не только полезный, но и увлекательный навык. Начав с простых проектов, можно постепенно перейти к созданию сложных приложений и даже стать профессиональным разработчиком!

**2.5 Разработка простых программ**

Начать программировать можно с создания простых программ, которые помогают освоить основные концепции: переменные, условия, циклы и функции. Рассмотрим несколько примеров простых программ на разных языках.

**1. Простой калькулятор (Python)**

Этот код позволяет выполнять арифметические операции (сложение, вычитание, умножение, деление).

# Ввод чисел и операции

num1 = float(input("Введите первое число: "))

operation = input("Введите операцию (+, -, \*, /): ")

num2 = float(input("Введите второе число: "))

# Выполнение операции

if operation == "+":

 result = num1 + num2

elif operation == "-":

 result = num1 - num2

elif operation == "\*":

 result = num1 \* num2

elif operation == "/":

 if num2 != 0:

 result = num1 / num2

 else:

 result = "Ошибка: деление на ноль"

else:

 result = "Неизвестная операция"

print("Результат:", result)

**2. Простая игра «Угадай число» (Python)**

Компьютер загадывает число, а игрок должен его угадать.

import random

secret\_number = random.randint(1, 10)

attempts = 0

while True:

 guess = int(input("Угадайте число от 1 до 10: "))

 attempts += 1

 if guess < secret\_number:

 print("Загаданное число больше.")

 elif guess > secret\_number:

 print("Загаданное число меньше.")

 else:

 print(f"Поздравляем! Вы угадали число за {attempts} попыток.")

 break

**3. Анимация в Scratch**

В Scratch можно создать простую анимацию:

1. Выберите спрайт (например, кота).
2. Добавьте следующий код в редактор Scratch:
	* **Когда нажат зелёный флаг**
	* **Повторять всегда**
		+ **Переместиться на 10 шагов**
		+ **Если касается края, оттолкнуться**

После запуска программы кот будет двигаться по сцене и отскакивать от краёв.

**4. Светофор (Scratch)**

Можно создать анимацию светофора:

* Используйте 3 цветных круга (красный, жёлтый, зелёный).
* Напишите код для переключения цветов через определённое время.

**Пример кода:**

1. **Когда нажат зелёный флаг**
2. **Повторять всегда**
	* **Показать красный свет, ждать 3 секунды**
	* **Скрыть красный, показать жёлтый, ждать 1 секунду**
	* **Скрыть жёлтый, показать зелёный, ждать 3 секунды**

Эти примеры показывают, что программирование может быть простым и интересным!

Вот несколько примеров задач по темам: логические операции, переводы чисел, алгоритмы, таблицы истинности и графы:

Задача 1: Логические операции

Заданы логические выражения:

1. (A AND B)

2. (A OR B)

3. NOT A

Для каких значений A и B выражения будут истинными (1)?

Решение:

1. Выражение "A AND B" истинно, когда оба A и B равны 1.

2. Выражение "A OR B" истинно, если хотя бы одно из значений A или B равно 1.

3. Выражение "NOT A" истинно, если A равно 0.

Задача 2: Преобразование чисел

Преобразуйте число 25 из десятичной системы счисления в двоичную.

Решение: Для перевода числа в двоичную систему нужно делить число на 2, записывать остатки и читать их в обратном порядке.

25 ÷ 2 = 12 (остаток 1)
12 ÷ 2 = 6 (остаток 0)
6 ÷ 2 = 3 (остаток 0)
3 ÷ 2 = 1 (остаток 1)
1 ÷ 2 = 0 (остаток 1)

Ответ: 25 в двоичной системе = 11001.

Задача 3: Алгоритм

Напишите алгоритм нахождения наибольшего из трех чисел.

Решение: Алгоритм:

1. Ввести три числа: A, B, C.

2. Если A > B и A > C, то наибольшее число — A.

3. Если B > A и B > C, то наибольшее число — B.

4. Если C > A и C > B, то наибольшее число — C.

Задача 4: Таблица истинности

Составьте таблицу истинности для логического выражения: (A AND B) OR (NOT C)

Решение: Создаем таблицу для всех возможных значений A, B и C:

Задача 5: Основы работы с графами

Дано дерево (граф), состоящее из узлов: A, B, C, D, E. Укажите, какие узлы можно достичь, начиная с узла A.

Решение: Предположим, что дерево представлено следующим образом:

A — B, C

B — D

C — E

Ответ: Из узла A можно достичь узлов B, C, D, E.

### ****3.1 Виды сетей и их топология****

Компьютерные сети можно классифицировать по разным критериям, таким как масштаб, способы соединения устройств и архитектура передачи данных.

## ****1. Виды сетей****

### ****1.1. По масштабу (размеру) сети****

* **Персональная сеть (PAN, Personal Area Network)**
	+ Охватывает небольшую территорию, например, связь между смартфоном, ноутбуком и беспроводными наушниками.
	+ Пример: Bluetooth-соединение.
* **Локальная сеть (LAN, Local Area Network)**
	+ Охватывает небольшую территорию (дом, офис, учебное заведение).
	+ Использует кабельные (Ethernet) или беспроводные (Wi-Fi) соединения.
	+ Пример: сеть в офисе или школе.
* **Городская сеть (MAN, Metropolitan Area Network)**
	+ Объединяет несколько локальных сетей в пределах города.
	+ Пример: сеть провайдера интернет-услуг в городе.
* **Глобальная сеть (WAN, Wide Area Network)**
	+ Связывает сети по всему миру.
	+ Пример: Интернет.

### ****1.2. По способу соединения устройств****

* **Проводные сети** (Ethernet, оптоволоконные сети)
* **Беспроводные сети** (Wi-Fi, мобильные сети 4G/5G)

### ****2. Топология сетей****

Топология сети определяет, как устройства соединены друг с другом и как передаются данные.

#### ****2.1. Виды топологий****

* **Шинная (Bus)**
* Все устройства подключены к одному общему каналу (шине).
* Недорого, но если кабель повреждён, вся сеть выходит из строя.
* Использовалась в старых Ethernet-сетях.
* **Звезда (Star)**
* Все устройства подключены к центральному узлу (например, коммутатору).
* Если один компьютер выйдет из строя, сеть продолжит работать.
* Используется в современных локальных сетях (LAN).
* **Кольцо (Ring)**
* Каждое устройство соединено с двумя соседними, образуя кольцо.
* Данные передаются по кругу, но если одно соединение разорвётся, сеть не будет работать.
* Ранее использовалась в Token Ring-сетях.
* **Ячеистая (Mesh)**
* У каждого узла есть несколько соединений с другими узлами.
* Высокая надёжность, но сложна в настройке.
* Используется в военных и промышленных сетях.
* **Древовидная (Tree)**
* Иерархическая структура, похожая на несколько соединённых сетей типа «звезда».
* Применяется в крупных корпоративных сетях.
* **Смешанная (Hybrid)**
* Комбинирует элементы разных топологий.
* Используется в реальных крупных сетях, таких как интернет.

### ****3. Итоги****

| **Топология** | **Достоинства** | **Недостатки** |
| --- | --- | --- |
| **Шина** | Проста и дешева | Если кабель повреждён, сеть не работает |
| **Звезда** | Надёжность, простота управления | Если центральный узел выходит из строя, сеть отключается |
| **Кольцо** | Равномерное распределение нагрузки | Сеть прекращает работу при разрыве соединения |
| **Ячеистая** | Высокая надёжность | Дорогая и сложная в настройке |
| **Древовидная** | Масштабируемость | Сложность управления |
| **Смешанная** | Гибкость | Высокая стоимость и сложность |

### ****3.2 Основные протоколы и способы передачи данных****

В компьютерных сетях передача данных осуществляется с использованием различных протоколов – наборов правил, которые обеспечивают взаимодействие устройств.

## ****1. Способы передачи данных****

### ****1.1. По типу связи между устройствами****

* **Однонаправленная (Simplex)** – данные передаются только в одном направлении.
	+ Пример: радиовещание, телевидение.
* **Полудуплексная (Half-Duplex)** – данные передаются в обе стороны, но по очереди.
	+ Пример: рации, где один говорит, а другой слушает.
* **Полный дуплекс (Full-Duplex)** – данные передаются одновременно в обоих направлениях.
	+ Пример: телефонный разговор.

### ****1.2. По способу адресации****

* **Unicast** – передача данных от одного устройства к другому (точка-точка).
	+ Пример: просмотр веб-страницы.
* **Broadcast** – передача данных сразу всем устройствам в сети.
	+ Пример: отправка сообщений DHCP-серверу.
* **Multicast** – передача данных группе устройств.
	+ Пример: онлайн-трансляции.

## ****2. Основные сетевые протоколы****

### ****2.1. Протоколы модели OSI****

Модель OSI (Open Systems Interconnection) – эталонная модель, описывающая, как данные передаются по сети. Она включает 7 уровней:

1. **Физический уровень** – определяет физическое соединение (кабели, радиоволны).
2. **Канальный уровень** – отвечает за передачу кадров данных (Ethernet, Wi-Fi).
3. **Сетевой уровень** – маршрутизация данных между сетями (IP, ICMP).
4. **Транспортный уровень** – контроль доставки данных (TCP, UDP).
5. **Сеансовый уровень** – управление сессиями между приложениями.
6. **Представительный уровень** – кодирование и шифрование данных.
7. **Прикладной уровень** – взаимодействие с пользователем (HTTP, FTP).

### ****2.2. Основные протоколы TCP/IP****

Модель **TCP/IP** (интернет-протоколы) используется в реальных сетях и включает 4 уровня:

* **Прикладной уровень**
* **HTTP/HTTPS** – передача веб-страниц.
* **FTP** – передача файлов.
* **SMTP, POP3, IMAP** – передача электронной почты.
* **Транспортный уровень**
* **TCP (Transmission Control Protocol)** – надёжная передача данных.
* **UDP (User Datagram Protocol)** – быстрая передача без контроля ошибок (используется в видеозвонках, стримингах).
* **Сетевой уровень**
* **IP (Internet Protocol)** – маршрутизация пакетов данных (IPv4, IPv6).
* **ICMP (Internet Control Message Protocol)** – передача сообщений об ошибках (ping, traceroute).
* **Канальный уровень**
* **Ethernet** – проводная передача данных.
* **Wi-Fi** – беспроводная передача данных.

## ****3. Итоги****

| **Протокол** | **Уровень** | **Назначение** |
| --- | --- | --- |
| **HTTP/HTTPS** | Прикладной | Загрузка веб-страниц |
| **FTP** | Прикладной | Передача файлов |
| **SMTP, POP3, IMAP** | Прикладной | Электронная почта |
| **TCP** | Транспортный | Надёжная передача данных |
| **UDP** | Транспортный | Быстрая передача без контроля ошибок |
| **IP** | Сетевой | Маршрутизация пакетов данных |
| **ICMP** | Сетевой | Передача сообщений об ошибках |
| **Ethernet** | Канальный | Проводная передача данных |
| **Wi-Fi** | Канальный | Беспроводная передача данных |

### ****3.3 Основы кибербезопасности****

Кибербезопасность — это защита компьютерных систем, сетей и данных от атак, утечек и несанкционированного доступа.

## ****1. Основные угрозы в кибербезопасности****

* **Вредоносное ПО (Malware)** – программы, которые наносят вред системе.
* **Вирусы** – внедряются в файлы и распространяются при их запуске.
* **Черви** – размножаются без участия пользователя.
* **Трояны** – маскируются под полезные программы.
* **Шпионское ПО (Spyware)** – крадёт данные.
* **Выкупное ПО (Ransomware)** – шифрует файлы и требует выкуп.
* **Фишинг (Phishing)** – обман с целью получения конфиденциальной информации (логины, пароли, банковские данные).
* Пример: письмо от «банка» с просьбой ввести пароль.
* **Атаки на пароли**
* **Brute force** – подбор пароля методом перебора.
* **Словарная атака** – использование списка популярных паролей.
* **Перехват пароля** – через кейлоггеры или фишинговые сайты.
* **DDoS-атаки** – перегрузка сервера множеством запросов, чтобы сделать сайт недоступным.
* **Социальная инженерия** – манипуляции, заставляющие жертву раскрыть личные данные.
* Пример: звонок «из техподдержки» с просьбой назвать пароль.

## ****2. Методы защиты****

* **Использование надёжных паролей**
* Длина **12+ символов**.
* Сочетание **букв, цифр, спецсимволов**.
* Уникальный пароль для каждого сервиса.
* **Двухфакторная аутентификация (2FA)**
* Дополнительный код (SMS, приложение, ключ безопасности).
* **Антивирусы и брандмауэры**
* Антивирус: защита от вредоносного ПО.
* Брандмауэр (firewall): фильтрация входящего и исходящего трафика.
* **Осторожность в интернете**
* Проверять ссылки и отправителей e-mail.
* Не скачивать файлы из неизвестных источников.
* Обновлять ПО (устранение уязвимостей).
* **Шифрование данных**
* **HTTPS** – защищённый интернет-протокол.
* **VPN** – защита трафика от перехвата.
* **Резервное копирование**
* Хранение копий важных данных (в облаке, на внешних дисках).

## ****3. Итоги****

| **Угроза** | **Описание** | **Способы защиты** |
| --- | --- | --- |
| **Вирусы, трояны, шпионское ПО** | Вредоносные программы | Антивирус, брандмауэр |
| **Фишинг** | Обман для кражи данных | Проверка ссылок, 2FA |
| **DDoS-атака** | Перегрузка сервера | Использование защитных сервисов (Cloudflare) |
| **Атаки на пароли** | Подбор или перехват паролей | Длинные пароли, 2FA |
| **Социальная инженерия** | Манипуляция людьми | Внимательность, обучение |

### ****3.4 Работа с облачными сервисами****

Облачные сервисы — это технологии, позволяющие хранить данные и запускать приложения на удалённых серверах, доступ к которым осуществляется через интернет. Они широко используются для хранения файлов, работы с документами, анализа данных и запуска программного обеспечения без необходимости установки на локальные устройства.

## ****1. Виды облачных сервисов****

* **Хранение данных**
* Google Drive
* Dropbox
* OneDrive
* iCloud
* Яндекс.Диск
* **Облачные вычисления (Cloud Computing)**
* Amazon Web Services (AWS)
* Microsoft Azure
* Google Cloud Platform (GCP)
* **Облачные офисные приложения**
* Google Docs, Sheets, Slides
* Microsoft 365 (Word, Excel, PowerPoint)
* OnlyOffice
* **Платформы для резервного копирования**
* Backblaze
* Acronis
* Carbonite
* **Облачные платформы для разработчиков**
* GitHub
* GitLab
* Bitbucket

## ****2. Преимущества облачных сервисов****

✅ **Доступ отовсюду** – можно работать с любого устройства через интернет.
✅ **Безопасность** – данные защищены шифрованием и аутентификацией.
✅ **Автоматическое резервное копирование** – файлы не теряются при поломке устройства.
✅ **Экономия места** – нет необходимости хранить всё на жёстком диске.
✅ **Совместная работа** – можно редактировать файлы одновременно с другими пользователями.
✅ **Гибкость и масштабируемость** – можно быстро увеличить объём хранилища или вычислительных мощностей.

## ****3. Основные функции облачных сервисов****

* **Хранение и синхронизация данных**
* Файлы автоматически обновляются на всех устройствах.
* Можно настроить папки для автозагрузки в облако.
* **Совместная работа**
* В Google Docs и Microsoft 365 можно редактировать документы одновременно с коллегами.
* Можно оставлять комментарии и отслеживать изменения.
* **Автоматическое резервное копирование**
* Dropbox, Google Drive и iCloud позволяют настроить автоматическое копирование данных.
* **Доступ к мощным вычислениям**
* В AWS, Azure и Google Cloud можно запускать виртуальные машины и базы данных.

## ****4. Основы безопасности в облаке****

🔒 **Используйте сложные пароли и 2FA**
🔒 **Шифруйте важные файлы перед загрузкой**
🔒 **Ограничивайте доступ к файлам по ролям**
🔒 **Регулярно проверяйте настройки конфиденциальности**
🔒 **Используйте VPN при доступе из общественных Wi-Fi**

## ****5. Популярные облачные сервисы и их возможности****

| **Сервис** | **Основное назначение** | **Бесплатный объём** |
| --- | --- | --- |
| **Google Drive** | Хранение файлов, документы | 15 ГБ |
| **Dropbox** | Файлы, резервное копирование | 2 ГБ |
| **OneDrive** | Интеграция с Windows, Office 365 | 5 ГБ |
| **iCloud** | iPhone, Mac-устройства | 5 ГБ |
| **AWS** | Облачные вычисления, базы данных | 12 месяцев бесплатно (ограниченные ресурсы) |
| **Google Cloud** | Машинное обучение, виртуальные машины | $300 на пробный период |
| **GitHub** | Код и совместная разработка | Неограниченные публичные репозитории |

### ****Вывод****

Облачные сервисы значительно упрощают работу с данными, позволяют совместно редактировать файлы, экономят место на устройствах и обеспечивают безопасность данных. Однако важно соблюдать правила безопасности: использовать сложные пароли, включать двухфакторную аутентификацию и шифровать конфиденциальные файлы.

### ****4.1 Форматы файлов и их особенности****

Формат файла определяет способ хранения и представления данных внутри него. В зависимости от назначения файлы могут быть текстовыми, графическими, аудио, видео и другими.

## ****1. Основные категории файлов****

📄 **Текстовые файлы**
🎨 **Графические файлы**
🎵 **Аудиофайлы**
🎥 **Видеофайлы**
📦 **Архивы и сжатые файлы**
💾 **Исполняемые файлы и программный код**
📊 **Документы и таблицы**

## ****2. Текстовые файлы****

Используются для хранения текстовой информации.

| **Формат** | **Описание** | **Особенности** |
| --- | --- | --- |
| **.txt** | Обычный текст | Не содержит форматирования |
| **.docx** | Документ MS Word | Поддержка форматирования, картинок |
| **.pdf** | Электронный документ | Нельзя легко редактировать, но удобно читать |
| **.rtf** | Форматированный текст | Поддержка разных шрифтов и стилей |
| **.md** | Markdown | Используется для разметки текста |

✍ **Примечание**: Файлы .txt занимают мало места, но не поддерживают стили и изображения, в отличие от .docx или .pdf.

## ****3. Графические файлы****

Используются для хранения изображений.

| **Формат** | **Описание** | **Особенности** |
| --- | --- | --- |
| **.jpg/.jpeg** | Сжатое изображение | Потеря качества при каждом сохранении |
| **.png** | Изображение с прозрачностью | Без потерь, но больше размер |
| **.gif** | Анимации | Ограниченная цветовая палитра (256 цветов) |
| **.bmp** | Несжатое изображение | Высокое качество, но большой размер |
| **.svg** | Векторное изображение | Масштабируется без потери качества |

🎨 **Совет**: Используйте .png для качественных изображений с прозрачностью, а .jpg для фото, где важен баланс между качеством и размером.

## ****4. Аудиофайлы****

Форматы для хранения звука.

| **Формат** | **Описание** | **Особенности** |
| --- | --- | --- |
| **.mp3** | Сжатый звук | Потеря качества, но маленький размер |
| **.wav** | Несжатый звук | Высокое качество, но большой размер |
| **.flac** | Сжатие без потерь | Отличное качество, меньший размер чем .wav |
| **.ogg** | Альтернатива .mp3 | Открытый формат, лучшее сжатие |

🎧 **Примечание**: .mp3 удобен для большинства случаев, а .flac предпочтителен для любителей качественного звука.

## ****5. Видеофайлы****

Форматы для хранения видео.

| **Формат** | **Описание** | **Особенности** |
| --- | --- | --- |
| **.mp4** | Популярный формат видео | Хорошее сжатие, совместимость с большинством устройств |
| **.avi** | Старый, но распространённый формат | Большой размер файлов |
| **.mkv** | Поддержка субтитров и многодорожечного звука | Гибкость, но не все устройства поддерживают |
| **.mov** | Формат Apple | Используется в MacOS и iOS |
| **.webm** | Формат для веба | Оптимизирован для интернета |

📺 **Совет**: Используйте .mp4 для совместимости и хорошего качества, а .mkv для фильмов с субтитрами.

## ****6. Архивы и сжатые файлы****

Используются для упаковки и сжатия данных.

| **Формат** | **Описание** | **Особенности** |
| --- | --- | --- |
| **.zip** | Популярный архив | Поддержка во всех ОС |
| **.rar** | Альтернативный архив | Лучшее сжатие, но требует WinRAR |
| **.7z** | Улучшенное сжатие | Лучше .zip и .rar, но меньше поддерживается |
| **.tar.gz** | Архив для Linux | Используется в Unix-системах |

📦 **Примечание**: Используйте .zip для универсальной совместимости, а .7z для максимального сжатия.

## ****7. Исполняемые файлы и программный код****

Программы и скрипты.

| **Формат** | **Описание** | **Особенности** |
| --- | --- | --- |
| **.exe** | Исполняемый файл Windows | Запускает программы |
| **.bat** | Скрипт командной строки Windows | Автоматизация задач |
| **.sh** | Скрипт командной строки Linux | Аналог .bat для Unix |
| **.py** | Программа на Python | Требует интерпретатора |
| **.html** | Веб-страница | Открывается в браузере |

💻 **Совет**: Осторожно открывайте .exe-файлы из ненадёжных источников — возможны вирусы.

## ****8. Форматы документов и таблиц****

Используются для офисных задач.

| **Формат** | **Описание** | **Особенности** |
| --- | --- | --- |
| **.xls / .xlsx** | Таблицы Excel | Поддержка формул и макросов |
| **.csv** | Таблица в виде текста | Простота, совместимость, но без форматирования |
| **.ppt / .pptx** | Презентации PowerPoint | Поддержка анимации и графики |
| **.odt** | Альтернативный текстовый документ | Поддержка в LibreOffice |

📊 **Примечание**: Используйте .xlsx для сложных таблиц, а .csv для обмена данными между разными программами.

## ****Вывод****

Формат файла определяет, как именно хранятся данные. Выбор зависит от задачи:

* **Текст** → .txt, .docx, .pdf
* **Изображения** → .png, .jpg, .svg
* **Аудио** → .mp3, .wav, .flac
* **Видео** → .mp4, .mkv
* **Архивы** → .zip, .rar, .7z
* **Исполняемые файлы** → .exe, .sh, .py
* **Таблицы и документы** → .xlsx, .csv, .pptx

### ****4.2 Основы работы с графикой и мультимедиа****

Графика и мультимедиа охватывают широкий спектр технологий, используемых для создания, редактирования и обработки изображений, видео и звука.

## ****1. Виды компьютерной графики****

Компьютерная графика делится на два основных типа:

📌 **Растровая графика**
📌 **Векторная графика**

### ****Растровая графика****

📷 Используется для работы с фотографиями и сложными изображениями.
🔸 Состоит из пикселей (квадратных точек).
🔸 Теряет качество при увеличении.

**Форматы:**

* .jpg (сжатие с потерями, фотографии)
* .png (прозрачность, без потерь)
* .gif (анимация, 256 цветов)
* .bmp (несжатый, большой размер)

**Программы:**
✅ Adobe Photoshop
✅ GIMP
✅ Paint.NET

### ****Векторная графика****

🖊 Используется для логотипов, схем, иконок.
🔸 Создаётся с помощью математических формул.
🔸 Масштабируется без потери качества.

**Форматы:**

* .svg (основной формат для веба)
* .ai (Adobe Illustrator)
* .eps (графика для печати)

**Программы:**
✅ Adobe Illustrator
✅ CorelDRAW
✅ Inkscape

## ****2. Основы работы с видео****

🎥 Видеофайлы состоят из множества изображений (кадров), сменяющихся с определённой скоростью (например, 30 кадров в секунду).

### ****Форматы видео****

* .mp4 – универсальный формат, поддерживается на всех устройствах.
* .mkv – поддерживает несколько дорожек звука и субтитры.
* .avi – устаревший, но всё ещё используется.
* .mov – формат Apple.
* .webm – оптимизирован для веба.

### ****Программы для редактирования видео****

✅ Adobe Premiere Pro
✅ DaVinci Resolve
✅ Sony Vegas Pro
✅ iMovie (Mac)

### ****Основные понятия монтажа****

🎞 **Таймлайн** – временная шкала, где располагаются клипы.
🎬 **Кадрирование** – изменение размера и положения видео.
🔊 **Саунд-дизайн** – работа со звуком и музыкой.
🟢 **Цветокоррекция** – изменение цветового баланса.

## ****3. Основы работы со звуком****

🎵 Аудио используется в мультимедиа-проектах для создания атмосферы и передачи эмоций.

### ****Форматы аудиофайлов****

* .mp3 – сжатый формат, небольшой размер.
* .wav – несжатый, высокое качество.
* .flac – сжатие без потерь, аудиофилы его любят.
* .ogg – альтернатива .mp3.

### ****Программы для работы со звуком****

✅ Adobe Audition
✅ Audacity (бесплатная)
✅ FL Studio (для создания музыки)

### ****Основные параметры звука****

🔊 **Частота дискретизации** (44.1 кГц – стандарт для музыки).
🔉 **Битрейт** (192 кбит/с – хорошее качество для .mp3).
🎚 **Громкость и баланс** – регулировка звуковых дорожек.

## ****4. Интерактивные мультимедиа****

📲 **Интерактивные элементы**: анимации, презентации, видеоигры.

### ****Инструменты для создания мультимедиа****

✅ Adobe Animate – анимация и интерактивные ролики.
✅ Unity / Unreal Engine – разработка игр и 3D-графики.
✅ Blender – 3D-моделирование и анимация.
✅ PowerPoint / Keynote – создание презентаций.

## ****Вывод****

Работа с графикой и мультимедиа требует знаний форматов, инструментов и принципов обработки.
🔹 **Растровая графика** – фотографии и сложные изображения.
🔹 **Векторная графика** – логотипы и иконки.
🔹 **Видео** – монтаж, цветокоррекция, спецэффекты.
🔹 **Аудио** – музыка, эффекты, озвучка.
🔹 **Интерактивные мультимедиа** – анимации, игры, презентации.

**4.3 Создание и редактирование презентаций, настройка анимации**

 Существует множество видов презентаций, в которых используются как слайды, содержащие теоретическую часть, так и видеоролики, музыкальные композиции и голосовое сопровождение. Самой популярной программой для их создания считается PowerPoint.

 Создание презентаций – это дело не лёгкое, как может показаться на первый взгляд. Первое, что потребуется, чтобы создать презентацию – программа PowerPoint. Как правило, она является частью пакета Microsoft (Word и Excel). Если их нет, то придётся для начала скачать их на свой компьютер с официального сайта Microsoft.

**Подготовка к созданию презентации**

 Чтобы материал получился действительно красивым, требуется тщательная подготовка. Во-первых, составить план презентации. Это поможет определиться с количеством слайдов и их оформлением. Во-вторых, заранее подобрать картинки и составить тексты. После того как всё будет готово, можно приступать непосредственно к созданию презентации.

 Надо хорошо продумать содержание будущей презентации. Определиться с целью, структурой и содержанием. Выяснить особенности аудитории. Внимание слушателей быстро пропадает и поэтому необходимо сделать к определенному времени динамичный сюжет. Если с помощью презентации идёт объяснение нового материала, то темп смены слайдов нужно "притормозить".

**Подбор материала**

Первое, что потребуется для наполнения слайдов – тексты. Брать их можно как «из головы», излагая собственный опыт, так и из различных источников: книг, рефератов, научных или исследовательских работ.

 Так как создать красивую презентацию без картинок не представляется возможным, выбирать их стоит особенно тщательно. Для этого можно использовать любую поисковую систему или личные фото.

 Графики, диаграммы и таблицы для презентаций можно составить с помощью специальных программ, конвертировав затем изображение в формат jpg, png или gif. Также можно нарисовать графики вручную, используя обычную бумагу и фломастеры. Затем это изображение нужно отсканировать или сфотографировать. Программа PowerPoint также содержит множество шаблонов, которые можно адаптировать для конкретной темы.

**Создание слайдов**

 На компьютере (на рабочем столе) создаём документ PowerPoint и открываем его. Открываем вкладку под названием «Главная» и находим в её верхней левой части кнопку «Создать слайд». Для начала делаем титульный слайд презентации, для чего выбираем из предложенного списка соответствующий слайд. Далее добавляем наиболее подходящие по форме и расположению изображений и текстов слайды.

 Открыв раздел «Дизайн» на верхней панели окна программы, выбираем цветовое оформление. Правее каталога тем расположены кнопка «Цвета». Предназначена она для изменения [цвета фона.](http://fb.ru/article/138218/kak-izmenit-tsvet-fona-na-sayte) В разделе «Стили фона» можно изменить цвет нижней части слайдов.

 Если в ходе создания презентации обнаружили, что один из них подобран неверно, его можно изменить. Делается это так: переходим в раздел «Главная», выделяем нужный слайд и нажимаем расположенную правее кнопки создания слайдов сноску «Макет». Перед нами открывается набор стандартных слайдов, из которых выбираем подходящий нам вариант.

Используйте для создания презентации слайды с одинаковым цветовым решением.

**Текст слайда**

 Работа с текстом презентации по редактированию и форматированию построена на тех же принципах, что и работа в Microsoft Word. Работа похожа на технологию WorЧтобы вставить текст, следует поставить курсор в поле "Заголовок слайда" или "Текст слайда", а после на вкладке "Главная" войти в группу "Шрифт".

 Расположение заголовков и фраз, а также их оформление (шрифт) можно изменить тем же способом, как и в обычных текстовых документах.

 Далее нам потребуется немного фантазии, потому что на этом этапе стоит узнать, как создать красивую презентацию, в которой заголовки будут оформлены необычно. Для того чтобы сделать это, нужно выделить поле для заголовка и ввести название слайда. Далее переходим в раздел «Формат» на главной странице программы. Выбираем оригинальное оформление для заголовка, подбираем более подходящее цветовое решение (в выпадающем меню) и сохраняем его.

Не надо размещать на одном слайде кучу текста и рисунков. Это сильно отвлекает внимание и ухудшает его концентрацию. Текст и фон должны хорошо контрастировать друг с другом, иначе информацию будет сложно прочитать. Для разных типов объектов следует использовать разные размеры шрифта. Заголовок слайда оформляем размером шрифта 22-28, подзаголовок – 20-24, текст, заголовки осей в диаграммах, информацию в таблицах – 18-22.

**Размещение картинок, видеороликов и графики**

 Заголовок слайРазмещение различных схем, фотографий и рисунков доступно во вкладке "Вставка". Переходим во вкладку «Вставка», выбираем раздел «Диаграммы». В открывшемся окне выбираем подходящие по внешнему виду элементы и сохраняем результат. При необходимости вводим показатели, которые будут отображать значение каждого сектора.

 Таблицы на слайде можно разместить так же, как диаграммы. Для этого достаточно найти раздел «Вставить таблицу» и выбрать необходимое количество столбцов и строк в ней.

 Прямо на выбранном слайде кликнуть по значку таблицы, диаграммы, видеоролика или картинки, после чего откроется окно, в котором можно будет осуществить переход к папке, в которой сохранены изображения для презентации.

 Так как создать правильную презентацию – это самая важная задача, запомните такой совет: никогда не размещайте на одном слайде более двух изображений! В противном случае аудитория не сможет или не успеет их рассмотреть.

Используйте фотографии и картинки только хорошего качества, чтобы не испортить эффект от презентации.

## Звуковое сопровождение

 Размещение фотограф Чтобы разместить аудиозапись или видеозапись на одном из слайдов, переходим в раздел «Вставка» и на верхней панели ищем блок «Клипы мультимедиа». Выбираем, что будем загружать и откуда: из файла, организатора клипов или напрямую из интернета. Также программа предусматривает [запись звука](http://fb.ru/article/35287/kak-osuschestvit-zapis-zvuka-v-domashnih-usloviyah) в режиме реального времени.

 После этого устанавливаем вариант воспроизведения звука: автоматически или по щелчку.

**Настройка анимации**

 Эффекты анимации, добавляют разные автоматические действия, происходящие на слайде. Для этого на вкладке "Анимация" переходим в группу "Анимация" и открываем область задач "Настройка анимации". После этого щёлкаем по объекту, которому надо задать изменение. В области задач "Настройка анимации" нажимаем кнопку "Добавить эффект", а после выполняем действия по применению эффектов. В поле "Изменение эффекта" указывают начало анимации, её направление и скорость изменения.

 Переходы для смены слайдов делают презентацию PowerPoint более эффектной. Для добавления одинаковых переходов между слайдами, на вкладке "Анимация" щелкаем по эскизу слайда и в группе "Переход к следующему слайду" выбираем эффект смены слайдов.

 Для установки скорости смены слайдов, в группе "Переход к следующему слайду" раскрываем кнопку «Скорость перехода», а затем задаем нужную скорость. В группе "Смена слайда" указываем порядок смены: автоматически или по щелчку.

 К переходам между слайдами можно добавить звук. На вкладке "Анимация" в группе "Переход к следующему слайду" раскрываем кнопку "Звук перехода" и добавляем звук из списка. Для добавления звук, которого нет в списке, выбираем команду "Другой звук".

 Слишком частое использование разного звукового сопровождения переходов слайдов ухудшает восприятие содержания. Для звукового сопровождения выбирайте спокойную инструментальную или классическую музыку. Это не будет отвлекать слушателей от содержания презентации.

 Для просмотра получившейся презентации, в правом нижнем углу нажимаем кнопку «Показ слайдов».

Не следует перенасыщать презентацию эффектами анимации. Большое количество мигающих и скачущих объектов, внезапных звуков, картинок с анимацией лишь отвлечет аудиторию. Анимация используется по минимуму с полезной функциональной нагрузкой.

## Настройки показа

 Последний, и самый важный этап в создании красивой и качественной презентации – настройка времени отображения каждого слайда, а также временного промежутка, когда будут воспроизводиться видеоролики. Настройки можно выставить в разделе «Показ слайдов» в меню «Настройка демонстрации». При этом важно постараться читать вслух речь, которой вы собираетесь описывать происходящее на экране. Чтобы не ошибиться с временными рамками, включите «Настройки времени», и не торопясь произнесите текст. Программа автоматически засечёт время, потраченное вами на комментирование одного слайда, а вам останется только сохранить результат.

 Откорректированную презентацию сохраняем в какой-нибудь папке на жестком диске через команду "Сохранить как" - "Презентация PowerPoint".

В настоящее время значимость презентаций достаточно высока. Поэтому умение её создавать ещё не раз пригодится в жизни. Осталось дело за малым - научиться создавать.

5.1 **Процесс разработки и защиты мини-проекта включает несколько ключевых этапов:**

**1. Определение темы и целей проекта**

* Выбор актуальной и интересной темы.
* Постановка целей и задач проекта.
* Определение ожидаемых результатов.

**2. Планирование работы**

* Разработка структуры проекта (из каких частей он будет состоять).
* Определение сроков выполнения каждого этапа.
* Распределение обязанностей (если работа командная).

**3. Сбор информации и ресурсов**

* Изучение теоретической базы.
* Поиск примеров, кейсов, исследований.
* Подготовка необходимых инструментов и материалов.

**4. Разработка проекта**

* Создание прототипа, макета или программы (если технический проект).
* Написание текста, подготовка графики, видео или других материалов.
* Проведение тестирования, проверка работы.

**5. Оформление проекта**

* Написание отчета (введение, основная часть, выводы).
* Подготовка презентации (слайды, схемы, графики).
* Дизайн материалов (если требуется).

**6. Подготовка к защите**

* Репетиция выступления.
* Подготовка ответов на возможные вопросы.
* Тренировка уверенного поведения.

**7. Защита проекта**

* Четкое и структурированное изложение идей.
* Демонстрация результатов (видео, код, макеты, графики).
* Ответы на вопросы комиссии или жюри.

### ****План реализации мини-проекта по информатике****

#### ****1. Определение темы и целей проекта****

* Выбор темы проекта (например, разработка веб-приложения, чат-бота, игры, базы данных и т. д.).
* Постановка целей (что должен уметь делать проект, какую проблему решает).
* Определение ожидаемого результата.

#### ****2. Анализ и планирование****

* Анализ аналогичных проектов и поиск решений.
* Определение необходимых технологий (язык программирования, фреймворки, базы данных и т. д.).
* Разработка структуры проекта (основные модули, компоненты, алгоритмы).
* Распределение обязанностей (если проект командный).
* Определение сроков выполнения каждого этапа.

#### ****3. Разработка проекта****

* Создание базовой структуры (папки, файлы, настройка среды разработки).
* Реализация основных функций проекта (написание кода, создание базы данных, разработка интерфейса).
* Тестирование каждого компонента.

#### ****4. Оформление проекта****

* Написание документации (описание проекта, инструкции по установке и использованию).
* Подготовка презентации (слайды, графики, демонстрация работы проекта).

#### ****5. Тестирование и отладка****

* Проверка функциональности проекта.
* Исправление багов и оптимизация кода.
* Тестирование на разных устройствах или платформах (если применимо).

#### ****6. Подготовка к защите****

* Репетиция презентации.
* Подготовка ответов на возможные вопросы.
* Финальная проверка проекта.

#### ****7. Защита проекта****

* Краткое, четкое объяснение идеи и процесса разработки.
* Демонстрация работы проекта.
* Ответы на вопросы комиссии или преподавателя.

**Заключение**

Данное методическое пособие по информатике для 7 класса разработано с целью облегчения процесса преподавания и обучения, а также формирования у учащихся прочных знаний и навыков работы с информационными технологиями.

В пособии рассмотрены основные темы курса, включая алгоритмы и программирование, работу с текстовыми и графическими редакторами, основы компьютерных сетей и информационной безопасности. Материал представлен в доступной форме, с примерами и практическими заданиями, что способствует развитию логического мышления, алгоритмического подхода и цифровой грамотности учащихся.

Использование данного пособия позволит учителям эффективно организовать учебный процесс, а учащимся — систематизировать и углубить знания по информатике, подготовиться к выполнению практических и проектных работ.

Мы надеемся, что данное пособие станет полезным инструментом в обучении, поможет заинтересовать школьников предметом и заложит основу для их дальнейшего изучения современных информационных технологий.