

1. Билет	1. Билет
1. Основные понятия информатики: информация, информационные системы, информационные технологии. Каналы передачи данных. Источник, приемник информации. Виды информации. Свойства информации.	<p>1. Основные понятия информатики: информация, информационные системы, информационные технологии. Каналы передачи данных. Источник, приемник информации. Виды информации. Свойства информации.</p> <ul style="list-style-type: none"> - Информация — это данные, которые имеют смысл и могут быть использованы для принятия решений. - Информационные системы — это системы, которые обрабатывают, хранят и передают информацию. - Информационные технологии — это методы и инструменты, применяемые для обработки и передачи информации. - Каналы передачи данных — это способы передачи информации, например, проводные и беспроводные сети. - Источник информации — это место, откуда поступает информация, а приемник — это место, куда она поступает. - Виды информации: текстовая, числовая, графическая, аудио и видео информация. - Свойства информации: достоверность, актуальность, полнота, доступность, структурированность.
2. Создать и отредактировать однотабличную базу данных в MS Access.	<p>2. Создать и отредактировать однотабличную базу данных в MS Access:</p> <ul style="list-style-type: none"> - Открыть MS Access и создать новую базу данных. - Добавить таблицу, определить поля (например, ID, Название, Описание). - Заполнить таблицу данными и сохранить изменения. - Отредактировать существующие записи или добавить новые.
2. Билет	2. Билет
1. Системы счисления. <ul style="list-style-type: none"> • Двоичная, • восьмеричная, • шестнадцатеричная 	<p>1. Системы счисления:</p> <ul style="list-style-type: none"> - Двоичная система — основана на двух цифрах (0 и 1). - Восьмеричная система — основана на восьми цифрах (0-7). - Шестнадцатеричная система — основана на шестнадцати цифрах (0-9 и A-F).
2. Создать табличный документ в MS Excel.	<p>2. Создать табличный документ в MS Excel:</p> <ul style="list-style-type: none"> - Открыть MS Excel и создать новый документ. - Ввести данные в ячейки, применить формулы и функции, отформатировать таблицу.
3. Билет	3. Билет
1. Арифметические операции в позиционных системах счисления.	<p>Сложение: поразрядно, начиная с младших разрядов, перенос при превышении основания системы.</p> <p>Вычитание: поразрядно, заем из старшего разряда при меньшем разряде уменьшаемого.</p> <p>Умножение: каждый разряд одного числа умножается на каждый разряд второго числа, результаты сдвигаются влево и складываются.</p>

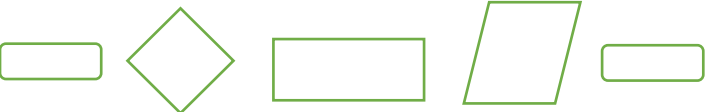
	Деление: делимое разбивается поразрядно, операция деления выполняется для каждого разряда до остатка.
2. Создать табличный документ в MS Excel	2. Создать табличный документ в MS Excel: - Создать новую таблицу, ввести данные и применить формулы для арифметических операций.
4. Билет	4. Билет
1. Перевод из десятичной системы счисления в двоичную, восьмеричную, шестнадцатеричную и обратный перевод.	1. Перевод из десятичной системы счисления в двоичную, восьмеричную, шестнадцатеричную и обратный перевод: $123_{10}/2 (1)$ $61/2 (1)$ $30/2 (0)$ $15/2 (1)$ $7/2 (1)$ $3/2 (1)$ (1) $123_{10}=1111011_2=2^6*1+2^5*1+2^4*1+2^3*1+2^2*0+2^1*1+2^0*1$ $123_{10}/8 (3)$ $15/8 (7)$ $(1) \quad 173_8=8^2*1+8^1*7+8^0*3=123_{10}$ $123_{10}/16 (B)$ (7) $7B_{16}=16^1*7+16^0*11=123_{10}$
2. Создать презентацию в MS Power Point	2. Создать презентацию в MS Power Point: - Открыть PowerPoint, создать новую презентацию, добавить слайды, текст и графику.
5. Билет	5. Билет
1. Операции над машинными кодами. Прямой, обратный, дополнительный код.	Прямой код используется для представления чисел с указанием их знака. Обратный код представляет отрицательные числа как инверсию всех битов прямого кода (кроме знакового разряда). Дополнительный код используется для выполнения операций над числами, чтобы избежать необходимости учитывать знак отдельно. Чтобы получить дополнительный код, нужно прибавить 1 к обратному коду отрицательного числа.
2. Выполнить задание по созданию, редактированию и форматированию текстового документа в MS Word	2. Создание текстового документа в MS Word: - Открыть Word, создать новый документ, ввести текст, отредактировать и отформатировать его.
6. Билет	6. Билет
1. Компьютер как техническое средство реализации технологий. Процессор, память компьютера, контроллеры, шины, платы. ЭВМ: понятие. Принципы Дж. фон Неймана.	1. Компьютер как техническое средство реализации технологий: 1. Компьютер: Комплекс технических средств и программного обеспечения, способный реализовать любой алгоритм, оформленный в виде программы, хранимой в памяти.

	<p>2. Центральный процессор (ЦП): Основной компонент компьютера, выполняющий арифметические и логические операции, управляющий вычислительным процессом и координирующий работу всех устройств.</p> <p>3. Память компьютера: Компонент, запоминающий информацию, построенный из двоичных запоминающих элементов (битов), объединенных в группы по 8 битов, называемые байтами.</p> <p>Контроллер – это специальный процессор в компьютере, который управляет внешними устройствами.</p> <p>Шина — соединение, служащее для передачи данных между функциональными блоками компьютера.</p> <p>Плата – представляющая собой основу конструкций различных электронных устройств.</p> <p>Электронная вычислительная машина (ЭВМ): Комплекс технических средств, в котором основные функциональные элементы (логические, запоминающие, индикационные и др.) выполнены на электронных компонентах и предназначены для автоматической обработки информации при решении вычислительных и информационных задач.</p> <p>Принципы Дж. фон Неймана</p> <ol style="list-style-type: none">1. Принцип двоичности: Данные и команды представляются в двоичной системе.2. Принцип программного управления: Программа состоит из последовательности команд, выполняемых процессором.3. Принцип однородности памяти: Программы и данные хранятся в одной памяти и могут обрабатываться одинаково.4. Принцип адресуемости памяти: Память состоит из пронумерованных ячеек, к которым процессор имеет произвольный доступ.5. Принцип последовательного программного управления: Команды выполняются последовательно, одна за другой.6. Принцип условного перехода: Выполнение команд может изменяться в зависимости от условий, определяемых данными.																															
2. Выполнить задание по созданию, редактированию и форматированию текстового документа в MS Word	2. Создание текстового документа в MS Word: - Создать и отредактировать документ с использованием различных инструментов форматирования.																															
7. Билет	7. Билет																															
<p>1. Логические элементы компьютера. Таблицы истинности основных логических операций.</p> <p>Логические элементы компьютера — это базовые компоненты, которые выполняют логические операции над двоичными значениями (0 и 1). Основные логические операции включают:</p> <ol style="list-style-type: none">1. AND (И): Возвращает 1, если оба входа равны 1.2. OR (ИЛИ): Возвращает 1, если хотя бы один из входов равен 1.	<table><tr><td colspan="3">Таблицы истинности</td><td colspan="2">3. NOT (НЕ)</td><td rowspan="5">Эти логические операции и таблицы истинности являются основой для построения более сложных логических схем и вычислительных</td></tr><tr><td colspan="3">1. AND (И)</td><td colspan="2"></td></tr><tr><td>A</td><td>B</td><td>A AND B</td><td>A</td><td>NOT A</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td colspan="2">2. OR (ИЛИ)</td></tr></table>	Таблицы истинности			3. NOT (НЕ)		Эти логические операции и таблицы истинности являются основой для построения более сложных логических схем и вычислительных	1. AND (И)					A	B	A AND B	A	NOT A	0	0	0	0	1	0	1	0	1	0	1	0	0	2. OR (ИЛИ)	
Таблицы истинности			3. NOT (НЕ)		Эти логические операции и таблицы истинности являются основой для построения более сложных логических схем и вычислительных																											
1. AND (И)																																
A	B	A AND B	A	NOT A																												
0	0	0	0	1																												
0	1	0	1	0																												
1	0	0	2. OR (ИЛИ)																													

3. NOT (HE) : Инвертирует значение входа (0 становится 1, а 1 становится 0).	<table><tr><td>1</td><td>1</td><td>1</td></tr></table>	1	1	1	<table><tr><th>A</th><th>B</th><th>A OR B</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	A OR B	0	0	0	0	1	1	1	0	1	1	1	1	процессов в компьютерах.
1	1	1																			
A	B	A OR B																			
0	0	0																			
0	1	1																			
1	0	1																			
1	1	1																			
2. Выполнить задание на создание и редактирование табличного документа в MS Excel																					
8. Билет	8. Билет																				
1. Характеристика и классификация программных средств. Прикладное программное обеспечение.	Программные средства — это компьютерные программы, представленные на языке программирования или в машинном коде описания действий, которые должна выполнить ЭВМ в соответствии с алгоритмом решения конкретной задачи или группы задач. Классификация программных средств - Системное, Прикладное, Инструментальное. Прикладное программное обеспечение (ППО) — класс программ, предназначенный для решения практических задач и предназначенный на непосредственное взаимодействие с пользователями.																				
2. Выполнить задание на создание и редактирование табличного документа в MS Excel	2. Создание табличного документа в MS Excel: - Ввод данных о программных средствах и их классификации.																				
9. Билет	9. Билет																				
1. Программные средства общего назначения. Системное программное обеспечение. Программные средства общего назначения и системное программное обеспечение играют важную роль в функционировании компьютеров и других вычислительных систем. Давайте рассмотрим каждую из этих категорий подробнее. Программные средства общего назначения Программные средства общего назначения — это программы, которые могут использоваться для выполнения широкого спектра задач и не привязаны к конкретной области применения. Они предназначены для облегчения работы пользователей в различных сферах. К таким средствам относятся: 1. Текстовые редакторы: ○ Примеры: Microsoft Word, Google Docs, LibreOffice Writer. ○ Используются для создания и редактирования текстовых документов. 2. Табличные процессоры: ○ Примеры: Microsoft Excel, Google Sheets, LibreOffice Calc. ○ Используются для работы с таблицами, анализа данных и выполнения вычислений. 3. Программы для презентаций: ○ Примеры: Microsoft PowerPoint, Google Slides, LibreOffice Impress. ○ Используются для создания презентаций и визуализации информации. 4. Графические редакторы:	1. Программные средства общего назначения: - Системное программное обеспечение и его функции. Системное программное обеспечение — это набор программ, который управляет аппаратными ресурсами компьютера и предоставляет платформу для запуска прикладного программного обеспечения. Оно обеспечивает взаимодействие между аппаратным обеспечением и программами, а также выполняет основные функции системы. Основные компоненты системного программного обеспечения включают: 1. Операционные системы: ○ Примеры: Windows, macOS, Linux, Android. ○ Управляют аппаратными ресурсами, обеспечивают интерфейс для пользователя и запускают прикладные программы. 2. Драйверы устройств: ○ Примеры: драйверы для принтеров, видеокарт, сетевых адаптеров. ○ Обеспечивают взаимодействие операционной системы с аппаратными устройствами. 3. Системные утилиты: ○ Примеры: антивирусные программы, программы для резервного копирования, утилиты для управления дисками.																				



<ul style="list-style-type: none"> ○ Примеры: Adobe Photoshop, GIMP, CorelDRAW. ○ Используются для редактирования изображений и создания графики. <p>5. Веб-браузеры:</p> <ul style="list-style-type: none"> ○ Примеры: Google Chrome, Mozilla Firefox, Microsoft Edge. ○ Используются для доступа к интернету и просмотра веб-страниц. <p>6. Электронные таблицы:</p> <ul style="list-style-type: none"> ○ Примеры: Microsoft Access, MySQL Workbench. ○ Используются для работы с базами данных и управления информацией. 	<ul style="list-style-type: none"> ○ Предоставляют дополнительные функции для управления, настройки и защиты системы. <p>4. Средства разработки:</p> <ul style="list-style-type: none"> ○ Примеры: компиляторы, интерпретаторы, интегрированные среды разработки (IDE) такие как Visual Studio, Eclipse. ○ Используются для создания, тестирования и отладки программного обеспечения. <p>5. Сетевые операционные системы:</p> <ul style="list-style-type: none"> ○ Примеры: Windows Server, Linux Server. ○ Обеспечивают управление сетевыми ресурсами и службами, такими как файловые и печатные серверы. <p>Программные средства общего назначения и системное программное обеспечение являются основными компонентами современного вычислительного окружения. Первые предоставляют пользователям инструменты для выполнения различных задач, в то время как вторые обеспечивают функционирование и управление аппаратными ресурсами компьютера. Вместе они создают основу для работы и взаимодействия с вычислительными системами.</p>
<p>2. Выполнить задание на создание и редактирование табличного документа в MS Excel</p> <p>Шаг 1: Открытие Microsoft Excel</p> <p>Запустите Microsoft Excel на вашем компьютере.</p> <p>Выберите "Создать" или "Новый документ", чтобы открыть пустую книгу.</p> <p>Шаг 2: Создание таблицы</p> <p>В первой строке введите заголовки столбцов. Например:</p> <p>A1: "Дата"</p> <p>B1: "Описание"</p> <p>C1: "Сумма"</p> <p>D1: "Категория"</p> <p>Введите данные в следующие строки. Например:</p> <p>A2: "2023-10-01"</p> <p>B2: "Продукты"</p> <p>C2: "1500"</p> <p>D2: "Питание"</p> <p>A3: "2023-10-03"</p> <p>B3: "Транспорт"</p> <p>C3: "500"</p> <p>D3: "Транспорт"</p> <p>A4: "2023-10-05"</p> <p>B4: "Развлечения"</p> <p>C4: "2000"</p> <p>D4: "Развлечения"</p>	<p>Шаг 3: Форматирование таблицы</p> <p>Выделите заголовки (A1:D1).</p> <p>На вкладке "Главная" выберите "Жирный шрифт" для выделения заголовков.</p> <p>Вы можете изменить цвет фона заголовков, выбрав "Заливка" и выбрав желаемый цвет.</p> <p>Выделите всю таблицу (A1:D4) и выберите "Вставить" -> "Таблица". Убедитесь, что опция "Таблица с заголовками" отмечена.</p> <p>Шаг 4: Добавление формул</p> <p>Если вы хотите подсчитать общую сумму расходов, выберите ячейку, например, C5.</p> <p>Введите формулу: =СУММ(C2:C4) и нажмите Enter. Это даст вам общую сумму всех расходов.</p> <p>Шаг 5: Сохранение документа</p> <p>Нажмите "Файл" -> "Сохранить как".</p> <p>Выберите место для сохранения (например, "Мой компьютер" или "OneDrive").</p> <p>Введите имя файла (например, "Учёт_расходов") и нажмите "Сохранить".</p> <p>Шаг 6: Редактирование таблицы</p> <p>Чтобы добавить новые расходы, просто введите данные в следующую строку (например, A5, B5, C5, D5).</p> <p>Чтобы изменить данные, дважды щелкните по нужной ячейке и введите новые значения.</p> <p>Вы также можете изменить формат ячеек, например, отобразить суммы в денежном формате, выделив столбец C, щелкнув правой кнопкой мыши, выбрав "Формат ячеек" и выбрав "Денежный".</p>

<p>10. Билет</p> <p>1. Прикладные программные средства обработки текстовой информации. Текстовые процессоры. Текстовый процессор MS Word. Экранный интерфейс программы MS Word. Основы работы в MS Word. Документ, абзац, форматирование, редактирование. Форматирование текста (шрифт, размер, цвет и т. д.)</p> <ul style="list-style-type: none"> • Создание и редактирование таблиц • Вставка изображений и графиков • Проверка орфографии и грамматики • Создание оглавлений и сносок <p>Microsoft Word — это один из самых популярных текстовых процессоров, который входит в состав пакета Microsoft Office. Он предлагает множество возможностей для создания и редактирования документов и поддерживает различные форматы файлов.</p> <p><i>Экранный интерфейс программы MS Word</i></p> <p>Интерфейс MS Word состоит из нескольких основных элементов:</p> <ol style="list-style-type: none"> 1. Строка меню: В верхней части окна, где находятся основные команды и функции, такие как "Файл", "Главная", "Вставка", "Разметка страницы" и т. д. 2. Панель инструментов: Быстрый доступ к часто используемым функциям, таким как форматирование текста, вставка объектов и т. д. 3. Рабочая область: Основное пространство, где отображается документ, который вы редактируете. 4. Строка состояния: Внизу окна, где отображается информация о текущем документе, такая как номер страницы, количество слов и режим редактирования. 5. Линейки: Слева и сверху рабочей области, которые помогают устанавливать отступы и размеры. 	<p>10. Билет</p> <p>Текстовые процессоры</p> <p>Текстовые процессоры — это специализированные программы, которые позволяют создавать и редактировать текстовые документы. Они предлагают широкий набор функций, включая:</p> <p><i>Основы работы в MS Word</i></p> <ol style="list-style-type: none"> 1. Создание документа: <ul style="list-style-type: none"> ○ Откройте MS Word и выберите "Создать" для нового документа или "Открыть" для существующего. 2. Редактирование текста: <ul style="list-style-type: none"> ○ Вводите текст в рабочей области. Для редактирования выделите текст и используйте клавиши "Delete" или "Backspace" для удаления. 3. Форматирование текста: <ul style="list-style-type: none"> ○ Выделите текст, который хотите отформатировать, и используйте инструменты на панели инструментов (например, жирный шрифт, курсив, подчеркивание). ○ Измените шрифт, размер шрифта и цвет текста через вкладку "Главная". 4. Работа с абзацами: <ul style="list-style-type: none"> ○ Для изменения выравнивания абзаца (по левому, правому краю или по центру) используйте соответствующие кнопки на панели инструментов. ○ Установите отступы и межстрочный интервал через параметры форматирования абзаца. 5. Сохранение документа: <ul style="list-style-type: none"> ○ Нажмите "Файл" -> "Сохранить как" и выберите место для сохранения. 6. Печать документа: <ul style="list-style-type: none"> ○ Нажмите "Файл" -> "Печать" для настройки параметров печати и отправки документа на печать. <p>Документ, абзац, форматирование, редактирование</p> <ul style="list-style-type: none"> • Документ: Это файл, созданный в текстовом процессоре, содержащий текст, изображения и другие элементы. Документы могут быть сохранены в различных форматах, таких как .docx, .pdf и др. • Абзац: Это группа предложений, объединенных общей темой. В MS Word абзацы могут быть отформатированы с помощью отступов, выравнивания и межстрочного интервала. • Форматирование: Это процесс изменения внешнего вида текста и абзацев. Включает в себя выбор шрифта, размера, цвета, стиля, а также настройку отступов и интервалов. • Редактирование: Это процесс внесения изменений в текст, включая добавление, удаление или изменение содержимого документа. MS Word предлагает инструменты для поиска и замены текста, а также для проверки орфографии и грамматики.
---	---

2. Выполнить задание составить блок-схему алгоритма и написать программу согласно задания	
<p>11. Билет</p> <p>1. Прикладные программные средства обработки табличной информации. Электронные таблицы. Табличный процессор MS Excel. Экранный интерфейс программы MS Excel. Особенности работы в MS Excel. Абсолютная и относительная адресация.</p> <p>Microsoft Excel — один из самых популярных табличных процессоров, входящий в состав пакета Microsoft Office. Он предлагает мощные инструменты для работы с данными, включая формулы, функции, диаграммы и сводные таблицы.</p> <p>Интерфейс MS Excel состоит из следующих основных элементов:</p> <ol style="list-style-type: none"> 1. Строка меню: Содержит основные команды, такие как "Файл", "Главная", "Вставка", "Данные" и т. д. 2. Панель инструментов: Быстрый доступ к часто используемым функциям, таким как форматирование ячеек и вставка функций. 3. Рабочая область: Сетка, состоящая из ячеек, где вводятся и отображаются данные. 4. Строка формул: Позволяет вводить и редактировать содержимое текущей ячейки, включая формулы. 5. Лист: Рабочая область, состоящая из строк и столбцов, где размещаются данные. <p>Абсолютная и относительная адресация</p> <p>Относительная адресация: Ссылки на ячейки, которые изменяются при копировании формулы в другую ячейку. Например, если формула в ячейке A1 ссылается на B1, при копировании в A2 она будет ссылаться на B2.</p> <p>Абсолютная адресация: Ссылки, которые не изменяются при копировании формулы. Они обозначаются знаком доллара (\$). Например, если формула в A1 ссылается на B\$1, при копировании в A2 она все равно будет ссылаться на B\1.</p> <p>2. Выполнить задание на составление блок-схемы алгоритма и программы согласно задания</p>	<p>11. Билет</p> <p>Прикладные программные средства обработки табличной информации – это программы, которые используются для создания, редактирования, анализа и визуализации данных, организованных в табличной форме.</p> <p>Электронная таблица – компьютерный эквивалент обычной таблицы. Электронная таблица (ЭТ) позволяет хранить в табличной форме большое количество исходных данных, результатов, а также связей (алгебраических или логических соотношений) между ними.</p> <p>Табличный процессор – комплекс программ, предназначенных для создания и обработки электронных таблиц.</p> <p>Ячейка – минимальный объект табличного процессора;</p> <p>Строка – горизонтальный набор ячеек, заголовки столбцов – A, B, C,...,IV;</p> <p>столбец – вертикальный набор ячеек, заголовки строк – 1, 2, 3,...65536</p> <p>Адрес ячейки – определяется пересечением столбца и строки (A1, F123, AC72);</p> <p>Указатель ячейки – рамка;</p> <p>Активная ячейка – выделенная рамкой, с ней можно производить какие-либо операции;</p> <p>Смежные ячейки – ячейки расположенные последовательно;</p> <p>Диапазон (блок) ячеек – выделенные смежные ячейки, образующие прямоугольный участок таблицы;</p> <p>Адрес диапазона (блока) ячеек - определяется адресом верхней левой и нижней правой ячейки, разделенных двоеточием (:), B2:C7 → B2, B3, B4, B5, B6, B7, C2, C3, C4, C5, C6, C7.</p> <p>Книга – документ электронной таблицы, состоящий из листов, объединенных одним именем и являющихся файлом;</p> <p>Лист – рабочее поле, состоящее из ячеек.</p> <p>Относительная ссылка - это изменяющийся при копировании или перемещении формулы адрес ячейки, содержащей исходные данные (например: B5, G11).</p> <p>Абсолютная ссылка - это не изменяющийся при копировании и перемещении адрес ячейки, содержащей исходное значение. Для указания абсолютной адресации вводится символ \$. Различают два типа абсолютной ссылки - полная и частичная.</p>
12. Билет	12. Билет

<p>1. Представление графической и мультимедийной информации с помощью компьютерных презентаций в MS Power.Point. Подготовка компьютерных презентаций.</p> <p>Создание компьютерных презентаций в Microsoft PowerPoint — это эффективный способ представления графической и мультимедийной информации. Презентации могут использоваться для различных целей, включая обучение, деловые встречи и выставки. Вот основные шаги по подготовке компьютерных презентаций в PowerPoint:</p> <ol style="list-style-type: none"> 1. Подготовка к созданию презентации <ul style="list-style-type: none"> - Определите цель презентации: Понять, что вы хотите донести до вашей аудитории. - Исследуйте тему: Соберите информацию, данные и материалы, которые будете использовать. - Создайте план: Определите основные разделы и порядок их представления. 2. Создание презентации <ul style="list-style-type: none"> - Запустите PowerPoint: Откройте программу и выберите "Создать новую презентацию". - Выберите шаблон: PowerPoint предлагает различные шаблоны, которые могут помочь сделать вашу презентацию более привлекательной. - Добавьте слайды: Используйте кнопку "Новый слайд" для добавления слайдов. Вы можете выбрать различные макеты слайдов в зависимости от содержания. 3. Наполнение слайдов содержанием <ul style="list-style-type: none"> - Текст: Добавьте заголовки и основной текст. Используйте краткие и четкие формулировки. - Графика: <ul style="list-style-type: none"> - Изображения: Вставьте изображения, чтобы проиллюстрировать ваши идеи. Используйте функцию "Вставка" → "Рисунок". - Графики и диаграммы: Для визуализации данных используйте встроенные инструменты для создания графиков и диаграмм. - Мультимедиа: <ul style="list-style-type: none"> - Аудио: Вы можете добавить звуковые файлы, чтобы сделать презентацию более интерактивной. - Видео: Вставьте видеоролики, чтобы продемонстрировать информацию наглядно. Используйте функцию "Вставка" → "Видео". 	<ol style="list-style-type: none"> 4. Дизайн и оформление <ul style="list-style-type: none"> - Шрифты и цвета: Выберите читаемые шрифты и гармоничные цветовые схемы. Используйте не более двух-трех шрифтов для всей презентации. - Стили слайдов: Используйте стили и эффекты для улучшения визуального восприятия, но не переусердствуйте, чтобы не отвлекать внимание от содержания. - Анимации и переходы: Добавьте анимации для объектов на слайде и переходы между слайдами, чтобы сделать презентацию более динамичной. 5. Подготовка к показу <ul style="list-style-type: none"> - Репетиция: Проведите несколько репетиций, чтобы отработать свою речь и убедиться, что все работает должным образом. - Проверьте оборудование: Убедитесь, что проектор или экран, на котором будет демонстрироваться презентация, работают корректно. - Сохраните презентацию: Сохраните файл в нужном формате, чтобы избежать потери данных. Рекомендуется также сохранить резервную копию. 6. Проведение презентации <ul style="list-style-type: none"> - Взаимодействие с аудиторией: Поддерживайте контакт с аудиторией, задавайте вопросы и отвечайте на них. - Используйте заметки: Если необходимо, используйте заметки для вспомогательной информации, чтобы не отвлекаться от основного содержания. <p>Заключение</p> <p>Создание презентаций в Microsoft PowerPoint — это мощный инструмент для визуального представления информации. Следуя этим шагам, вы сможете подготовить эффективные и привлекательные презентации, которые помогут донести ваши идеи до аудитории.</p>
<p>2. Выполнить задание на составление блок-схемы алгоритма и программы согласно задания.</p>	

<p>13. Билет</p> <p>1. Базы данных. Классификация и модели баз данных. База данных: понятие. СУБД: понятие, виды, пример. Табличная форма представления баз данных.</p>	<p>13. Билет</p> <ul style="list-style-type: none"> База данных — это организованная совокупность данных, хранящихся и обрабатываемых с помощью компьютерных систем. Информация в базе данных структурирована для эффективного сохранения, обновления и извлечения. <p>Классификация баз данных</p> <ul style="list-style-type: none"> По способу хранения: реляционные, иерархические, сетевые, объектно-ориентированные, документо-ориентированные, колонно-ориентированные. По использованию: оперативные, аналитические. По масштабу: малые, крупные. Модели баз данных Реляционная модель: данные представлены в виде таблиц, состоящих из строк и столбцов. Иерархическая модель: данные организованы в виде иерархии с одним родительским элементом. Сетевая модель: данные связаны через несколько родителей и дочерних элементов. Объектно-ориентированная модель: данные хранятся в виде объектов. Документо-ориентированная модель: данные хранятся в виде документов. <p>Понятие СУБД</p> <ul style="list-style-type: none"> Система управления базами данных (СУБД) — это программное обеспечение для создания, управления и обслуживания баз данных. СУБД обеспечивает доступ, изменение, удаление и хранение данных в безопасной и организованной форме. <p>Виды СУБД</p> <ul style="list-style-type: none"> Реляционные СУБД (RDBMS): хранят данные в виде таблиц, управляют данными с использованием SQL. Документо-ориентированные СУБД: хранят данные в виде документов, например, JSON или BSON. Объектно-ориентированные СУБД: хранят данные в виде объектов. Графовые СУБД: используют графовую структуру для хранения и обработки данных. Колонно-ориентированные СУБД: хранят данные по колонкам для аналитических запросов. Ин-мемори СУБД: работают с данными, хранящимися в оперативной памяти.
<p>2. Выполнить задание на составление блок-схемы алгоритма и программы согласно задания.</p>	

<p>14. Билет</p> <p>1. Системы управления базами данных: основные характеристики, типы, характер использования. Основные понятия в MSAccess.</p>	<p>14. Билет</p> <p>Microsoft Access — это реляционная СУБД, входящая в пакет Microsoft Office, которая позволяет пользователям создавать и управлять базами данных. Основные понятия в MS Access</p> <p>База данных: Основная единица хранения данных, содержащая таблицы, запросы, формы и отчеты.</p> <p>Таблица: Основная структура для хранения данных в виде строк и столбцов. Каждая строка представляет собой запись, а каждый столбец — поле (атрибут).</p> <p>Запись: Строка в таблице, представляющая отдельный объект или элемент данных (например, информация о клиенте).</p> <p>Поле: Столбец в таблице, представляющий конкретный атрибут записи (например, имя, адрес, телефон).</p> <p>Запрос: Инструмент для извлечения и манипулирования данными из одной или нескольких таблиц с помощью языка SQL. Запросы могут быть простыми (выборка) или сложными (с объединением, фильтрацией и агрегацией).</p> <p>Форма: Графический интерфейс для ввода и отображения данных из таблиц. Формы упрощают взаимодействие с данными для пользователей.</p> <p>Отчет: Форматированный вывод данных, который можно распечатать или просмотреть на экране. Отчеты используются для представления информации в удобочитаемом виде.</p> <p>Связи: Определяют, как таблицы связаны друг с другом. Например, связь "один ко многим" между таблицами "Клиенты" и "Заказы".</p>
<p>2. Выполнить задание на составление блок-схемы алгоритма и программы согласно задания.</p>	
<p>15. Билет</p> <p>1. Элементы реляционной модели данных. Создание структуры реляционной базы данных. Привести примеры реляционной базы данных. Выполнить задание на составление блок-схемы алгоритма и программы согласно задания.</p> 	<p>15. Билет</p> <pre>-- Создание таблицы "Книги" CREATE TABLE Книги (ID_Книги INT PRIMARY KEY AUTO_INCREMENT, Название VARCHAR(255), Автор VARCHAR(255), Год_издания INT, Жанр VARCHAR(100)); -- Создание таблицы "Читатели" CREATE TABLE Читатели (ID_Читателя INT PRIMARY KEY AUTO_INCREMENT, Имя VARCHAR(255),</pre>

	Фамилия VARCHAR(255), Дата_регистрации DATE); -- Создание таблицы "Заказы" CREATE TABLE Заказы (ID_Заказа INT PRIMARY KEY AUTO_INCREMENT, ID_Читателя INT, ID_Книги INT, Дата_заказа DATE, Дата_возврата DATE, FOREIGN KEY (ID_Читателя) REFERENCES Читатели(ID_Читателя), FOREIGN KEY (ID_Книги) REFERENCES Книги(ID_Книги));
2. Выполнить задание на составление блок-схемы алгоритма и программы.	
16. Билет	16. Билет
1. Инструментальные программные средства для решения прикладных математических задач. Построение графиков.	Инструментальные программные средства для решения прикладных математических задач В современном мире существует множество программных средств, которые помогают решать прикладные математические задачи, включая построение графиков. Эти инструменты могут быть как специализированными математическими программами, так и более общими программами для анализа данных и визуализации. Вот несколько популярных инструментов: 1. MATLAB Описание: MATLAB — это высокоуровневый язык программирования и среда для численных вычислений, визуализации и программирования.
2. Выполнить задание на составление блок-схемы алгоритма и программы.	2. Составление блок-схемы алгоритма и программы: - Создание схемы и кода для решения математических задач.
17. Билет	17. Билет
1. Прикладные программы для математических вычислений. Работа с матрицами и векторами.	
2. Выполнить задание по созданию базы данных в MS Access	2. Создание базы данных в MS Access: - Создание и редактирование базы данных с использованием MS Access.
18. Билет	18. Билет
1. Локальные вычислительные сети, их назначение, виды. Топологии сетей, их преимущества и недостатки. Локальные вычислительные сети (ЛВС) — это сети, которые обеспечивают связь между компьютерами и другими устройствами в ограниченной географической области, такой как дом, офис или кампус.	- Шинная топология: Все устройства подключены к одному кабелю. Преимущества: простота установки и низкие затраты. Недостатки: если кабель поврежден, вся сеть выходит из строя. - Звездобразная топология: Все устройства подключены к центральному коммутатору или хабу. Преимущества: легкость в управлении и устранении неисправностей. Недостатки: если центральное устройство выходит из строя, вся сеть перестает работать.

<p>Они позволяют пользователям обмениваться данными и ресурсами, такими как принтеры и файлы.</p> <p>Назначение ЛВС</p> <ul style="list-style-type: none"> - Обмен данными: ЛВС позволяет пользователям обмениваться файлами и данными между устройствами. - Совместное использование ресурсов: Устройства, такие как принтеры и сканеры, могут быть использованы несколькими пользователями. - Коммуникация: ЛВС поддерживает различные формы коммуникации, включая электронную почту и мессенджеры. - Централизованное управление: ЛВС позволяет администраторам управлять пользователями и ресурсами из одного места. <p>Виды ЛВС</p> <ul style="list-style-type: none"> - Ethernet: Наиболее распространенный тип ЛВС, использующий кабели для передачи данных. - Wi-Fi: Беспроводная сеть, позволяющая подключать устройства без проводов. - Token Ring: Использует токен для управления доступом к сети, менее распространен в современных сетях. - FDDI (Fiber Distributed Data Interface): Использует оптоволоконные кабели для передачи данных на высоких скоростях. <p>Топологии ЛВС</p>	<ul style="list-style-type: none"> - Кольцевая топология: Устройства соединены в кольцо, и данные передаются по кругу. Преимущества: предсказуемая производительность. Недостатки: если одно устройство выходит из строя, это может повлиять на всю сеть. - Смешанная топология: Комбинация различных топологий для достижения оптимальной производительности и надежности. <p>Преимущества ЛВС</p> <ul style="list-style-type: none"> - Высокая скорость передачи данных: ЛВС обеспечивает высокую скорость обмена данными между устройствами. - Экономия затрат: Совместное использование ресурсов снижает затраты на оборудование. - Удобство: Легкость в подключении и управлении устройствами. - Безопасность: Возможность настройки прав доступа и контроля за использованием ресурсов. <p>Недостатки ЛВС</p> <ul style="list-style-type: none"> - Сложность настройки: Требуется знаний для правильной настройки и управления. - Зависимость от центральных устройств: Поломка центрального устройства может привести к сбоям в работе всей сети. - Ограниченная география: ЛВС ограничены в своем радиусе действия, что может потребовать использования дополнительных технологий для расширения сети. <p>Локальные вычислительные сети играют важную роль в современных организациях, обеспечивая эффективное взаимодействие и обмен данными между пользователями и устройствами..</p>
<p>2. Выполнить задание по созданию базы данных в MS Access</p>	<p>2. Создание базы данных в MS Access:</p> <ul style="list-style-type: none"> - Создание и редактирование базы данных, связанной с сетями.
<p>19. Билет</p>	<p>19. Билет</p>
<p>1. Глобальная сеть: основные понятия. Интернет. Сервисы интернета.</p> <p>Глобальная сеть – это большая сеть, которая объединяет миллионы компьютеров и устройств по всему миру. Она позволяет пользователям обмениваться данными и ресурсами, независимо от физического местоположения.</p> <p>Интернет - это всемирная компьютерная сеть, объединяющая в единое целое десятки тысяч разнородных локальных и глобальных компьютерных сетей, связанных определенными соглашениями (протоколами).</p> <p>IP-адреса — это идентификатор, который позволяет передавать информацию между устройствами в сети: они содержат информацию о местоположении и делают устройства доступными для связи. Протокол — это набор правил, по которым передаются данные. С помощью протоколов связываются между собой компьютеры в сети.</p>	<p>СЕРВИСЫ ИНТЕРНЕТА:</p> <p>Электронная почта (англ. e-mail от англ. electronic mail) — технология и служба по пересылке и получению электронных сообщений между пользователями компьютерной сети (в том числе — Интернета)</p> <p>Сервис FTP — система файловых архивов, обеспечивающая хранение и пересылку файлов различных типов</p> <p>World Wide Web (WWW, W3) — гипертекстовая (гипермедиа) система, предназначенная для интеграции различных сетевых ресурсов в единое информационное пространство;</p> <p>Сервис IRC, предназначенный для поддержки текстового общения в реальном времени (chat)</p> <p>Виртуальная частная сеть (VPN) - это расширение сети вашей организации за счет уже существующей общей сети, например Internet. VPN позволяет управлять сетевым потоком данных и предоставляет такие важные функции, как идентификация и защита данных.</p>

	DNS — это система, которая связывает между собой доменное имя сайта, то есть его название, и IP-адрес (Благодаря DNS-серверу вам не нужно знать IP-адрес сайта, чтобы попасть на него.)																																				
2. Выполнить задание на системы счисления.	2. Задание на системы счисления: - Решение задач, связанных с переводами между системами счисления.																																				
20. Билет	20 Билет																																				
1. Понятие об алгоритме, свойства, способы представления алгоритмов, свойства алгоритма. Алгоритм — это упорядоченная последовательность действий, выполнение которых приводит к достижению определённой цели. Он должен быть чётко определён и понятен для исполнителя. Важно, чтобы алгоритм завершался за конечное время, приводя к результату	Свойства алгоритма 1. Дискретность : алгоритм разбит на отдельные, чётко определённые шаги. 2. Определённость : каждое действие алгоритма должно быть однозначным. 3. Результативность : алгоритм всегда приводит к результату за конечное число шагов. 4. Массовость : алгоритм может быть применён для решения множества однотипных задач. 5. Точность : все действия алгоритма точно определены и не допускают двусмысленности Способы представления алгоритмов 1. Словесное описание: последовательность шагов записана текстом. 2. Графическая форма: используется блок-схема, где каждый блок представляет отдельное действие. 3. Программный код : алгоритм записан на языке программирования. 4. Табличная форма : данные и действия представлены в виде таблицы																																				
2. Построить таблицу истинности для логического выражения:																																					
Таблицы истинности 1. AND (И) <table><tr><td>A</td><td>B</td><td>A AND B</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	A AND B	0	0	0	0	1	0	1	0	0	1	1	1	3. NOT (НЕ) <table><tr><td>A</td><td>NOT A</td></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table> 2. OR (ИЛИ) <table><tr><td>A</td><td>B</td><td>A OR B</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	NOT A	0	1	1	0	A	B	A OR B	0	0	0	0	1	1	1	0	1	1	1	1
A	B	A AND B																																			
0	0	0																																			
0	1	0																																			
1	0	0																																			
1	1	1																																			
A	NOT A																																				
0	1																																				
1	0																																				
A	B	A OR B																																			
0	0	0																																			
0	1	1																																			
1	0	1																																			
1	1	1																																			
1. Виды алгоритмов. Основные алгоритмические конструкции. Алгоритмы — это последовательности шагов, предназначенные для решения определенных задач. Они могут быть классифицированы по различным критериям, включая тип решаемой задачи, структуру и	Последовательность: Это базовая конструкция, в которой операции выполняются одна за другой. Каждое действие последовательно следует за предыдущим. A																																				

<p>подход к решению. Ниже приведены основные виды алгоритмов и основные алгоритмические конструкции.</p> <p>Виды алгоритмов</p> <p>По типу решаемых задач:</p> <p>Алгоритмы сортировки: Например, пузырьковая сортировка, быстрая сортировка, сортировка слиянием.</p> <p>Алгоритмы поиска: Линейный поиск, бинарный поиск.</p> <p>Алгоритмы оптимизации: Например, жадные алгоритмы, динамическое программирование.</p> <p>Алгоритмы обработки данных: Работа с графами (например, алгоритм Дейкстры, алгоритм Флойда-Уоршелла).</p> <p>Алгоритмы шифрования: Например, AES, RSA.</p> <p>По структуре:</p> <p>Линейные алгоритмы: Выполняются последовательно, шаг за шагом.</p> <p>Разветвляющиеся алгоритмы: Используют условные конструкции для выполнения различных действий в зависимости от условий (например, if-else).</p> <p>Циклические алгоритмы: Выполняют одни и те же шаги несколько раз (например, while, for).</p> <p>Основные алгоритмические конструкции</p>	<p>В</p> <p>С</p> <p>Выбор (разветвление):</p> <p>Используется для выполнения различных действий в зависимости от условий. Основные конструкции: if, else if, else, switch.</p> <pre> if (условие) { // действия, если условие истинно } else { // действия, если условие ложно } </pre> <p>Цикл (повторение):</p> <p>Позволяет повторять выполнение определенного блока кода несколько раз. Основные конструкции: for, while, do while.</p> <pre> while (условие) { // действия, выполняемые пока условие истинно } </pre> <p>Функции и процедуры:</p> <p>Позволяют группировать код для повторного использования. Функции могут возвращать значения, а процедуры — выполнять действия без возврата.</p> <pre> function имяФункции(параметры) { // тело функции return значение; // если функция возвращает значение } </pre> <p>Рекурсия:</p> <p>Это метод, при котором функция вызывает саму себя для решения подзадачи. Рекурсия должна иметь базовый случай для завершения.</p> <pre> function рекурсивнаяФункция(параметры) { if (условие) { return базовый случай; } else { return рекурсивнаяФункция(новые параметры); } } </pre>
<p>2. Выполнить задание на создание интерактивной презентации.</p>	
<p>22. Билет</p>	<p>22 Билет</p>
<p>1. Способы записи алгоритмов. Блок-схемы. Основные элементы блок-схем.</p> <p>Текстовая запись:</p>	<p>1. Способы записи алгоритмов. Блок-схемы. Основные элементы блок-схем.</p> <p>- Опишите способы записи алгоритмов, включая текстовые и графические способы. Приведите примеры блок-схем и их элементы: прямоугольники, ромбы и т.д.</p>

Алгоритмы могут быть записаны в виде последовательности шагов на естественном языке или в псевдокоде. Псевдокод — это упрощенный, неформальный язык, который использует конструкции, схожие с программированием, но не привязан к конкретному языку.

Пример псевдокода:

Начало

Ввод A, B

Если A > B тогда

 Вывести A

Иначе

 Вывести B

Конец

Блок-схемы:

Способы записи алгоритмов

Визуальное представление алгоритма, состоящее из различных геометрических фигур, каждая из которых обозначает определенный тип операции или действия. Блок-схемы помогают лучше понять структуру алгоритма и его логику.

Блок-схемы

Блок-схема — это графическое представление алгоритма, состоящее из различных блоков, соединенных стрелками, которые показывают порядок выполнения шагов.

Основные элементы блок-схем

Начало/Конец (Овал):

Используется для обозначения начала и конца алгоритма.

Процесс (Прямоугольник):

Обозначает выполнение операции или действия, такого как вычисление, присваивание значения и т.д.

Условие (Ромб):

Используется для обозначения условия, которое требует принятия решения. Имеет два выхода: "да" (истина) и "нет" (ложь).

Ввод/Вывод (Параллелограмм):

Обозначает операции ввода или вывода данных, такие как чтение с клавиатуры или вывод на экран.

Соединительные элементы (Стрелки):

Показывают направление потока выполнения алгоритма от одного блока к другому.

Пример блок-схемы

Вот пример блок-схемы для алгоритма, который сравнивает два числа и выводит большее из них:

2. Дано логическое выражение, зависящее от 5 логических переменных: Сколько существует различных наборов значений переменных, при которых выражение ложно? Решить задачу в Excel.

32


```
print("a b c d")
for a in range(2):
    for b in range(2):
        for c in range(2):
            for d in range(2):
                if (a and (b or not(c) or d) and (not(b and not(d)))) == True:
                    print(a,b,c,d)
```

- Используйте Excel для создания таблицы с 5 переменными и определите количество наборов, при которых выражение является ложным. 32

A	B	A	D	E
0	0	0	0	0
0	0	0	0	1
0	0	0	1	0
0	0	0	1	1
0	0	1	0	0
0	0	1	0	1
0	0	1	1	0
0	0	1	1	1
0	1	0	0	0

23. Билет	23 Билет																														
1. Арифметические и логические операции на языке программирования высокого уровня.	1. Арифметические и логические операции на языке программирования высокого уровня. - Приведите примеры арифметических и логических операций на языке программирования (например, Python, C++) с пояснением их работы. <pre>print("a b c d") for a in range(2): for b in range(2): for c in range(2): for d in range(2): if (a and (b or not(c) or d) and (not(b and not(d))))==True: print(a,b,c,d)</pre>																														
2. Решить логическую задачу табличным способом. Для решения логической задачи табличным способом, мы можем использовать метод составления таблицы истинности. Давайте рассмотрим пример логической задачи и решим её с помощью таблицы истинности. Пример задачи Предположим, у нас есть два утверждения: <ul style="list-style-type: none">АА: "Сегодня идет дождь."ВВ: "Я возьму зонт."	Мы хотим выяснить, в каких случаях я возьму зонт, основываясь на следующих логических условиях: <ol style="list-style-type: none">Если идет дождь (АА), то я возьму зонт (ВВ).Если не идет дождь ($\neg A \neg A$), то я не возьму зонт ($\neg B \neg B$). Шаг 1: Определяем переменные <ul style="list-style-type: none">АА: "Сегодня идет дождь" (истинно или ложно)ВВ: "Я возьму зонт" (истинно или ложно) Шаг 2: Составляем таблицу истинности Теперь мы можем составить таблицу истинности для двух переменных АА и ВВ: <table><tr><th>АА</th><th>ВВ</th><th>$A \rightarrow B$</th><th>$\neg A \neg A$</th><th>$\neg B \neg B$</th><th>$\neg A \rightarrow \neg B$</th></tr><tr><td>T</td><td>T</td><td>T</td><td>F</td><td>F</td><td>T</td></tr><tr><td>T</td><td>F</td><td>F</td><td>F</td><td>T</td><td>T</td></tr><tr><td>F</td><td>T</td><td>T</td><td>T</td><td>F</td><td>F</td></tr><tr><td>F</td><td>F</td><td>T</td><td>T</td><td>T</td><td>T</td></tr></table> Шаг 3: Анализируем результаты Теперь давайте проанализируем каждую строку таблицы: <ol style="list-style-type: none">Первая строка: Если идет дождь (Т) и я беру зонт (Т), то оба условия выполняются ($A \rightarrow B = T$).Вторая строка: Если идет дождь (Т), но я не беру зонт (F), то условие не выполняется ($A \rightarrow B = F$).Третья строка: Если не идет дождь (F), но я беру зонт (Т), то условие не выполняется для $\neg A \rightarrow \neg B$ (это F).	АА	ВВ	$A \rightarrow B$	$\neg A \neg A$	$\neg B \neg B$	$\neg A \rightarrow \neg B$	T	T	T	F	F	T	T	F	F	F	T	T	F	T	T	T	F	F	F	F	T	T	T	T
АА	ВВ	$A \rightarrow B$	$\neg A \neg A$	$\neg B \neg B$	$\neg A \rightarrow \neg B$																										
T	T	T	F	F	T																										
T	F	F	F	T	T																										
F	T	T	T	F	F																										
F	F	T	T	T	T																										

	<p>4. Четвертая строка: Если не идет дождь (F) и я не беру зонт (F), то оба условия выполняются ($A \rightarrow B = T$ и $\neg A \rightarrow \neg B \neg A \rightarrow \neg B = T$).</p> <p>Из таблицы истинности видно, что я возьму зонт только в случае, если идет дождь. Если дождя нет, то я не возьму зонт, что подтверждает условия задачи. Таким образом, логическая задача решена с помощью табличного метода.</p>
24. Билет	24 Билет
<p>1. Основные элементы программы и алфавит языка программирования высокого уровня. Арифметические и логические операции на языке программирования высокого уровня</p> <p>1. Основные элементы программы</p> <p>Программа на языке высокого уровня состоит из различных элементов, которые выполняют определенные функции. Основные элементы программы включают:</p> <p>Переменные:</p> <p>Переменные используются для хранения данных. Они имеют имя и тип, который определяет, какие значения могут быть в них сохранены. Пример: int a;, float pi;, string name;</p> <p>Константы:</p> <p>Константы — это фиксированные значения, которые не изменяются в процессе выполнения программы. Пример: const int MAX_SIZE = 100;</p> <p>Операторы:</p> <p>Операторы выполняют операции над переменными и значениями. Они могут быть арифметическими, логическими, сравнительными и другими. Пример: +, -, *, /, &&, </p> <p>Условия:</p> <p>Условные конструкции позволяют выполнять разные действия в зависимости от выполнения определенного условия. Пример: if, else, switch</p> <p>Циклы:</p> <p>Циклы позволяют повторять блоки кода несколько раз. Пример: for, while, do...while</p> <p>Функции:</p> <p>Функции — это блоки кода, которые выполняют определенную задачу и могут быть вызваны из других частей программы. Пример: int add(int a, int b) { return a + b; }</p> <p>Классы и объекты (в языках, поддерживающих ООП):</p> <p>Классы определяют шаблоны для создания объектов, которые могут содержать данные и методы.</p>	<p>Пример:</p> <pre>class Car { public: string color; void drive() { /* код для движения */ } };</pre> <p>Алфавит языка программирования высокого уровня</p> <p>Алфавит языка программирования высокого уровня включает в себя:</p> <p>Идентификаторы:</p> <p>Имена переменных, функций, классов и других элементов программы. Они должны начинаться с буквы или символа подчеркивания и могут содержать буквы, цифры и символы подчеркивания. Пример: myVariable, _count, Sum</p> <p>Ключевые слова:</p> <p>Зарезервированные слова, имеющие специальное значение в языке программирования. Они не могут использоваться в качестве идентификаторов. Пример: if, else, while, return, class</p> <p>Литералы:</p> <p>Конкретные значения, которые могут быть использованы в программе, такие как числа, строки и логические значения. Пример: 42, "Hello, World!", true</p> <p>Операторы:</p> <p>Символы, используемые для выполнения операций. Они могут быть арифметическими, логическими, сравнительными и другими.</p> <p>Символы:</p> <p>Другие символы, такие как скобки, запятые, точки с запятой, которые используются для структурирования кода. Пример: (), {}, ;, ,</p> <p>Арифметические и логические операции на языке программирования высокого уровня</p> <p>Арифметические операции</p> <p>Арифметические операции выполняются с числовыми значениями и включают:</p> <p>Сложение (+):</p> <p>Пример: int sum = a + b;</p>

	<p>Вычитание (-): Пример: <code>int difference = a - b;</code></p> <p>Умножение (*): Пример: <code>int product = a * b;</code></p> <p>Деление (/): Пример: <code>float quotient = a / b;</code></p> <p>Остаток от деления (%): Пример: <code>int remainder = a % b;</code></p> <p>Логические операции</p> <p>Логические операции используются для работы с булевыми значениями (истина или ложь) и включают:</p> <p>Логическое И (&&): Возвращает истину, если оба операнда истинны. Пример: <code>if (a > 0 && b > 0) { /* код */ }</code></p> <p>Логическое ИЛИ (): Возвращает истину, если хотя бы один из операндов истинен. Пример: <code>if (a > 0 b > 0) { /* код */ }</code></p> <p>Логическое НЕ (!): Инвертирует логическое значение. Пример: <code>if (!isTrue) { /* код */ }</code></p>
2. По заданной логической функции построить логическую схему	<p>2. По заданной логической функции построить логическую схему. - Используйте логические элементы (AND, OR, NOT) для построения схемы.</p> 
25. Билет	25 Билет
<p>1. Алгоритмические конструкции линейного алгоритма, их реализация на языке программирования высокого уровня.</p> <p>Алгоритмические конструкции линейного алгоритма представляют собой последовательность действий, которые выполняются последовательно, без разветвлений и циклов. В линейном алгоритме каждое действие выполняется одно за другим. Рассмотрим основные алгоритмические конструкции и их реализацию на языке программирования высокого уровня, например, на Python.</p> <p>Основные алгоритмические конструкции линейного алгоритма</p> <p>Ввод данных: Получение данных от пользователя или из внешних источников.</p> <p>Обработка данных: Выполнение арифметических операций, преобразование данных и другие манипуляции.</p> <p>Вывод данных:</p>	<p>Объяснение кода</p> <p>Ввод данных: Используем функцию <code>input()</code> для получения чисел от пользователя. Функция <code>float()</code> преобразует введенные строки в числа с плавающей точкой.</p> <p>Обработка данных: Выполняем арифметическую операцию сложения и сохраняем результат в переменной <code>сумма</code>.</p> <p>Вывод данных: Используем функцию <code>print()</code> для отображения результата.</p> <p>Другие примеры линейных алгоритмов</p> <p>Пример 2: Вычисление площади прямоугольника</p> <p># 1. Ввод данных</p> <p><code>длина = float(input("Введите длину прямоугольника: "))</code> # Ввод длины</p> <p><code>ширина = float(input("Введите ширину прямоугольника: "))</code> # Ввод ширины</p> <p># 2. Обработка данных</p>

Отображение результатов пользователю.

Пример реализации линейного алгоритма на Python

Рассмотрим пример линейного алгоритма, который запрашивает у пользователя два числа, вычисляет их сумму и выводит результат.

1. Ввод данных

```
a = float(input("Введите первое число: ")) # Ввод первого числа
```

```
b = float(input("Введите второе число: ")) # Ввод второго числа
```

2. Обработка данных

```
сумма = a + b # Вычисление суммы
```

3. Вывод данных

```
print("Сумма чисел:", сумма) # Вывод результата
```

```
площадь = длина * ширина # Вычисление площади
```

3. Вывод данных

```
print("Площадь прямоугольника:", площадь) # Вывод результата
```

Пример 3: Конвертация температуры

1. Ввод данных

```
цельсий = float(input("Введите температуру в градусах Цельсия: ")) # Ввод температуры
```

2. Обработка данных

```
фаренгейт = (цельсий * 9/5) + 32 # Конвертация в Фаренгейт
```

3. Вывод данных

```
print("Температура в градусах Фаренгейта:", фаренгейт) # Вывод результата
```

2. Выполнить задание на перевод десятичного числа в прямой, обратный, дополнительный код.

Перевод десятичного числа в различные коды — это важная тема в информатике и компьютерной архитектуре. Мы рассмотрим, как преобразовать десятичное число в прямой, обратный и дополнительный код. Для примера возьмем десятичное число -5_{10} и рассмотрим его представление в 8-битной системе.

1. Прямой код

Прямой код — это представление числа в двоичном формате, где старший бит (бит знака) указывает на знак числа: 0 для положительных чисел и 1 для отрицательных.

Пример: Прямой код для -5

1. Положительное представление:

- Десятичное число 5_{10} в двоичном формате:

$$5_{10} = 00000101_2$$

2. Отрицательное представление:

- Для -5_{10} старший бит будет 1, поэтому прямой код:

$$-5_{10} = 10000101_2$$

2. Обратный код

Обратный код получается путем инверсии всех битов прямого кода. То есть, 0 становится 1, а 1 становится 0.

Пример: Обратный код для -5

1. Прямой код:

$$-5_{10} = 10000101_2$$

2. Инверсия битов:

Дополнительный код: 01111011_2

Пример: Прямой код для -5

С 0 контекстными подсказками

1. Положительное представление:

- Десятичное число 5 в двоичном формате:

$$5_{10} = 00000101_2$$

2. Отрицательное представление:

- Для -5_{10} старший бит будет 1, поэтому прямой код:

$$-5_{10} = 10000101_2$$

2. Обратный код

Обратный код получается путем инверсии всех битов прямого кода. То есть, 0 становится 1, а 1 становится 0.

С 0 контекстными подсказками

Пример: Обратный код для -5

1. Прямой код:

$$-5_{10} = 10000101_2$$

2. Инверсия битов:

$$01111010_2$$

Таким образом, обратный код для -5 будет:

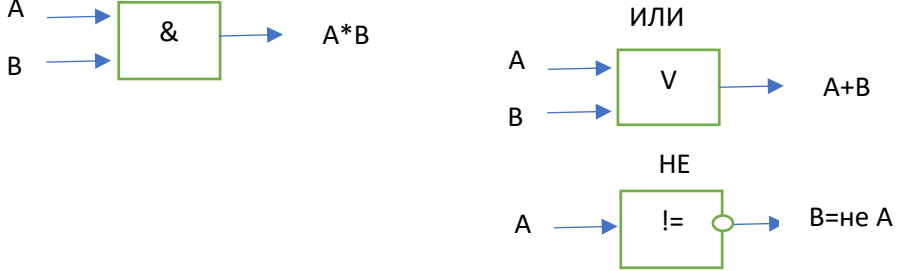
$$-5_{10} = 01111010_2$$

3. Дополнительный код

Дополнительный код получается путем добавления 1 к обратному коду. Это наиболее распространенный способ представления отрицательных чисел в компьютерах.

<ul style="list-style-type: none"> ◦ 01111010_2 <p>Таким образом, обратный код для -5 будет:</p> <ul style="list-style-type: none"> • $-5_{10} = 01111010_2$ <p>3. Дополнительный код</p> <p>Дополнительный код получается путем добавления 1 к обратному коду. Это наиболее распространенный способ представления отрицательных чисел в компьютерах.</p> <p>Пример: Дополнительный код для -5</p> <ol style="list-style-type: none"> 1. Обратный код: <ul style="list-style-type: none"> ◦ 01111010_2 2. Добавление 1: <ul style="list-style-type: none"> ◦ $01111010_2 + 00000001_2 = 01111011_2$ <p>Таким образом, дополнительный код для -5 будет:</p> <ul style="list-style-type: none"> • $-5_{10} = 01111011_2$ <p>Итоговое представление</p> <ul style="list-style-type: none"> • Прямой код: 10000101_2 • Обратный код: 01111010_2 	<p>Пример: Дополнительный код для -5 С 0 ко</p> <ol style="list-style-type: none"> 1. Обратный код: <ul style="list-style-type: none"> ◦ 01111010_2 2. Добавление 1: <ul style="list-style-type: none"> ◦ $01111010_2 + 00000001_2 = 01111011_2$ <p>Таким образом, дополнительный код для -5 будет:</p> <ul style="list-style-type: none"> • $-5_{10} = 01111011_2$ <p>Итоговое представление</p> <ul style="list-style-type: none"> • Прямой код: 10000101_2 • Обратный код: 01111010_2 • Дополнительный код: 01111011_2
<p>26. Билет</p> <p>1. Алгоритмические конструкции ветвления, их реализация на языке программирования высокого уровня.</p> <p>2. Выполнить задание на создание и редактирование табличного документа в MS Excel</p>	<p>26 Билет</p> <p>1. Алгоритмические конструкции ветвления, их реализация на языке программирования высокого уровня. - Объясните конструкции ветвления (if, switch) и приведите пример кода на языке программирования.</p> <p>2. Выполнить задание на создание и редактирование табличного документа в MS Excel. - Создайте таблицу с данными и выполните формулы для анализа.</p>
<p>27. Билет</p> <p>1. Алгоритмические конструкции цикла, их реализация на языке программирования высокого уровня.</p> <p>1. Алгоритмические конструкции цикла, их реализация на языке программирования высокого уровня.</p> <p>Алгоритмические конструкции цикла:</p> <p>Цикл с параметром (for):</p> <pre># Цикл с фиксированным числом повторений for i in range(5): # от 0 до 4 print(i)</pre> <p># Цикл по элементам коллекции</p>	<p>27 Билет</p> <p>Цикл с предусловием (while):</p> <pre># Выполняется пока условие истинно count = 0 while count < 5: print(count) count += 1</pre> <p>Цикл с постусловием (в Python напрямую нет, эмулируется через while):</p> <pre>x=5 while (x!=0): print("Действие")</pre>

<pre>colors = ["red", "green", "blue"] for color in colors: print(color)</pre>	<pre>x-=1</pre> <p>Управление циклом: break - прерывает цикл continue - пропускает текущую итерацию</p>
<p>2. Два числа -34_{10} и 42_{10} перевести в двоичную систему счисления. Сложить два числа в однобайтовом формате в обратном коде. Сделать проверку</p> <p>2. Два числа -34_{10} и 42_{10} перевести в двоичную систему счисления. Сложить два числа в однобайтовом формате в обратном коде. Сделать проверку.</p> <p>Перевод -34_{10} в двоичную систему:</p> <ul style="list-style-type: none"> • 34 делим на 2 пока не дойдем до 0: • $34 \div 2 = 17$ (остаток 0) • $17 \div 2 = 8$ (остаток 1) • $8 \div 2 = 4$ (остаток 0) • $4 \div 2 = 2$ (остаток 0) • $2 \div 2 = 1$ (остаток 0) • $1 \div 2 = 0$ (остаток 1) • $34_{10} = 100010_2$ <ul style="list-style-type: none"> • В 8-битном формате: 00100010 <p>Для отрицательного числа инвертируем: 11011101</p> <p>2. Перевод 42_{10} в двоичную систему:</p> <ul style="list-style-type: none"> • 42 делим на 2: • $42 \div 2 = 21$ (остаток 0) • $21 \div 2 = 10$ (остаток 1) • $10 \div 2 = 5$ (остаток 0) • $5 \div 2 = 2$ (остаток 1) • $2 \div 2 = 1$ (остаток 0) • $1 \div 2 = 0$ (остаток 1) • $42_{10} = 101010_2$ • 42_{10} в 8-битном формате: 00101010 <p>3. Сложение в обратном коде: 11011101 (-34) 00101010 (42)</p>	<pre>00101010 (42) + 11011110 (-34) ----- 11110100</pre> <p>4. Проверка:</p> <ul style="list-style-type: none"> • Результат 11110100 в обратном коде соответствует • $-34 + 42 = 8$, что не соответствует полученному результату из-за переполнения однобайтового формата • Для корректной работы с такими числами нужен больший формат <p>В данном случае мы видим ограничение однобайтового формата, который не может корректно представить результат сложения этих чисел.</p>
28. Билет	28 Билет
<p>1. Массивы. Виды массивов.</p> <pre>a=[1,2,3,4,5] print(a) for i in range(5): c=int(input()) a.append(c)</pre>	<pre>a=[1,2,3,4,5] print(a) b=[] for i in range(5): c=int(input()) b.append(c)</pre>

<pre>print(a) for i in range(5): c=int(input()) a.insert(c,i) print(a) a.clear() print(a) b=[[1,2,3,4,5],[1,2,3,4,5]] print(b[1][3])</pre>	<pre>print(b)</pre>
<p>2. Выполнить задание на создание логической схемы.</p>	 <p>The diagrams show three basic logic gates: an AND gate (labeled '&') with inputs A and B and output A*B; an OR gate (labeled 'ИЛИ' and 'V') with inputs A and B and output A+B; and a NOT gate (labeled 'НЕ' and '!=') with input A and output V=не A.</p>
<p>29. Билет</p>	<p>29 Билет</p>
<p>1. Понятие объектно-ориентированного программирования и базовые определения.</p> <p>Объектно-ориентированное программирование (ООП) — это парадигма программирования, основанная на концепции "объектов", которые могут содержать как данные, так и код: данные в виде полей (или атрибутов), и код в виде процедур (или методов). ООП помогает организовать код, делает его более понятным и удобным для сопровождения.</p> <h3>Основные понятия и определения ООП</h3> <ol style="list-style-type: none"> Объект: Объект — это экземпляр класса, представляющий собой конкретную реализацию абстракции. Объекты могут взаимодействовать друг с другом и с внешним миром через методы. Например, объект "Автомобиль" может иметь атрибуты (цвет, модель, скорость) и методы (ускориться, остановиться). Класс: Класс — это шаблон или чертеж для создания объектов. Он определяет атрибуты и методы, которые будут у объектов этого класса. Например, класс "Автомобиль" может определять, что у каждого автомобиля есть цвет и модель, а также методы для управления движением. 	<ol style="list-style-type: none"> Атрибуты (или свойства): Атрибуты — это переменные, которые хранят состояние объекта. Они определяют характеристики объекта. Например, атрибутами класса "Автомобиль" могут быть цвет, модель и скорость. Методы: Методы — это функции, определенные в классе, которые описывают поведение объектов. Методы могут изменять состояние объекта или выполнять действия. Например, метод ускориться может увеличивать скорость автомобиля. Инкапсуляция: Инкапсуляция — это принцип, который позволяет скрыть внутреннюю реализацию объекта от внешнего мира и предоставлять доступ к данным только через методы. Это помогает защищать данные и уменьшает зависимость между объектами. Наследование: Наследование — это механизм, позволяющий создавать новый класс на основе существующего. Новый класс (наследник) наследует атрибуты и методы родительского класса, что способствует повторному использованию кода. Например, класс "Спортивный автомобиль" может наследовать от класса "Автомобиль" и добавлять свои специфические атрибуты, такие как максимальная скорость.

	<p>7. Полиморфизм: Полиморфизм — это возможность использовать объекты разных классов через один и тот же интерфейс. Это позволяет писать более универсальный код. Например, если у вас есть метод управлять, который работает с объектами классов "Автомобиль" и "Велосипед", вы можете использовать его для управления обоими типами объектов, не заботясь о различиях в их реализации.</p>																																				
<p>2. Составить таблицу истинности для логического выражения</p> <p>Логические элементы компьютера — это базовые компоненты, которые выполняют логические операции над двоичными значениями (0 и 1). Основные логические операции включают:</p> <p>1. AND (И): Возвращает 1, если оба входа равны 1.</p> <p>2. OR (ИЛИ): Возвращает 1, если хотя бы один из входов равен 1.</p> <p>3. NOT (НЕ): Инвертирует значение входа (0 становится 1, а 1 становится 0).</p>	<p>Таблицы истинности</p> <p>1. AND (И)</p> <table><tr><td>A</td><td>B</td><td>A AND B</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table> <p>3. NOT (НЕ)</p> <table><tr><td>A</td><td>NOT A</td></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table> <p>2. OR (ИЛИ)</p> <table><tr><td>A</td><td>B</td><td>A OR B</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	A AND B	0	0	0	0	1	0	1	0	0	1	1	1	A	NOT A	0	1	1	0	A	B	A OR B	0	0	0	0	1	1	1	0	1	1	1	1
A	B	A AND B																																			
0	0	0																																			
0	1	0																																			
1	0	0																																			
1	1	1																																			
A	NOT A																																				
0	1																																				
1	0																																				
A	B	A OR B																																			
0	0	0																																			
0	1	1																																			
1	0	1																																			
1	1	1																																			
30. Билет	30 Билет 1.																																				
<p>1. Операционные системы (ОС). Важнейшие функции ОС. Управление памятью.</p> <p>Операционная система (ОС) — это комплекс программ, который управляет аппаратными ресурсами компьютера и предоставляет услуги для выполнения программного обеспечения. ОС служит посредником между пользователем и аппаратным обеспечением, обеспечивая взаимодействие между ними.</p> <p>Важнейшие функции операционных систем</p> <p>Управление процессами: ОС отвечает за создание, планирование и завершение процессов. Она управляет выполнением программ, распределяет процессорное время между ними и обеспечивает их взаимодействие.</p> <p>Управление памятью:</p>	<p>ОС управляет файлами на дисках, обеспечивая их создание, удаление, чтение и запись. Она также отвечает за организацию и хранение данных в файловой системе.</p> <p>Управление устройствами ввода-вывода: ОС управляет взаимодействием с устройствами, такими как клавиатура, мышь, принтеры и дисковые накопители. Она предоставляет драйверы для этих устройств и управляет их работой.</p> <p>Обеспечение безопасности и защиты: ОС обеспечивает безопасность данных и защиту от несанкционированного доступа. Она контролирует права доступа пользователей и защищает систему от вредоносного ПО.</p> <p>Управление сетевыми соединениями: ОС управляет сетевыми ресурсами и обеспечивает взаимодействие между компьютерами в сети. Она обрабатывает сетевые запросы и управляет сетевыми протоколами.</p>																																				

ОС управляет оперативной памятью, выделяя и освобождая память для процессов. Она обеспечивает защиту памяти, чтобы один процесс не мог случайно изменить данные другого.

Управление файловой системой:

2. Решить логическую задачу табличным способом.

Трое друзей — Анна, Борис и Виктор — обсуждают свои любимые виды спорта: футбол, теннис и плавание. Известно следующее:

- 1. Анна не любит футбол.
- 2. Борис любит только теннис.
- 3. Виктор не любит теннис.

Необходимо определить, какой вид спорта любит каждый из друзей.

Шаги для решения:

1. Определим переменные:

- A: Анна
- B: Борис
- V: Виктор
- F: Футбол
- T: Теннис
- P: Плавание

2. Создадим таблицу:

Друзья	Футбол	Теннис	Плавание
Анна	0	0	1
Борис	0	1	0
Виктор	1	0	0

Управление виртуальной памятью:

Виртуальная память позволяет использовать дисковое пространство как дополнительную память. Это позволяет запускать больше процессов, чем может вместить физическая память, путем перемещения неактивных страниц в файл подкачки на диске.

3. Заполним таблицу на основе условий:

- Анна не любит футбол: 0 в колонке Футбол.
- Борис любит только теннис: 1 в колонке Теннис, 0 в остальных.
- Виктор не любит теннис: 0 в колонке Теннис.

Теперь мы можем заполнить оставшиеся ячейки:

4. Заполнение таблицы:

- Учитывая, что у каждого друга должен быть один вид спорта, мы видим, что:
 - У Анны может быть только Плавание (так как футбол и теннис она не любит).
 - У Бориса — Теннис.
 - У Виктора остается только Футбол (так как он не любит теннис).

5. Итоговая таблица:

Друзья	Футбол	Теннис	Плавание
Анна	0	0	1
Борис	0	1	0
Виктор	1	0	0

Ответ:

- Анна любит плавание.
- Борис любит теннис.
- Виктор любит футбол.

1. Вычислить значение по следующей формуле при действительных значениях всех переменных:

$$\frac{x+y}{y+1} - \frac{xy-12}{34+x}$$

Реализовать алгоритм на языке программирования

```
x=int(input())
y=int(input())
print((x+y)/(y+1)-(xy-12)/(34+x))
```

2. Вычислить значение по следующей формуле при действительных значениях всех переменных:

$$\frac{a}{c} \frac{b}{d} - \frac{ab-c}{cd}$$

Реализовать алгоритм на языке программирования

```
a=int(input())
b=int(input())
c=int(input())
d=int(input())
print((a*b)/(c*d)-(a*b-c)/(c*d))
```

3. Составить алгоритм решения задачи: с клавиатуры вводится любое число. Определить, какое число (положительное, отрицательное или ноль) и вывести на экран соответствующее сообщение. Реализовать алгоритм на языке программирования.

Алгоритм:

1. Ввод данных: Запросить у пользователя ввод числа.
2. Проверка числа:
 - Если число больше 0, то оно положительное.
 - Если число меньше 0, то оно отрицательное.
 - Если число равно 0, то это ноль.

Python

```
# Шаг 1: Ввод данных
number = float(input("Введите любое число: "))
# Шаг 2: Проверка числа
if number > 0:
    result = "Положительное число"
elif number < 0:
    result = "Отрицательное число"
```

else:

```
result = "Ноль"
```

```
# Шаг 3: Вывод результата
```

```
print(result)
```

3. Вывод результата: Вывести соответствующее сообщение на экран.

4. Составить алгоритм решения задачи: для данного x вычислить значение функции:

Алгоритм:

1. Ввод данных: Запросить у пользователя ввод значения x .

2. Проверка условия:

- Если x меньше или равно 3, вычислить $F(x) = x^2 - 3x + 9$.

- Если x больше 3, вычислить $F(x) = 1/(x^3 - 6)$.

3. Вывод результата: Вывести вычисленное значение $F(x)$ на экран.

5.

$$F(x) = \begin{cases} x^2 - 3x + 9, & \text{если } x \leq 3 \\ \frac{1}{x^3 - 6}, & \text{если } x > 3 \end{cases}$$

5. Реализовать алгоритм на языке программирования.

```
# python
```

```
# Шаг 1: Ввод данных
```

```
x = float(input("Введите значение x: "))
```

```
# Шаг 2: Проверка условия и вычисление функции
```

```
if x <= 3:
```

```
    F_x = x**2 - 3*x + 9
```

```
else:
```

```
    # Проверяем, чтобы избежать деления на ноль
```

```
    if x**3 - 6 == 0:
```

```
        F_x = "Недопустимое значение (деление на ноль)"
```

```
    else:
```

```
        F_x = 1 / (x**3 - 6)
```

```
# Шаг 3: Вывод результата
```

```
print("Значение функции F(x) =", F_x)
```

6. Составить алгоритм решения задачи: для данного x вычислить значение функции:

1. $F(x) = x^2 - x$, если $0 \leq x \leq 1$

2. $F(x) = x^2 - \sin(\pi x)$, если $x > 1$ или $x < 0$

Алгоритм можно описать следующим образом:

1. Ввод значения x .
2. Проверка, попадает ли x в диапазон $[0, 1]$:
 - Если да, вычислить $F(x) = x^2 - x$.
 - Если нет, вычислить $F(x) = x^2 - \sin(\pi x)$.
3. Вывести результат.

$$F(x) = \begin{cases} x^2 - x, & \text{если } 0 \leq x \leq 1 \\ x^2 - \sin \pi x, & \text{если } x > 1 \text{ или } x < 0 \end{cases}$$

Реализовать алгоритм на языке программирования.

```
import math
def c_F(x):
    if 0 <= x <= 1:
        # Если x в диапазоне [0, 1]
        F_x = x**2 - x
    else:
        # Если x < 0 или x > 1
        F_x = x**2 - math.sin(math.pi * x)
    return F_x
# Пример использования
x = float(input("Введите значение x: "))
result = c_F(x)
print(f"F({x}) = {result}")
```

7. Составить алгоритм решения задачи: найти количество цифр в целом числе, заданном с клавиатуры. Реализовать алгоритм на языке программирования.

Алгоритм решения:

Ввести целое число с клавиатуры

Преобразовать число в строку

Подсчитать длину строки (которая будет равна количеству цифр)

Вывести результат

```
number = abs(int(input("Введите целое число: ")))
digit_count = len(str(number))
print(f"Количество цифр в числе: {digit_count}")
```

8. Определить по номеру семестра курс, к которому относится введенный семестр (1 и 2 семестр – 1 курс, 3 и 4 семестр – 2 курс и т.д.). Реализовать алгоритм на языке программирования.

Алгоритм решения:

Ввести номер семестра

Проверить корректность (от 1 до 8)

Вычислить курс по формуле: $\text{курс} = (\text{семестр} + 1) // 2$

Целочисленное деление // автоматически округляет результат

Например, для 1 семестра: $(1+1)//2 = 1$ курс

Для 3 семестра: $(3+1)//2 = 2$ курс

Ввод номера семестра с клавиатуры

Преобразуем введенное значение в целое число

`semester = int(input("Введите номер семестра (1-8): "))`

Проверка корректности введенного семестра

`if semester < 1 or semester > 8:`

`print("Некорректный номер семестра")`

`else:`

Вычисление курса по формуле

`course = (semester + 1) // 2`

Вывод результата

`print(f"Семестр {semester} относится к {course} курсу")`

9. Вычислить по возрасту человека (вводится в клавиатуры как целое число) его принадлежность к возрастной группе: от 0 до 13 – девочка, от 14 до 20 – девушка, от 21 до 70 – женщина, более 70 – старушка. Реализовать алгоритм на языке программирования.

Алгоритм решения:

1. Программа запрашивает у пользователя ввод возраста.

2. Введенное значение преобразуется в целое число с помощью `int()`.

3. С помощью условных операторов `if`, `elif` и `else` программа определяет, к какой возрастной группе принадлежит человек.

4. В конце выводится соответствующее сообщение с указанием возрастной группы.

Запрашиваем возраст у пользователя

`age = int(input("Введите ваш возраст: "))`

Определяем возрастную группу

`if 0 <= age <= 13:`

`g = "девочка"`

`elif 14 <= age <= 20:`

`g = "девушка"`

`elif 21 <= age <= 70:`

`g = "женщина"`

```
elif age > 70:
    g = "старушка"
else:
    g = "Некорректный возраст"
# Выводим результат
print(f"Ваша возрастная группа: {g}")
```

10. Единицы массы пронумерованы следующим образом: 1 – килограмм, 2 – миллиграмм, 3 – грамм, 4 – тонна, 5 – центнер. Дан номер единицы массы и масса тела в этих единицах (вещественное число). Вывести массу данного тела в килограммах. Реализовать алгоритм на языке программирования.

Алгоритм решения:

1. Программа выводит список доступных единиц массы и запрашивает номер единицы у пользователя.
2. Затем она запрашивает массу тела в выбранных единицах.
3. В зависимости от введенного номера единицы массы программа преобразует массу в килограммы:
 - Килограмм — остаётся без изменений.
 - Миллиграмм — делится на 1,000,000.
 - Грамм — делится на 1,000.
 - Тонна — умножается на 1,000.
 - Центнер — умножается на 100.

4. В конце программа выводит массу в килограммах.

Запрашиваем номер единицы массы у пользователя

```
print("Выберите единицу массы:")
print("1 - килограмм")
print("2 - миллиграмм")
print("3 - грамм")
print("4 - тонна")
print("5 - центнер")
unit_number = int(input("Введите номер единицы массы: "))
# Запрашиваем массу тела
mass = float(input("Введите массу тела: "))
# Переводим массу в килограммы в зависимости от выбранной единицы
if unit_number == 1: # килограмм
    mass_in_kg = mass
elif unit_number == 2: # миллиграмм
    mass_in_kg = mass / 1_000_000 # 1 кг = 1,000,000 мг
elif unit_number == 3: # грамм
```

```

    mass_in_kg = mass / 1_000 # 1 кг = 1,000 г
elif unit_number == 4: # тонна
    mass_in_kg = mass * 1_000 # 1 т = 1,000 кг
elif unit_number == 5: # центнер
    mass_in_kg = mass * 100 # 1 ц = 100 кг
else:
    mass_in_kg = None
    print("Некорректный номер единицы массы.")
# Выводим результат
if mass_in_kg is not None:
    print(f"Масса тела в килограммах: {mass_in_kg:.6f} кг")

```

11. Ежемесячная стипендия студента составляет сумму А рублей, а расходы на проживание превышают стипендию и составляют сумму В рублей в месяц. Рост цен ежемесячно увеличивает расходы на 3%. Определите сумму, которую следует одновременно попросить у родителей, чтобы прожить учебный год (10 месяцев), используя только эти деньги и стипендию.

```

def cf(A, B):
    # Условия
    m = 10 # Учебный год длится 10 месяцев
    t = 0 # Инициализация общей суммы расходов
    # Расчет расходов на каждый месяц с учетом повышения цен

    for month in range(m):
        t += B # Добавляем расходы за текущий месяц
        B *= 1.03 # Увеличиваем расходы на 3% на следующий месяц
    # Сумма стипендий за 10 месяцев

    ssh = A * m

    # Рассчитываем сумму, которую следует попросить у родителей
    fnd = t - ssh # Если fnd меньше 0, значит, стипендии хватает
    return max(fnd, 0) # Возвращаем неотрицательное значение

# Пример ввода
A = float(input("Введите сумму стипендии (А рублей): "))
B = float(input("Введите первоначальные расходы на проживание (В рублей): "))
# Вычисление необходимой суммы

na = cf(A, B)
print(f"Сумма, которую следует попросить у родителей: {na:.2f} рублей.")
# Ввод значений стипендии и расходов
A = float(input("Введите сумму стипендии (А рублей): "))

```

```
B = float(input("Введите первоначальные расходы на проживание (В рублей): "))
```

```
m = 10          # Условия
t = 0           # Учебный год длится 10 месяцев
                # Инициализация общей суммы расходов
                # Расчет расходов на каждый месяц с учетом повышения цен

for month in range(m):
    t += B       # Добавляем расходы за текущий месяц
    B *= 1.03    # Увеличиваем расходы на 3% на следующий месяц
                # Сумма стипендий за 10 месяцев

ssh = A * m     # Рассчитываем сумму, которую следует попросить у родителей

fnd = t - ssh   # Если fnd меньше 0, значит, стипендии хватает
fnd = max(fnd, 0) # Возвращаем неотрицательное значение
                # Вывод результата

print(f"Сумма, которую следует попросить у родителей: {fnd:.2f} рублей.")
```

12. Дано $A = A_{716}$, $B = 2518$. Найдите сумму $A + B$. Ответ укажите в двоичной системе счисления.

$$A_{716} = A \times 16^3 + 7 \times 16^2 + 1 \times 16^1 + 6 \times 16^0$$

$$A_{716} = 10 \times 4096 + 7 \times 256 + 1 \times 16 + 6 \times 1 = 40960 + 1792 + 16 + 6 = 42774$$

$$A + B = 42774 + 2518 = 45292$$

- 45292 в двоичной системе:
- $45292 \div 2 = 22646$ (0)
- $22646 \div 2 = 11323$ (0)
- $11323 \div 2 = 5661$ (1)
- $5661 \div 2 = 2830$ (1)
- $2830 \div 2 = 1415$ (0)
- $1415 \div 2 = 707$ (1)
- $707 \div 2 = 353$ (1)
- $353 \div 2 = 176$ (1)
- $176 \div 2 = 88$ (0)
- $88 \div 2 = 44$ (0)

- $44 \div 2 = 22$ $44 \div 2 = 22$ (0)
- $22 \div 2 = 11$ $22 \div 2 = 11$ (0)
- $11 \div 2 = 5$ $11 \div 2 = 5$ (1)
- $5 \div 2 = 2$ $5 \div 2 = 2$ (1)
- $2 \div 2 = 1$ $2 \div 2 = 1$ (0)
- $1 \div 2 = 0$ $1 \div 2 = 0$ (1)

45292 в двоичной системе: 1011001110111100

Таким образом, сумма A+B в двоичной системе счисления равна **1011001110111100**

13. Даны 4 целых числа, записанных в различных системах счисления: 3110, F116, 2618, 7118. Сколько среди них чисел, двоичная запись которых содержит ровно 5 единиц?

$$3110_{10} = 1100001101102 \quad 3110_{10} = 110000110110_2.$$

$$F116 = F \times 16_2 + 1 \times 16_1 + 6 \times 16_0 = 15 \times 256 + 1 \times 16 + 6 \times 1 = 3840 + 16 + 6 = 3862.$$

$$\text{Двоичная запись: } 3862_{10} = 1111000111102 \quad 3862_{10} = 111100011110_2.$$

$$\mathbf{2618_{10} = 1010001100102} \quad \mathbf{2618_{10} = 101000110010_2.}$$

$$7118 = 7 \times 8_3 + 1 \times 8_2 + 1 \times 8_1 + 8 \times 8_0 = 7 \times 512 + 1 \times 64 + 1 \times 8 + 8 \times 1 = 3584 + 64 + 8 + 8 = 3656.$$

$$\text{Двоичная запись: } 3656_{10} = 1110010110002 \quad 3656_{10} = 111001011000_2.$$

- 3110: 6 единиц
- F116: 8 единиц
- **2618: 5 единиц**
- 7118: 6 единиц

14. Найдите значение выражения $1116 + 118 : 112$.

Ответ запишите в двоичной системе счисления.

$$1116 = 1 \times 16^3 + 1 \times 16^2 + 1 \times 16^1 + 6 \times 16^0 =$$

$$= 1 \times 4096 + 1 \times 256 + 1 \times 16 + 6 \times 1 =$$

$$= 4096 + 256 + 16 + 6 = 4374.$$

$$118 = 1 \times 16^2 + 1 \times 16^1 + 8 \times 16^0 =$$

$$= 1 \times 256 + 1 \times 16 + 8 \times 1 =$$

$$= 256 + 16 + 8 = 280.$$

$$112 = 1 \times 16^2 + 1 \times 16^1 + 2 \times 16^0 =$$

$$= 1 \times 256 + 1 \times 16 + 2 \times 1 =$$

$$= 256 + 16 + 2 = 274.$$

1. Выполним операции:

- Сначала делим 118 на 112:

$280:274 \approx 1.021$ (но нам нужно целое число, поэтому это будет 1, если округлить вниз) .

Теперь добавим результат к 1116:

$$4374+1=4375.$$

2. Преобразуем результат обратно в двоичную систему:

4375 в двоичной системе:

Делим на 2 и записываем остатки:

$$4375 \div 2 = 2187 \text{ (1)}$$

$$2187 \div 2 = 1093 \text{ (1)}$$

$$1093 \div 2 = 546 \text{ (1)}$$

$$546 \div 2 = 273 \text{ (0)}$$

$$273 \div 2 = 136 \text{ (1)}$$

$$136 \div 2 = 68 \text{ (0)}$$

$$68 \div 2 = 34 \text{ (0)}$$

$$34 \div 2 = 17 \text{ (0)}$$

$$17 \div 2 = 8 \text{ (1)}$$

$$8 \div 2 = 4 \text{ (0)}$$

$$4 \div 2 = 2 \text{ (0)}$$

$$2 \div 2 = 1 \text{ (0)}$$

$$1 \div 2 = 0 \text{ (1)}$$

Теперь, записывая остатки в обратном порядке, получаем:

4375 в двоичной системе: 100001101011₂

Таким образом, значение выражения $1116+118:112$ в двоичной системе счисления равно **100001101011₂**.

15. Даны 4 целых числа, записанных в шестнадцатеричной системе: A8, AB, B5, CA. Сколько среди них чисел, больших, чем 2658?

Чтобы определить, сколько из данных чисел в шестнадцатеричной системе (A8, AB, B5, CA) больше 2658 в десятичной системе, сначала преобразуем каждое из этих чисел в десятичную систему.

1. **A8 (шестнадцатеричная система):**

$$A8 = A \times 16^1 + 8 \times 16^0 = 10 \times 16 + 8 \times 1 = 160 + 8 = 168.$$

2. **AB (шестнадцатеричная система):**

$$AB = A \times 16^1 + B \times 16^0 = 10 \times 16 + 11 \times 1 = 160 + 11 = 171.$$

3. **B5 (шестнадцатеричная система):**

$$B5 = B \times 16^1 + 5 \times 16^0 = 11 \times 16 + 5 \times 1 = 176 + 5 = 181.$$

4. **СА (шестнадцатеричная система):**

$$CA = C \times 16^1 + A \times 16^0 = 12 \times 16 + 10 \times 1 = 192 + 10 = 202.$$

Теперь у нас есть следующие десятичные значения:

$$A8 = 168$$

$$AB = 171$$

$$B5 = 181$$

$$CA = 202$$

Теперь сравним каждое из этих значений с 2658:

$$168 < 2658$$

$$171 < 2658$$

$$181 < 2658$$

$$202 < 2658$$

Все четыре числа меньше 2658.

Таким образом, среди них **нет** чисел, больших, чем 2658.

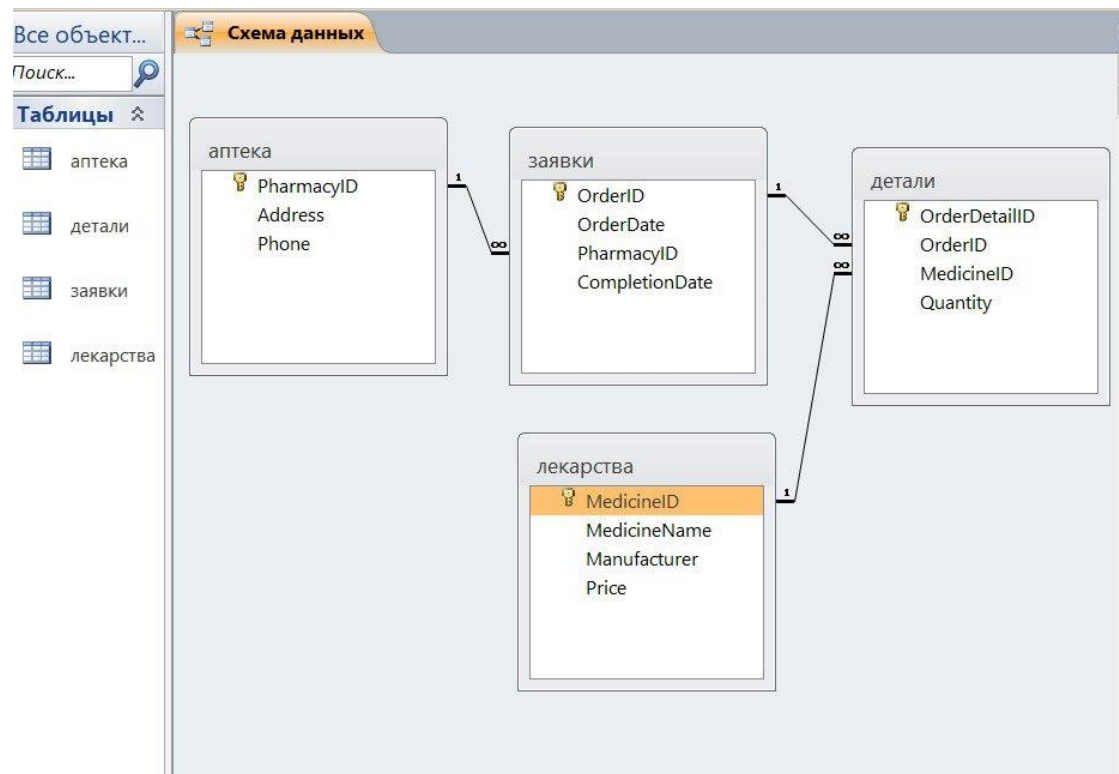
Ответ: **0**.

16. Аптечный склад осуществляет оптовую продажу лекарственных препаратов различным аптекам республики. Необходимо спроектировать базу данных «Аптечный склад» информация которой будет использоваться для учета продаж аптеками лекарственных препаратов. В базе данных должна храниться информация:

о лекарствах: код лекарства, название лекарства, производитель, цена;

об аптеках: номер аптеки, название аптеки, адрес и телефон аптеки;

о заявках: номер заявки, дата составления заявки, номер аптеки, дата выполнения заявки;



1. Таблица "Лекарства" (Medicines)

Код лекарства (MedicineID): Автоматический номер (Primary Key)

Название лекарства (MedicineName): Текст

Производитель (Manufacturer): Текст

Цена (Price): Число (формат: валютный)

2. Таблица "Аптеки" (Pharmacies)

Номер аптеки (PharmacyID): Автоматический номер (Primary Key)

Название аптеки (PharmacyName): Текст

Адрес (Address): Текст

Телефон (Phone): Текст

3. Таблица "Заявки" (Orders)

Номер заявки (OrderID): Автоматический номер (Primary Key)

Дата составления заявки (OrderDate): Дата/время

Номер аптеки (PharmacyID): Число (Foreign Key, ссылается на PharmacyID из таблицы "Аптеки")

Дата выполнения заявки (CompletionDate): Дата/время

4. Таблица "Детали заявки" (OrderDetails)

Номер детали заявки (OrderDetailID): Автоматический номер (Primary Key)

Номер заявки (OrderID): Число (Foreign Key, ссылается на OrderID из таблицы "Заявки")

Код лекарства (MedicineID): Число (Foreign Key, ссылается на MedicineID из таблицы "Лекарства")

Количество (Quantity): Число

Связи между таблицами

Связь между таблицами "Заявки" и "Аптеки": один ко многим (одна аптека может иметь много заявок).

Связь между таблицами "Детали заявки" и "Заявки": один ко многим (одна заявка может содержать много деталей).

Связь между таблицами "Детали заявки" и "Лекарства": один ко многим (одно лекарство может быть в многих деталях заявок).

Создание базы данных в Access

Откройте Microsoft Access и создайте новую базу данных.

Создайте таблицы, используя описанную выше структуру.

Установите связи между таблицами, используя окно "Связи" (Relationships).

Заполните таблицы данными для тестирования.

Пример SQL-кода для создания таблиц

Если вы хотите использовать SQL для создания таблиц, вы можете использовать следующий код:

17. Отделы крупного торгового дома ежедневно продают различные виды товаров и ведут учет сведений о проданных товарах. Необходимо спроектировать базу данных «Торговля», информация которой будет использоваться для анализа выполнения плана реализации продукции в отделах, определения товаров, пользующихся наибольшим спросом и др.

Для проектирования базы данных «Торговля» в Microsoft Access, которая будет использоваться для учета продаж товаров в различных отделах крупного торгового дома, вам необходимо создать несколько таблиц, отражающих основные аспекты бизнеса. Ниже представлена структура базы данных с описанием необходимых таблиц и их полей.

1. Таблица "Товары" (Products)

Код товара (ProductID): Автоматический номер (Primary Key)

Название товара (ProductName): Текст

Категория (Category): Текст

Цена (Price): Число (формат: валютный)

Количество на складе (StockQuantity): Число

2. Таблица "Отделы" (Departments)

Код отдела (DepartmentID): Автоматический номер (Primary Key)

Название отдела (DepartmentName): Текст

Менеджер (Manager): Текст

Контактный телефон (ContactPhone): Текст

3. Таблица "Продажи" (Sales)

Номер продажи (SaleID): Автоматический номер (Primary Key)

Дата продажи (SaleDate): Дата/время

Код товара (ProductID): Число (Foreign Key, ссылается на ProductID из таблицы "Товары")

Код отдела (DepartmentID): Число (Foreign Key, ссылается на DepartmentID из таблицы "Отделы")

Количество (Quantity): Число

Итого (TotalAmount): Число (формат: валютный, рассчитывается как Цена * Количество)

4. Таблица "Категории" (Categories) (опционально)

Код категории (CategoryID): Автоматический номер (Primary Key)

Название категории (CategoryName): Текст

Описание (Description): Текст

Связи между таблицами

Связь между таблицами "Продажи" и "Товары": многие к одному (много продаж могут относиться к одному товару).

Связь между таблицами "Продажи" и "Отделы": многие к одному (много продаж могут относиться к одному отделу).

Связь между таблицами "Товары" и "Категории" (если используется): многие к одному (много товаров могут относиться к одной категории).

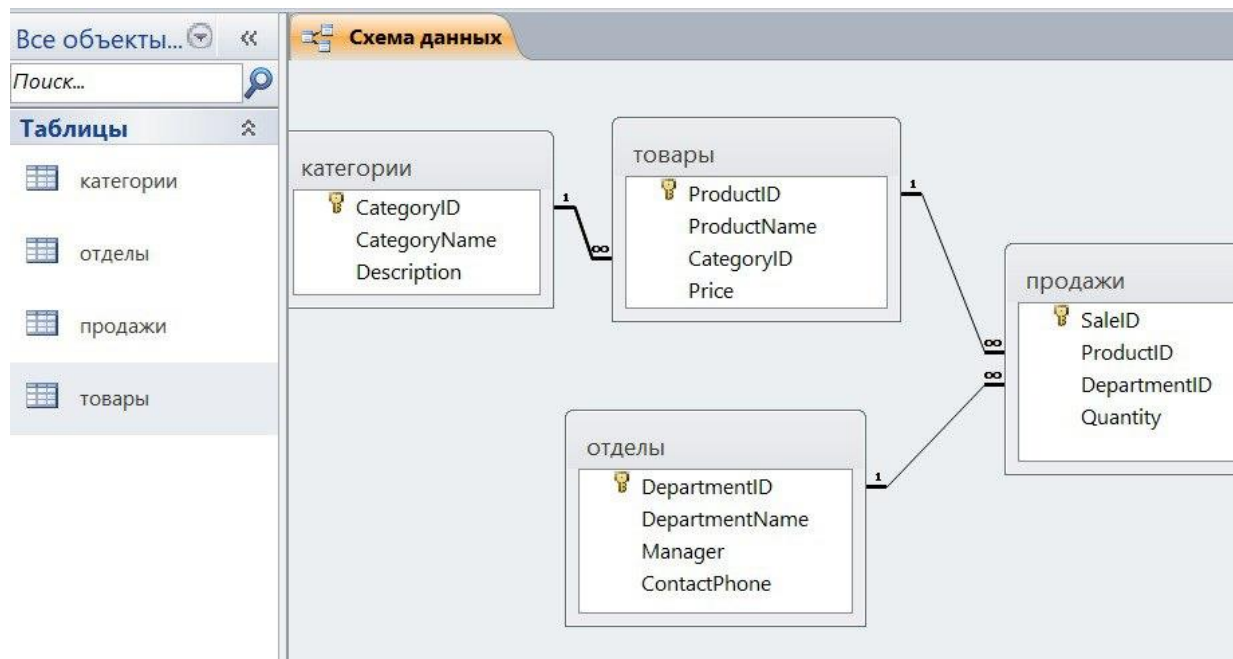
Создание базы данных в Access

Откройте Microsoft Access и создайте новую базу данных.

Создайте таблицы, используя описанную выше структуру.

Установите связи между таблицами, используя окно "Связи" (Relationships).

Заполните таблицы данными для тестирования и анализа.



18. В базе данных должна храниться информация:

об отделах: код отдела, наименование отдела, ФИО начальника отдела, телефон, объем реализации в день; о товарах: артикул товара, наименование товара, единица измерения, розничная цена товара;

19. Вычислить в таблице значения Годового объема продаж, а также Среднее, Минимальное и Максимальные значения квартального объема продаж.

20. Построить круговую диаграмму Годового объема продаж по торговым маркам. Добавить необходимые обозначения. Создайте схему по образцу