

Лекция1

1.Информация, как продукт взаимодействия данных и адекватных им методов.

Информатика-это научная и прикладная область знаний, изучающая законы, методы и способы накопления, обработки и передачи информации с помощью компьютерных и других технических средств

Информация возникает и существует в момент диалектического взаимодействия объективных данных и субъективных методов.

1. Данные (фиксируемые в виде определенных сигналов воспринимаемые факты окружающего мира) ОБЪЕКТИВНЫ
2. Методы (преобразуют, транспортируют, дают возможность потреблять данные) СУБЪЕКТИВНЫ

2. Свойства информации.

АТРИБУТИВНЫЕ(присутствуют всегда):

- неотрывность информации от физического носителя и языковая природа информации;

Хотя информация и неотрывна от физического носителя и имеет языковую природу она не связана жестко ни с конкретным языком, ни с конкретным носителем

- дискретность;

Знания характеризуют отдельные фактические данные, закономерности и свойства изучаемых объектов, которые распространяются в виде различных сообщений, состоящих из буквы, цифры, символа, знака

- непрерывность.

Информация имеет свойство сливаться с уже зафиксированной и накопленной ранее, тем

самым способствуя поступательному развитию и накоплению.

ПРАГМАТИЧЕСКИЕ(появляются в процессе использования):

полнота:

Информация полна, если ее достаточно для понимания и принятия решений. Не полнота информации сдерживает принятие решений или может повлечь ошибки

достоверность:

Информация достоверна, если она не искажает истинное положение дел. Недостоверная информация может привести к неправильному пониманию или принятию неправильных решений

адекватность:

Степень соответствия реальному объективному состоянию дел зависит от адекватности информации.

доступность:

Информация становится понятной, если она выражена языком, доступным людям, для которых она предназначена

актуальность:

При работе в постоянно изменяющихся условиях важно иметь актуальную, т. е. соответствующую действительности, информацию

ДИНАМИЧЕСКИЕ (**характеризуют изменение информации во времени**): рост:

Определяет свойство многократного распространения или повторяемости информации

старение

3. Виды информации.

ПО СПОСОБАМ ВОСПРИЯТИЯ:

визуальная аудиальная тактильная обонятельная вкусовая

ПО ФОРМЕ ПРЕДСТАВЛЕНИЯ:

текстовая числовая графическая звуковая

ПО ОБЩЕСТВЕННОМУ ЗНАЧЕНИЮ:

массовая специальная личная

4. Информационная система. Схемы управления информационных системам

- это взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели.

Схемы управления информационных систем:

Любая информационная система может действовать по правилам разомкнутой или замкнутой схемы управления.

В **разомкнутой информационной системе** получаемая потребителем информация используется произвольно. От потребителя в информационную систему ничего не поступает (библиотечный каталог)

В **замкнутой информационной системе** существует тесная связь между ее структурой и потребителем. Это достигается за счет введения в ее структуру канала обратной связи (система бронирования билетов)

5. Информационные процессы.

Информационный процесс - совокупность последовательных действий (операций), производимых над информацией (в виде данных, сведений, фактов, идей, гипотез, теорий и пр.) для получения какого-либо результата (достижения цели).

К основным информационным процессам, изучаемым в курсе информатики, относятся:

- ● поиск
- ● отбор
- ● хранения
- ● передача
- ● кодирование
- ● обработка
- ● защита информации (сл. 18)
-

6. Информационные революции. Их важность с точки зрения информатики.

Этапы появления средств и методов обработки информации, вызвавшие кардинальные изменения в обществе, определяются как **информационные революции**.

Информационные революции привели к появлению многих новых областей знаний, одной из которых является информатика.

ПЕРВАЯ ИНФОРМАЦИОННАЯ РЕВОЛЮЦИЯ

Связана с изобретением письменности, обусловившей гигантский качественный скачок в развитии цивилизации. Появилась возможность накопления знаний и их передачи последующим поколениям. С позиций информатики это можно оценить как появление средств и методов накопления информации.

ВТОРАЯ ИНФОРМАЦИОННАЯ РЕВОЛЮЦИЯ (середина 16 века)

Связана с изобретением книгопечатания, изменившего человеческое общество, культуру и организацию деятельности самым радикальным образом. С точки зрения информатики, значение этой революции в том, что она выдвинула качественно новый способ хранения информации.

ТРЕТЬЯ ИНФОРМАЦИОННАЯ РЕВОЛЮЦИЯ (конец 19 века)

Связана с изобретением электричества, благодаря которому появились телеграф, телефон, радио, позволяющие оперативно передавать и

накапливать информацию в любом объеме. Этот этап важен для информатики, прежде всего тем, что ознаменовал появления средств информационной коммуникации.

ЧЕТВЕРТАЯ ИНФОРМАЦИОННАЯ РЕВОЛЮЦИЯ (70-е годы 20 века)

Связана с изобретением микропроцессорной технологии и появлением персонального компьютера. Толчком к четвертой инф революции послужило изобретение в середине 40-х годов электронно-вычислительной машины (ЭВМ).

7. Кодирование текстовой и числовой информации. Системы счисления.

Для реализации работы с данными, относящимися к различным типам, используется прием кодирования, то есть выражение данных одного типа через данные другого типа.

Естественные человеческие языки - это не что иное, как системы кодирования понятий для выражения мыслей посредством речи.

Система кодирования существующая в вычислительной технике называется двоичным кодированием и основана на представлении данных последовательностью всего двух знаков: **0** и **1**.

Эти знаки называются **двоичными цифрами**, по-английски — **binary digit** или сокращенно bit (**бит**).

Одним битом могут быть выражены два понятия: **0** или **1** (**да** или **нет**, **истина** или **ложь**).

Если количество битовувеличить до двух, то уже можно выразить четыре различных понятия:

00 01 10 11

0123

Тремя битами можно закодировать восемь различных значений:

000 001 010 011 100 101 110 111

0 1234567

Система, основанная на16разрядном кодировании символов, получила **название универсальной — UNICODE**. Шестнадцать разрядов позволяют обеспечить уникальные коды для 65 536 различных символов — этого поля достаточно для размещения в одной таблице символов большинства языков планеты.На текущий момент стандарт является основным в Интернете. Символический метод записи чисел, когда каждому числу дается уникальное представление.

Системыразделяются на: 1. позиционные

- • двоичные
- • восьмеричные
- • десятичные
- • шестнадцатеричные
-

2. непозиционные римская

8. Правила перевода из десятичной системы счисления в любую другую позиционную систему счисления и наоборот. ДЕСЯТИЧНАЯ СС:

Основание системы - 10.

Содержит 10 цифр: 1,2,3,4,5,6,7,8,9,0.

Любое десятичное число можно представить как сумму степеней числа 10.

Пример:

$$3751_{10} = 3 \cdot 10^3 + 7 \cdot 10^2 + 5 \cdot 10^1 + 1 \cdot 10^0$$

ВОСЬМЕРИЧНАЯ СС: Основание системы - 8.

Содержит 8 цифр: 0,1,2,3,4,5,6,7.

Любое восьмеричное число можно представить как сумму степеней числа 8.

Пример:

3751, 1607, 62

ШЕСТНАДЦАТИРИЧНАЯ СС: Основание системы - 16.

Содержит 16 символов: 0,1,2,3,4,5,6,7,8,9, A, B, C, D, E, F
1011 12131415

Любое шестнадцатеричное число можно представить как сумму степеней числа 16.

Примеры шестнадцатеричных чисел : 32FD8, A6E, 156, 3751.

ДВОИЧНАЯ СС: Основание системы - 2.

Содержит 2 символа: 0,1

Любое двоичное число можно представить как сумму степеней числа 2.

Примеры двоичных чисел : 101101, 101, 1101011001.

СЛАЙД 14, 15, 16, 17 в обратном направлении (там картинки наглядно объясняют происходящее)

- Для перехода из восьмеричной системы счисления в десятичную необходимо восьмеричное число представить в виде суммы степеней восьмерки и найти ее десятичное значение.

Пример: $35_8 = 3 \cdot 8^1 + 5 \cdot 8^0 = 24 + 5 = 29_{10}$

- Для перехода из шестнадцатеричной системы счисления в десятичную необходимо шестнадцатеричное число представить в виде суммы степеней шестнадцати и найти ее десятичное значение.

Пример: $3A5_{16} = 3 \cdot 16^2 + 10 \cdot 16^1 + 5 \cdot 16^0 = 768 + 160 + 5 = 933_{10}$

9. Правила перевода из двоичной системы счисления в восьмеричную и шестнадцатеричную системы и наоборот.

Основание системы -2.

Содержит 2 символа: 0,1

Любое двоичное число можно представить как сумму степеней числа 2.

Примеры двоичных чисел : 101101, 101,1101011001.

- из двоичной в восьмеричную: Разбить двоичное число на триады справа налево. Заменить каждую триаду восьмеричной цифрой. Недостающие позиции в крайней левой триаде дополнить нулями.

Пример: $10010101_2 = 010 \cdot 010 \cdot 101 = 225_8$

- из восьмеричной в двоичную: Каждую цифру восьмеричного числа представить в двоичном виде (три двоичных разряда). Отбросить нули в крайней левой позиции

Пример: $225_8 = 010 \cdot 010 \cdot 101 = 10\ 010\ 101_2$

- из двоичной в шестнадцатеричную: Разбить двоичное число на четвертки справа налево. Заменить каждую четвертку двоичных цифр шестнадцатеричной цифрой. Недостающие позиции в крайней левой четвертки дополнить нулями.

Пример: $110010101_2 = 0001 \cdot 1001 \cdot 0101 = 195_{16}$

- из шестнадцатеричную в двоичную: Каждую цифру шестнадцатеричного

числа представить в двоичном виде (4 двоичных разряда). Отбросить нули в крайних левых позициях

Пример: $195_{16} = 0001 \cdot 1001 \cdot 0101 = 11\ 001\ 0101_2$

10. Кодирование растровых изображений. Преимущества и недостатки.

Графическая и звуковая информация в ЭВМ представляется в **дискретной** форме.

Преобразование информации из аналоговой формы в дискретную называется дискретизацией, при этом графическое изображение или звуковой сигнал разбивается на отдельные элементы и каждому элементу присваивается в виде кода.

В соответствии с методами, применяемыми для представления изображений, их можно разделить на две категории: растровые и векторные.

Растровое изображение представляет собой **набор точек** (элементов изображения), которые называются пикселями.

Основные форматы графических файлов:

.bmp – хранит информации о каждой точке изображения, **.gif**, **.jpg**, **.tiff**, **.png** – используется сжатие файла.

Векторное изображение представляет собой **набор линий и дуг**. Основные форматы графических файлов:

.wmf -универсальный формат для Windows-приложений.

.eps- формат файлов, предназначенный для обмена графическими данными между различными приложениями.

.cdr -файл проекта, созданный в программе CorelDRAW.

Представление растровых изображений: Линейные координаты и яркость каждой точки можно выразить с помощью целых чисел, т.е. можно сказать, что растровое кодирование позволяет использовать двоичный код для представления графических данных.

Для кодирования цветных графических изображений применяется принцип **декомпозиции** произвольного цвета на основные составляющие. В качестве таких составляющих используют три основных цвета: **красный (Red , R)**, **зеленый (Green , G)** и **синий (Blue , B)**.

Такая система кодирования называется системой **RGB**.

Расчет объема графической информации сводится к вычислению произведения количества точек на изображении на количество разрядов, необходимых для кодирования цвета одной точки.

Например, для цветной картинки, составленной из 256 цветов в графическом режиме монитора 640 x 480, требуется объем видеопамати, равный: $8 \cdot 640 \cdot 480 = 2457600$ бит = 307200 байт = 300 Кбайт.

Основные преимущества:

- позволяет создать максимально реалистичные изображения (к примеру, фотографии), которые можно использовать в рекламном производстве, оформлении сайтов и т.д.;
- позволяет создать рисунок любой сложности, с плавными переходами цветов, с различной глубиной цвета, большим количеством деталей;
- намного большая распространенность, по сравнению с векторной, растровую графику можно встретить на плакатах, сайтах и других часто встречаемых местах;
- открыть файл с растровым изображением намного проще, так как большинство программ для просмотра изображений поддерживают форматы, в котором хранятся такие изображения;
- возможность быстро обработать изображение.

Недостатки:

- потеря качества при значительном увеличении (картинка становится «зернистой», то есть, становится видно те самые пиксели);
- если изображение имеет большое разрешение, то имеет и большой размер файла;
- невозможно масштабировать без потери первоначального качества;

- невозможно произвольно поворачивать изображение (только от 90 градусов) без искажения картинки;
- самое простое изображение будет занимать больше места, чем такое же векторное;
- невозможно вывести на печать навекторный графопостроитель.

11. Кодирование векторных изображений.

Преимущества и недостатки.

Векторные изображение строятся с помощью математических описаний объектов (например, прямая описывается уравнением, окружность - координатами центра и радиусом).

Всеобъекты имеют свойства (атрибуты): толщины, цвет, тип линий. Такое описание заставляет устройство само рисовать изображение, а не воспроизводить комбинацию пикселей.

Преимущества:

- Масштабирование, растягивание, перемещение без ухудшения качества изображений.
- Небольшой размер простых иллюстраций, упрощающий хранение и

отправку файлов заказчикам.

- Универсальность применения: на сайте, для контекстной и таргетированной рекламы, в полиграфии.

Основные недостатки:

- Большой размер файла с высокой детализацией графики и, как следствие, повышенные требования к производительности компьютера.
- Ограниченность в использовании эффектов: теней, градиентов, свечения и пр.
- Сложность создания детализированных фотореалистичных изображений.
- Трудности в совместимости форматов с приложениями из-за конкуренции фирм-производителей.

12. Кодирование звуковой информации.

При преобразовании звуковой информации в цифровую форму ее подвергают **дискретизации и квантованию**.

Дискретизация заключается в замерах величины аналогового сигнала огромное количество раз в секунду. Полученной величине аналогового сигнала сопоставляется определенное значение из заранее выделенного диапазона: 256 (8бит) или 65536 (16бит). Приведение в соответствие уровня сигнала определенной величине диапазона есть **квантование**.

Чем больше разделен сигнал в секунду, тем выше **частота дискретизации**.

Именно по ней мы часто и определяем качество звука: 44 кГц, 96 кГц и т.д. - т.е. на сколько тысяч таких участков была разделена секунда звука.

НА СЛАЙДЕ 36 ПОКАЗАН ГРАФИК

13. Этапы создания ЭВМ.

1. Первое поколение ЭВМ (1946 — начало 50-х годов):

- • Использование электронных ламп в качестве элементной базы.
- • Большие размеры устройств и низкая производительность. Примером является ENIAC, способный выполнять 300 операций умножения или 5000 операций сложения в секунду.
- Проблемы: сложность программирования, высокая стоимость и

ненадежность, так как лампы часто выходили из строя.

- Основное назначение: научно-технические расчеты.

2. Второе поколение ЭВМ (конец 50-х — начало 60-х годов):

- Переход на транзисторы, что позволило уменьшить размеры и повысить производительность компьютеров.
- Быстродействие достигло 10⁵-10⁶ операций в секунду, ёмкость оперативной памяти увеличилась до 1000 Кбайт.
- Появление первых языков программирования высокого уровня, таких как Фортран и Кобол, что упростило процесс программирования.

3. Третье поколение ЭВМ (1965-1971 годы):

- • Использование микросхем в качестве элементной базы.
- • Увеличение производительности до 10 млн операций в секунду и уменьшение размеров устройств.
- Совместимость и возможность подключения различных периферийных устройств стали ключевыми особенностями, что позволило создавать сети и интегрировать новые устройства.

4. Четвертое поколение ЭВМ (с 1971 года по настоящее время):

- Использование микропроцессоров, больших и сверхбольших интегральных схем.
- Развитие персональных компьютеров и суперкомпьютеров. Современные ПК стали компактными, высокопроизводительными и доступными для массового пользователя.
- Примером суперкомпьютера является El Capitan, который выполняет 1,742 квинтиллиона операций в секунду (эксафлопс).

14. Принципы построения компьютера, сформулированные Дж. фон Нейманом.

- • Программа и данные хранятся в единой памяти.
- • Управление компьютером осуществляется с помощью инструкций, выполняемых последовательно.
- Все операции делятся на арифметические и логические, выполняемые центральным процессором.

- Использование памяти с произвольным доступом (RAM) для хранения временных данных.

Лекция 4

15. Последовательность действий по созданию ЦП.

1. Изготовление кристалла:

- Выращивание монокристалла кремния цилиндрической формы из особо чистого кварцевого песка.
- Разрезание кристалла на пластины толщиной около 0,9 мм и их полировка до зеркального состояния

2. Создание чипов:

- На пластинах создаются интегральные схемы методом фотолитографии и травления в специальных чистых помещениях.
- Происходит повторная чистка пластин перед тестированием.

3. Тестирование:

- Выборочное тестирование процессоров для проверки их функциональности с использованием специального оборудования.

4. Корпусировка:

- Готовые пластины разрезаются на отдельные процессорные ядра, которые помещаются в корпуса с контактами для установки на материнскую плату.

16. Основные параметры ЦП.

- **Тактовая частота:** определяет количество элементарных операций, выполняемых процессором за единицу времени, измеряется в ГГц (1 Герц соответствует выполнению одной операции за одну секунду). Современные компьютеры заряжены процессорами, тактовая частота которых колеблется от 1 до 4 ГГц.
- **Разрядность:** определяет количество бит данных, которые процессор может обработать за один такт.
- **Внутренняя разрядность процессора** определяет, какое количество битов он может обрабатывать одновременно при выполнении арифметических операций.
- **Внешняя разрядность процессора** определяет сколько битов одновременно он может принимать или передавать во внешние устройства. • Современные процессоры семейства Intel являются 32- и 64-разрядными. • **Рабочее напряжение:** обычно не превышает 3 В и обеспечивается мат.

платой.

- **Кэш-память:** память внутри процессора для хранения данных, к которым часто обращается процессор. Кэш бывает трех уровней (L1, L2, L3), различающихся скоростью и объемом.
- Самой быстрой памятью является кэш-память первого уровня (L1-cache), она является частью процессора, т.к. расположена на одном с ним кристалле и входит в состав функциональных блоков, без неё процессор не сможет функционировать. Память L1 работает на частоте процессора, объем этой памяти обычно невелик — не более 64 Кб.

- Второй по быстродействию является L2-cache (в отличие от L1 ее можно отключить с сохранением работоспособности процессора), память L2 обычно расположена на отдельном кристалле, но в границах процессора. Объем L2 от 128 Кб до 1—4 Мб.
- L3-cache наименее быстродействующий и обычно выполняется на отдельных быстродействующих микросхемах с расположением на материнской плате и имеет объем один и больше Мбайт. L3-cache значительно быстрее чем оперативная память.

17. Назначение чипсета.

Чипсет представляет собой набор микросхем системной логики, которые обеспечивают взаимодействие между компонентами ПК. Он состоит из двух микросхем:

Микросхема северного моста обеспечивает работу с наиболее скоростными подсистемами. Он содержит: контроллер системной шины, посредством которого происходит взаимодействие с процессором; контроллер памяти, осуществляющий работу с системной памятью; контроллер графической шины, обеспечивающий взаимодействие с графической подсистемой; контроллер шины связи с южным мостом. Частота работы этой микросхемы равна тактовой частоте материнской платы. Современные North Bridge работают на высоких тактовых частотах и поэтому дополнительно оборудованы устройствами охлаждения.

Южный мост обеспечивает работу с более медленными компонентами системы и периферийными устройствами. Он содержит: контроллер, обеспечивающий работу с внутренними накопителями, в частности с винчестерами и оптическими дисковыми; USB-контроллеры; контроллер,

который поддерживает работу внешних портов.

Лекция 5

18. Основные шины.

- **Шина данных:** передает данные между оперативной памятью и процессором.
- **Адресная шина:** передает адресачеек памяти или устройств ввода-вывода.
- **Командная шина:** передает команды процессору для выполнения операций. Команды представлены в виде байтов. Простые команды вкладываются в один байт, но есть и такие команды, для которых нужно два, три и больше байта.

19. Устройства памяти. ПЗУ и CMOS.

- **ПЗУ (постоянное запоминающее устройство):** содержит базовую систему ввода-вывода (BIOS), обеспечивающую начальную загрузку и

тестирование системы при включении. Основное назначение программ этого пакета состоит в том, чтобы проверить состав и работоспособность компьютерной системы и обеспечить взаимодействие с клавиатурой, монитором, жестким диском.

- **CMOS:** разновидность ПЗУ. Это память с невысоким быстродействием и минимальным электропотреблением от батарейки 3,6 V.
- Используется для хранения информации о конфигурации и составе оборудования компьютера, а также о режимах его работы (объем оперативной памяти, количество и тип гибких дисков, характеристики жестких дисков, порядок загрузки, энергосбережения, использование системных и встроенных контроллеров и т.д.).
- В CMOS-памяти хранится информация о текущих показаниях часов и конфигурации компьютера
- Содержимое CMOS изменяется специальной программой setup, находящейся в BIOS.

20. Устройства памяти. Динамическая и статическая память.

- **Динамическая память (DRAM):** представляют собой микроконденсаторы. Это наиболее распространенный и экономичный

вид памяти. Из-за переходных процессов запись данных происходит сравнительно медленно, к тому же заряды достаточно быстро (сотые доли секунды) рассеиваются в пространстве, что вызывает необходимость постоянной подзарядки ячеек памяти но дешевле.

- **Статическая память (SRAM):** можно представить как электронные микроэлементы – триггеры, состоящее из нескольких транзисторов. В триггере хранится состояние включен-выключен, что обеспечивает более высокое быстродействие.

- быстрее, но дороже и сложнее в производстве, используется в качестве кэш-памяти.

21. Память с произвольным доступом (RAM).

Информация в НЖМД записывается на жесткие и алюминиевые

или стеклянные пластины, покрытые слоем ферромагнитного материала, чаще всего диоксида хрома.

С целью адресации пространства поверхности пластин диска делятся на дорожки (track)—концентрические кольцевые области. Каждая дорожка делится на равные отрезки — секторы (sector).

Цилиндр (cylinder)— совокупность дорожек, равноотстоящих от центра, на всех рабочих поверхностях пластин (platter) жесткого диска. Номер головки

(head) задает используемую рабочую поверхность (то есть конкретную дорожку из цилиндра).

22. SSD.Преимущества и недостатки.

- **Преимущества:** высокая скорость чтения и записи, низкое энергопотребление и как результат меньшее нагревание системы, отсутствие движущихся частей, что делает SSD более устойчивыми к механическим повреждениям, не требует дефрагментации, полное отсутствие шума.
- **Недостатки:** высокая стоимость, ограниченный ресурс записи, риск потери данных при внезапном отключении питания.

23. Контроллеры.

это электронное устройство, предназначенное для подключения к магистральной компьютера разных по принципу действия, интерфейсу и конструктивному исполнению периферийных устройств. **Контроллеры** служат для уменьшения нагрузки на центральный процессор и повышают общую производительность системы. Значение контроллеров состоит в том, что они освобождают процессор от наиболее медленных функций ввода/вывода информации. Операционная система практически всегда имеет дело с контроллером, а не с самим устройством.

24. Разновидности видеокарты. Их преимущества и недостатки:

- **Интегрированные видеокарты:** блок обработки графики не имеет своего процессора и использует общую оперативную память и процессор ПК, что снижает производительность, но они дешевле и потребляют меньше энергии, что повышает общий срок работы батареи.
- **Дискретные видеокарты:** имеют собственный графический процессор и память, разгружается центральный процессор, обеспечивая высокую производительность всей системы, но требуют мощные блоки питания, больше энергии и дополнительного охлаждения.

25. Мониторы жидкокристаллические и плазменные.

1. Жидкокристаллический монитор (LCD):

- Состоит из двух стеклянных или пластиковых пластин.
- Между пластинами находятся тонкопленочный транзистор, цветной фильтр и суспензия с кристаллами.
- Кристаллы расположены параллельно, позволяя свету проходить.
- Тонкопленочный транзистор создает электрическое поле, изменяющее положение кристаллов и блокирующее свет.

2. Плазменные мониторы:

- Основаны на плазменной панели с инертными газами (ксенон и неон) в микрокамерах между стеклами.
- Содержат два типа электродов: управляющие и прозрачные сканирующие.
- При зарядке электродов происходит ионизация газа, что вызывает электрический разряд и испускание ультрафиолетовых фотонов.
- Ультрафиолетовые фотоны заставляют светиться фосфорное покрытие микрокамер, создавая видимый свет.

26. Уровни программного обеспечения.

Базовый уровень является низшим уровнем программного обеспечения. Отвечает за взаимодействие с базовыми аппаратными средствами.

Содержит базовую систему ввода-вывода BIOS, записанную в ПЗУ на этапе производства.

Системный уровень - является переходным.

Обеспечивает взаимодействие программ с базовым уровнем и аппаратным обеспечением. Влияет на эксплуатационные показатели вычислительной системы.

Другой класс программ системного уровня отвечает за взаимодействие с пользователем.

Служебный уровень - программы этого уровня взаимодействуют как с программами базового уровня, так и с программами системного уровня.

Назначение служебных программ (утилит) состоит в автоматизации работ по проверке и настройке компьютерной системы, а также для улучшения функций системных программ. Прикладной уровень - состоит из прикладных программ для выполнения конкретных задач. Тесная взаимосвязь с системным программным обеспечением.

27. Операционная система: основные принципы, эксплуатационные требования и функции современных ОС.

Операционная система - комплекс программ, предназначенных для управления ресурсами компьютера и организации взаимодействия с пользователем.

Типы ОС:

- Desktopные • Мобильные • Серверные

Основные принципы

- - управление ресурсами
- - поддержка программного обеспечения
- - управление файлами и папками Эксплуатационные требования
- - расширяемость
- - переносимость
- - совместимость
- - надежности отказоустойчивость.
- - безопасность (аутентификация, авторизация, аудит)
- - производительность. Функции современных ОС
 - Управление конфигурацией ПК
 - Управление процессами, потоками и заданиями.
 - Управление памятью.
 - Обеспечение информационной безопасностью.
 - Управление подсистемой ввода-вывода.
 - Управление внешней памятью.
 - Управление файловой системой.
 - Поддержка сетей

28. Основные компоненты операционных систем

- ядро операционной системы — основа ОС и связующее звено между аппаратным и программным обеспечением компьютера, выполняет множество сложных задач, обеспечивая работоспособность ОС и управляя ресурсами компьютера.
- модули, выполняющие вспомогательные функции ОС Состав ядра
 - Менеджер памяти.
 - Планировщик задач.
 - Драйверы устройств. Драйверы устройств — это специальные программы, которые позволяют ОС взаимодействовать с устройствами компьютера.
 - Системные вызовы.
 - Модуль безопасности.
 - Управление процессами и потоками.
 - Управление файловой системой. Вспомогательные модули
- утилиты – программы, решающие отдельные задачи управления и сопровождения вычислительной системы
 - системные обрабатывающие программы
 - программы предоставления пользователю дополнительных услуг

- библиотеки процедур и функций различного назначения, облегчающие разработку пользовательских приложений

29. Файловая система

Файловая система – это средство для организации хранения файлов на носителях данных, таких как жесткие диски, SSD-накопители, флеш-диски и др.

Она управляет созданием, удалением, чтением и записью файлов, а также предоставляет структуру для организации файлов.

Связывает носитель информации с API для доступа к файлам.

Файловая система организуется на уровне **разделов** или **томов** — областей устройства для хранения файловой системы.

Процесс создания файловой системы называется **форматированием**, в ходе которого определяются правила хранения информации и структура каталогов.

Каталоги: используются для организации файлов в иерархическую структуру, позволяя группировать файлы по критериям.

Путь состоит из имени тома, имен каталогов, имени файла, разделенных специальными символами.

Пример: C:\Учебные материалы\БД\cars.txt

Имя тома Имя каталога Имя каталога Имя файла

30. Прикладной уровень ПО.

Прикладное программное обеспечение: комплекс программ для выполнения конкретных задач, доступность которых зависит от операционной системы и взаимодействия человек-программа-оборудование.

1. Текстовые редакторы.

Основные функции - это ввод и редактирование текстовых данных.

Пример: Блокнот (Notepad)

2. Текстовые процессоры.

Функции: форматирование текста, взаимодействие с графикой и таблицами, автоматизация редактирования.

Примеры: Microsoft Office Word, LibreOffice Writer, Google Docs 3. Электронные таблицы.

Предоставляют средства для хранения и обработки данных; автоматическое обновление ячеек при изменении формул. Примеры: Microsoft Office Excel, LibreOffice Calc, Google Sheets.

4. •

• •

Графические редакторы растровые редакторы

(Adobe Photoshop, GIMP); векторные редакторы

(CorelDraw, Inkscape);

3-D редакторы (трехмерная графика)

(FreeCad, Autodesk AutoCad)

Системы управления базами данных (СУБД).

5.

Организация больших массивов данных в табличные структуры.

Основные функции СУБД:

•

•

•

•

6.

•

работ. Применяются в машиностроении, приборостроении,

создание базы; заполнение данными; доступ, поиск и фильтрация.

Пример: PostgreSQL; Microsoft Office Access

Системы автоматизированного проектирования (CAD-системы).

Предназначены для автоматизации проектно-конструкторских

архитектуре.

Пример: Компас 3D; SolidWorks; nanoCAD

7. Браузеры (средства просмотра Web-документов)

Программные средства предназначены для просмотра электронных документов, созданных в формате HTML.

Пример: Google Chrome; Mozilla Firefox; Microsoft Edge.

31. Технологии виртуализации и контейнеризации.

Технологии виртуализации

Виртуализация возникла в 60-х годах как способ расширения оперативной памяти. Разработана специалистами компаний, таких как GE, Bell Labs и IBM.

Виды виртуализации: серверная, сетевая, десктоп-виртуализация, виртуализация памяти и приложений. Наиболее активно развивается серверная виртуализация.

Виртуализация — это технология, с помощью которой можно создавать виртуальные версии физических ресурсов.

Главной составляющей технологии виртуализации является виртуальная машина (ВМ).

ВМ — это изолированная программная среда, эмулирующая аппаратное обеспечение. Она позволяет превращать один физический сервер в несколько виртуальных.

Гипервизор: Это программное решение, которое управляет несколькими ВМ, позволяя им работать параллельно и эффективно делить ресурсы сервера.

Преимущества виртуализации:

- **Изоляция:** Сбой в одной ВМ не влияет на другие.
- **Гибкость:** Быстрый перенос ВМ между серверами
- **Управляемость:** Оптимальное распределение ресурсов.
- **Оптимизация:** Эффективное использование ресурсов

физического сервера.

- **Обслуживание:** Централизованное обновление ВМ.

Недостатки виртуализации:

- **Высокие издержки:** Необходимость в мощном и дорогом оборудовании.
- **Сложность управления**
- **Задержки при миграции**
- **Сниженная производительность** • **Зависимость от гипервизора**

Технологии контейнеризации

Контейнеризация — это технология изоляции приложений и их зависимостей путём упаковки в единое исполняемое окружение — контейнер.

Первые контейнеры появились в 2004 году в Solaris 10.

В 2006 году Google разработала систему ProcessContainers, позже переименованную в control groups и включенную в ядро Linux.

В 2013 году был запущен проект Docker, который стал основоположником экосистемы для управления контейнерами. Docker позволяет упаковать код и зависимости приложения для быстрого запуска.

Kubernetes управляет большим количеством контейнеров и поддерживает сложные приложения.

Контейнеры служат связующим звеном между системным ядром и приложениями, разделяя ресурсы, но обеспечивая изоляцию без необходимости эмуляции ОС. **Преимущества контейнеризации**

- **Легковесность:** меньше ресурсов, быстрое развертывание.

Изоляция: повышенная безопасность и стабильность

- **Переносимость:** легкость в перемещении между средами.

Масштабируемость

- **Управляемость:** легкость в создании и удалении контейнеров.

Недостатки:

- **Ограниченность:** необходимость использования одной ОС

- **Меньшая изоляция:** контейнеры менее изолированы, чем виртуальные машины.

- **Уязвимость** общей ОС.

- **Сложность** настройки и управления.

32. Компьютерные сети. Классификация по размеру охваченной территории.

Компьютерная сеть — совокупность компьютеров, соединенных с помощью каналов связи и средств коммутации в единую систему для обмена сообщениями и доступа пользователей к программным, техническим, информационным и организационным ресурсам сети.

1. **BAN (Body Area Network):**

- Набор устройств, встроенных или закрепленных на теле человека.

Малые размеры и низкое энергопотребление.

- Датчики передают информацию на смартфоны для мониторинга здоровья пациента.

- Радиус действия 1-2 метра.

2. **PAN (Personal Area Network):**

- Сети для взаимодействия устройств одного владельца (например, ПК с мышкой).

- Используют технологии Wi-Fi или Bluetooth.

3. **LAN (Local Area Network):**

- Локальные сети с радиусом действия до 1 километра.

- Включают ПК, смартфоны, принтеры и промежуточные узлы (коммутаторы, роутеры).

- Основные функции: оптимизация работы, возможность общения,

удаленное администрирование, экономия и безопасность обмена информацией.

4. CAN (Campus Area Network):

- Объединяет несколько локальных сетей (например, в учебном заведении).
- Радиус действия от 1 до 5 километров.

5. MAN (Metropolitan Area Network):

- Сети масштаба города, управляемые провайдерами.
- Радиус действия 10-15 километров.

6. WAN (Wide Area Network):

- Глобальные сети, обычно не принадлежащие отдельным лицам или компаниям.
- Пример — Интернет, включающий сети провайдеров и крупных компаний.

33. Компьютерные сети. Классификация по типу функционального назначения.

1. Точка-точка:

- Простейший вид сети, соединяющий два компьютера напрямую через коммуникативное оборудование.
- Достоинства: простота и дешевизна.
- Недостаток: ограничение на соединение только двух компьютеров.

- Используется для быстрой передачи информации между двумя компьютерами

2. Технология "Клиент-сервер":

- Архитектура, где устройства выступают как клиенты (запрашивающие) или серверы (отвечающие на запросы).
- Клиентом обычно является ПК, сервером — машина, предоставляющая ресурсы.
- Оба термина могут относиться как к физическим устройствам, так и к программному обеспечению.

3. Сеть с выделенным сервером:

- Локальная сеть с централизованным управлением через один или несколько серверов.

- Рабочие станции (клиенты) обращаются к ресурсам сети через серверы.

4. Одноранговая сеть:

- Сети, основанные на равноправии участников, без выделенных

серверов.

- Каждый узел является и клиентом, и сервером.
- Позволяет сохранять работоспособность сети при любом

количестве узлов.

34. Компьютерные сети. Классификация по типу сетевой технологии.

Топология "Шина":

Описание: Общий кабель, к которому подключены все рабочие станции.

Достоинства:

- Быстрое время установки.
- Низкая стоимость (меньшая длина кабеля и меньше устройств).
- Простота настройки.
- Неисправность одной станции не влияет на всю сеть.

Недостатки:

- Неполадки (обрыв кабеля) блокируют всю сеть.
- Сложности в выявлении неисправностей.
- Падение производительности при добавлении новых станций.

Топология "Звезда":

Описание: Все компьютеры подключены к центральному узлу.

Достоинства:

- Неисправность одной станции не влияет на сеть.
- Высокая производительность.
- Низкая стоимость.

Недостатки:

- Выход из строя центрального узла блокирует сеть.
- Требуется больше кабеля для прокладки.
- Ограниченное число рабочих станций в зависимости от портов

узла.

Топология "Кольцо":

Описание: Каждый компьютер соединен с двумя другими, информация передается по кругу.

Достоинства:

- Простота установки.
- Минимальное количество дополнительного оборудования.
- Устойчивость к нагрузкам без значительного падения скорости.

Недостатки:

- Неисправность одной станции влияет на всю сеть.
- Сложности в конфигурировании и настройке.
- Добавление/удаление станции требует остановки сети.

Топология "Ячеистая":

Описание: Рабочие станции соединены друг с другом и могут выполнять роль коммутатора.

Достоинства:

- Высокая отказоустойчивость благодаря множеству связей.

Недостатки:

- Сложная настройка.

Топология "Дерево":

Описание: Узлы более высокого уровня связаны с узлами более низкого уровня, образуя комбинацию звезд.

Достоинства:

- Легкость в увеличении и контроле сети (поиск неисправностей)

Недостатки:

- Выход из строя родительского узла блокирует дочерние узлы и может затруднить доступ к сети.

35. Устройство сети Интернет. Протоколы TCP и IP. Адресация в сети.

Интернет – комплекс локальных сетей и автономных компьютеров, соединенных средствами связи и программным обеспечением, работающим на основе протоколов TCP и IP.

В Интернет существует единая система адресации, которая помогает компьютерам найти друг друга в процессе обмена информацией.

Интернет работает по протоколу TCP/IP, который отвечает за физическую пересылку сообщений между компьютерами в сети Интернет.

Протокол – это правило передачи информации в Сети.

- **TCP/IP:** Протоколы, отвечающие за передачу данных между

компьютерами в сети.

IP-протокол: Набор правил для доставки данных с использованием

уникальных IP-адресов отправителя и получателя. Программные модули протокола IP

- **IPv4:** Позволяет более четырехмиллиардов уникальных IP-адресов (формат: четыре десятичных числа от 1 до 255)
- **IPv6:** Расширяет длину IP-адреса до 128 бит, увеличивая число доступных идентификаторов практически до бесконечности.

ТСР: Обеспечивает пересылку больших объемов информации, разбивая данные на части, нумеруя их и оборачивая в ТСР-конверт, который помещается в IP-конверт для передачи по сети.

36. Устройство сети Интернет. Маршрутизаторы, шлюзы.

• **Опорная сеть Интернета:** Состоит из узловых компьютеров (серверов или хостов) и каналов связи между ними. Маршрутизаторы: Установлены на каждом узле, определяют направление передачи ТСР-пакетов по IP-адресу.

• **Функции маршрутизаторов:**

- Сканируют соседние серверы и общаются с

их маршрутизаторами.

- Учитывают состояние соседей для оптимальной переправки

пакетов.

Шлюзы: Подключают локальные сети, работающие на других протоколах, к Интернету, преобразуя данные между форматами локальной сети и Интернетом.

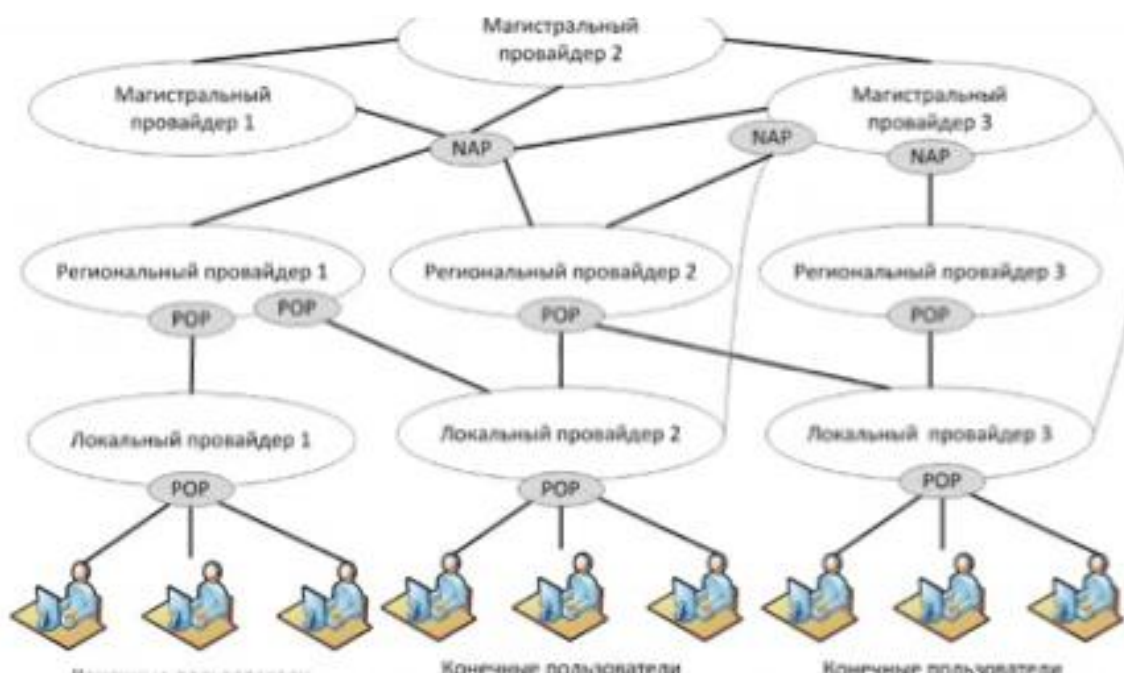
37. Устройство сети Интернет. Провайдеры. Система доменных имен.

ПРОВАЙДЕРЫ

Пользователи подключаются к сети благодаря провайдерам – организациям, оказывающим услуги доступа в Интернет и другие услуги, связанные с Интернетом, например выделение дискового пространства для хранения и обеспечения работы сайтов (хостинг); поддержка работы почтовых ящиков или виртуального почтового сервера; содержание линий связи, то есть поддержание их в рабочем состоянии, и другие.

Магистральные провайдеры имеют магистральные каналы связи в собственности, а региональные арендуют у них каналы связи.

Взаимоотношения между провайдерами осуществляются на основе пиринговых соглашений.



СИСТЕМА ДОМЕННЫХ ИМЕН

Чтобы не запоминать числовые адреса была введена система доменных имен Domain Name System (DNS). Она обеспечивает соответствие числовому IP-адресу каждого компьютера уникального доменного имени, которое обычно

состоит из двух-четырех слов, разделенных точками (доменов). Доменное имя читается слева направо. Самое правое слово в доменном имени является доменом верхнего, или первого, уровня. Существует два типа доменов верхнего уровня: географические (двухбуквенные – указывают на страну, в которой находится узел) и административные (трехбуквенные) – указывают на тип или профиль организации. Каждой стране мира выделен свой географический домен.

Таблицы соответствия DNS-адресов IP-адресам размещают на специальных DNS-серверах, подключаемых к Интернету. Если устройство не знает IP-адреса компьютера, с которым собирается установить связь, а имеет только символьный, то оно запрашивает DNS-сервер, предоставляя ему текстовый вариант, и получает в ответ IP-адрес нужного адресата.

5.255.255.60 - IP-адрес

yandex.ru - символьный DNS-адрес

38. Устройство сети Интернет. Сервисы сети.

ВСЕМИРНАЯ ПАУТИНА:

Всемирная Паутина – это распределенная система, предоставляющая доступ к связанным между собой документам, расположенным на различных компьютерах, подключенных к Интернету.

Всемирная паутина использует технологию гипертекста, в которой документы связаны между собой гиперссылками. Документы, содержащие гиперссылки, называются веб-страницами, а серверы Интернета, их хранящие, – веб-серверами.

Передача веб-страниц по сети Интернет осуществляется с помощью протокола пересылки гипертекста Hypertext Transfer Protocol

(HTTP).Посредством HTTP можно передавать любую информацию, в том числеизображения, звук, видео.

Всемирная паутина работает по принципу клиент-сервер. Веб-сервер принимает HTTP-запросы от клиентов, которыми обычно являются веб-браузеры, и выдает HTTP-ответы.

ЭЛЕКТРОННАЯ ПОЧТА:

Электронное письмо может содержать не только текстовое сообщение, но

и вложенные файлы(программы, графику, звук и т. д.). Кроме того, электронная почта позволяет посылать сообщение сразу нескольким абонентам, пересылать письма на другиеадреса и т. д.

Перваячасть электронного почтового адреса имеет произвольный характер и задаетсясамим пользователем при регистрации почтового ящика. Вторая часть является именем почтового сервера Интернета, на котором пользователь зарегистрировал свой почтовый ящик. Части адреса разделяются символом @.

Процесс передачи сообщения начинается с доставки сообщения в пользовательский почтовый ящик на удаленном почтовом сервере. Почтовый сервер сразу же отправит это сообщение через систему почтовых серверов Интернета на почтовый сервер получателя в его почтовый ящик.

ПОТОКОВОЕ МЕДИА:

Потоковоемедиа представляет собой Интернет-контент (аудио-, видео- или аудиовидеофайлы),который пользователь может смотреть или слушать как непрерывный поток в режимреального времени, не дожидаясь окончания загрузки всего файла на персональныйкомпьютер.

Потоковое медиа пересылается непрерывнымпотокком в виде последовательности сжатых пакетов и проигрывается по мере того,как передается на компьютер получателя.

Еслимедиа не потоковое, то посмотреть файл можно только после его полной загрузки на жесткий диск.

СЕРВИС ПЕРЕДАЧИ ФАЙЛОВ(FTP):

FTP-сайт- это компьютер в сети Интернет, на котором ведется файловый архив. FTP-серверэто программа, работающая на таком компьютере, и которая обеспечивает ведениеархива и обработку запросов к архиву. FTP-клиенты – это программы, используемыедля доступа к FTP-сайтам и открывающие папки на них (FTP-папки) какпростые папки на компьютере пользователя.

Благодарясервису FTP пользователи могут пересылать (копировать, передавать) файлы вИнтернете с удаленного компьютера на локальный и с

локальногона удаленный. В отличие от веб-серверов, которые предоставляют информациютолько для чтения, FTP-серверы позволяют пользователям не только скачиватьинформацию, но и добавлять ее на сервер.

ПОИСКОВЫЕ СИСТЕМЫ:

Поисковые системы решают задачу поиска по большим объемам неструктурированной информации.

В Интернете находится несколько тысяч поисковых систем. Это программно-аппаратные комплексы, предназначенные для осуществления поиска в сети и реагирующие на запрос пользователя выдачей списка ссылок на источники информации в порядке релевантности, сервис, который помогает пользователям быстро найти необходимые сведения. Поисковая система обычно имеет поле для ввода ключевых слов, по которым она находит документы, содержащие эти ключевые слова. Существует 2 основных типа поисковых систем: индексные и классификационные (каталоговые).

Индексные поисковые системы работая в автоматическом режиме обновления своей информации, просматривают в Интернет содержимое серверов, индексируя информацию, содержащуюся в них и внося информацию о расположении слов на страницах сайтов в свои базы данных. Каталогические системы поиска содержат тематически структурированный каталог серверов, и чаще всего пополняются вручную.

ВЭБ - ФОРУМЫ:

Веб-форум — сервис для общения между пользователями Интернета (более двух участников) на одну тему или на несколько тем (зависит от специализации форума).

Суть работы форума заключается в создании пользователями (посетителями форума) своих Тем с их последующим обсуждением, путём размещения сообщений внутри этих тем. Форумы могут существовать как автономно, без привязки к какому-либо сайту, так и быть частью веб-порталов. Форум отличается от чата разделением обсуждаемых тем и возможностью

общения не в реальном времени

СЕРВИС IMS (служба мгновенных сообщений):

С помощью IMS кроме текстовых сообщений можно передавать звуковые сигналы, картинки, видео, файлы, а также, например, производить такие действия как совместное рисование или игры. Для этого необходима клиентская программа, так называемый мессенджер. Ключевой особенностью многих клиентов обмена мгновенными сообщениями является возможность видеть, находится ли друг или коллега в сети и подключается ли он через выбранную службу – эта возможность известна как присутствие. Чат-боты — это вариант обмена мгновенными сообщениями. Они имитируют взаимодействие с пользователем при обмене мгновенными сообщениями, но вместо того, чтобы разговаривать с другим человеком, пользователь разговаривает с автоматизированной программой. Чат-боты используются для автоматизации простых запросов обслуживания клиентов. TELNET: Это сервис, который позволяет осуществлять удаленный доступ в другую вычислительную систему. Информация вводится на одном компьютере, передается на обработку другому, а результаты возвращаются на первый.

Telnet позволяет работать так, будто клавиатура одного компьютера подключена непосредственно к другому, то есть дает возможность пользоваться всеми средствами, которые удаленный компьютер предоставляет локальным терминалам, входить в систему, выполнять команды или получать доступ к множеству специальных сервисных средств.

39. Основные функции СУБД. Функции современных БД

ОСНОВНЫЕ ФУНКЦИИ СУБД.

Определение данных - определить, какая именно информация будет храниться в базе данных, задать свойства данных, их тип (например, число цифр или символов), а также указать, как эти данные связаны между собой.

Обработка данных с обязательным условием обеспечения целостности БД (база данных содержит полную и непротиворечивую информацию)

Управление данными - определение правил индивидуального и

коллективного доступа.

ФУНКЦИИ СОВРЕМЕННЫХ БД • описание данных, их структуры;

- первичный ввод информации; • удаление устаревшей информации;
- корректировка данных; • упорядочение (сортировка) данных;
- поиск информации по некоторым признакам; • подготовка и генерация отчетов; • защита информации;

- резервное хранение и восстановление БД;

- дружественный пользовательский интерфейс.

40. Иерархические и сетевые БД.

ИЕРАРХИЧЕСКИЕ БД

В иерархических базах данных каждая запись имеет одного «родителя». Это создаёт древовидную структуру, в которой записи классифицируются по их отношениям с цепочкой родительских записей.

СЕТЕВЫЕ БД

Записи могут иметь более одного родителя. А значит, можно моделировать сложные отношения.



41. NoSQL базы данных

NOSQL

База данных **NoSQL**, или нереляционная база данных, дает возможность хранить и обрабатывать неструктурированные или слабо структурированные данные

Документно-ориентированные базы данных (NoSQL)

Единицей хранения является документ (который может быть в формате json, или xml, или в каком-нибудь еще формате). Удобство таких баз в том, что в них быстро и легко записывать любые типы данных, при этом эти данные не обязаны обладать четкой структурой. Минус таких баз в том, что данные в них неудобно анализировать.

Пример: MongoDB

Key-value базы данных (NoSQL)

В базах данных «ключ-значение» для хранения информации вы предоставляете ключ и объект данных, который нужно сохранить.

Пример: **Redis - The Real-time Data Platform**

42. Логическая модель БД, ее основные компоненты. Первичные и внешние ключи.

Логическая модель базы данных — это абстрактное представление структуры данных, которое используется для планирования и проектирования баз данных.

Она описывает, как данные будут организованы и как они будут взаимодействовать друг с другом, не привязываясь к конкретной системе управления базами данных (СУБД).

Логическая модель помогает разработчикам и аналитикам понять требования к данным и спроектировать эффективную и масштабируемую базу данных.

ОСНОВНЫЕ КОМПОНЕНТЫ ЛОГИЧЕСКОЙ МОДЕЛИ:

Сущности — это объекты, которые мы хотим хранить в базе данных.

Каждая сущность имеет атрибуты, которые описывают её характеристики.
Например,
сущность "Студент"

атрибуты "Имя", "Фамилия", "Дата рождения", "Номер студенческого билета".
Связи между сущностями описывают как эти сущности будут взаимодействовать друг с другом. Они могут быть трех типов:

- один к одному (1:1)
- один ко многим (1:N)
- многие ко многим (M:N)

Например, связь между сущностями "Группа" и "Студент" может быть "Один ко многим", так как один студент может принадлежать только одной группе, но одна группа может включать многих студентов.

ПЕРВИЧНЫЙ И ВНЕШНИЙ КЛЮЧИ.

Первичный ключ — это уникальный идентификатор сущности.

Внешний ключ — это атрибут, который ссылается на первичный ключ другой сущности, создавая связь между ними.

Например, в сущности "Студент" первичным ключом может быть "Номер студенческого билета", а внешний ключ "Номер группы" будет ссылаться на соответствующий первичный ключ в сущности "Группа".

Первичные и внешние ключи обеспечивают уникальность и целостность данных в базе данных.

Они помогают избежать дублирования данных и обеспечивают правильность связей между сущностями.

4.3. Базовые понятия реляционных баз данных.

ОСНОВНЫЕ ПОНЯТИЯ РЕЛЯЦИОННЫХ БАЗ ДАННЫХ

тип данных домен

атрибут

кортеж первичный ключ отношение

Реляционная модель предусматривает единственный способ представления

данных – в виде набора двумерных таблиц, которые называют отношениями.

Например, таблица хранит информацию об элементах отношения

«студенты». Каждая строка отвечает одной сущности «студент», а каждый столбец отвечает одному из атрибутов множеству сущностей «студенты».

Отношения могут хранить не только множества сущностей, но и экземпляры связей.

ПРИМЕР ОТНОШЕНИЯ «СТУДЕНТ»

Фамилия	Имя	Отчество	Год	№ зачетки
Иванов	Петр	Сергеевич	2006	12345689
Орлова	Елена	Викторовна	2006	12345735
Седов	Олег	Петрович	2005	12345863

Строки – экземпляры сущности
Столбы - атрибуты

Верхней части таблицы отношения задается перечень наименований атрибутов. Атрибуты выполняют функцию заголовков столбцов и содержательно описывают смысл и назначение элементов данных в соответствующих ячейках.

Наименования отношения и атрибутов этого отношения называют схемой отношения. **Схема отношения** представляется в виде имени отношения, за которым идут список атрибутов заключенных в круглые скобки.
Пример: Студенты (Фамилия, Имя, Отчество, Год, Нозачетки)

Атрибуты схемы отношения образуют множество. Полагается, что с каждым **атрибутом** ассоциирован определенный **домен**. т.е. некоторый **базовый тип**. Значения атрибутов должны

принадлежать соответствующим доменам, определяемым каждым из атрибутов отношения.

Строки отношения, отличные от первой, которая представляет наименования атрибутов, называют **кортежами** (tuples). Кортеж содержит по одному компоненту для каждого атрибута отношения.

44. Язык SQL. Создание и заполнение таблиц. Просмотр введенной информации.

СОЗДАНИЕ ТАБЛИЦЫ

В команде **CREATE** указываем **имя таблицы**, **имена заголовков столбцов**, **тип данных в столбце**, **ширину столбцов** и **значения по умолчанию**.

CREATETABLE Products

(

Id SERIAL PRIMARY KEY, ProductName VARCHAR(30) NOT NULL,
Manufacturer VARCHAR(20) NOT NULL, ProductCount INTEGER DEFAULT 0,
Price NUMERIC

);

ЗАПОЛНЕНИЕ ТАБЛИЦЫ

Для занесения в таблицу данных в команде INSERT указываем имя

таблицы, названия заголовков и данные

```
INSERT INTO Products (ProductName, Manufacturer, ProductCount, Price) VALUES  
(  
'iPhone 16', 'Apple', 3, 170000),  
'iPhone 15', 'Apple', 2, 90000),  
'Galaxy 22', 'Samsung', 4, 46000),  
'Galaxy S8 Plus', 'Samsung', 2, 56000),  
'Xiaomi 15', 'Xiaomi', 3, 63000);  
);
```

45. Язык SQL. Фильтрация данных. Операторы IN, BETWEEN, LIKE

Фильтрация данных:

```
SELECT * FROM Products
```

```
WHERE Manufacturer = 'Apple';
```

```
SELECT * FROM Products WHERE Price < 39000;
```

```
SELECT * FROM Products
```

```
WHERE Price * ProductCount > 90000;
```

Операторы IN:

WHERE выражение (NOT) IN (выражение)

Выражение в скобках после IN определяет набор значений. SELECT *

FROM Products

WHERE Manufacturers IN

Оператор BETWEEN:

WHERE выражение [NOT] BETWEEN начальное_значение AND
конечное_значение

```
SELECT * FROM Products
```

```
WHERE Price BETWEEN 20000 AND 50000;
```

Оператор LIKE

WHERE выражение [NOT] LIKE шаблон_строки WHERE ProductName LIKE
'Galaxy%'

```
WHERE ProductName LIKE 'Galaxy _'
```

```
SELECT * FROM Products
```

WHERE ProductName LIKE 'iPhone%';

Вопрос 46.

ORDERBY. Сортировка SELECT * FROM Products

ORDER BY ProductCount;

По умолчанию данные сортируются по возрастанию, однако с помощью оператора **DESC** можно задать сортировку по убыванию
SELECT ProductName, Manufacturer
FROM Products
ORDER BY Manufacturer DESC;

DISTINCT. Выборка уникальных значений

SELECT DISTINCT Manufacturer FROM Products

Вопрос 47.

Получение диапазона строк.LIMIT

Оператор LIMIT позволяет извлечь определенное количество строк:

SELECT * FROM Products
ORDER BY ProductName
LIMIT 4;

Получение диапазона строк.OFFSET

Оператор **OFFSET** позволяет указать, с какой строки надо начинать выборку.

SELECT * FROM Products
ORDER BY ProductName
LIMIT 3 OFFSET 2;

Вопрос 48.

Запросы. Агрегатные функции

AVG: находит среднее значение.

COUNT(*): находит количество строк в запросе **SUM:** находит сумму значений

MIN: находит наименьшее значение

MAX: находит наибольшее значение

SELECT AVG(Price) FROM Products; SELECT MIN(Price) FROM Products;

Изменения названия выходного столбца

```
SELECT ProductCount AS Title, Manufacturer,  
Price * ProductCount AS TotalSum FROM Products;
```

Вопрос 49.

Запросы. Группировка данных

Оператор **group by** определяет, как строки будут группироваться. **SELECT Company, COUNT(*) AS ModelsCount** (**Company** представляет название группы, **Count** вычисляет количество строк в группе)

```
FROM Products GROUP BY Company;
```

Оператор **GROUP BY** может выполнять группировку по множеству столбцов

Группировка по количеству товара

```
SELECT Company, ProductCount, COUNT(*) AS ModelsCount
```

```
FROM Products
```

```
GROUP BY Company, ProductCount;
```

Выражение **GROUP BY** должно идти после выражения **WHERE**, но до выражения **ORDER BY**:

```
SELECT Company, COUNT(*) AS ModelsCount
```

```
FROM Products
```

```
WHERE Price > 30000
```

```
GROUP BY Company
```

```
ORDER BY ModelsCount DESC;
```

Вопрос 50.

Синтаксис. **SELECT** столбцы **FROM** таблица1

```
[INNER] JOIN таблица2 ON условие1
```

```
[ [INNER] JOIN таблица3 ON условие2]
```

```
CREATE TABLE Products (
```

```
Id SERIAL PRIMARY KEY,
```

```
ProductName VARCHAR(30) NOT NULL, Company VARCHAR(20) NOT NULL,
```

```
ProductCount INTEGER DEFAULT 0, Price NUMERIC NOT NULL
```

```
);
```

```

CREATE TABLE Orders (
  Id SERIAL PRIMARY KEY,
  ProductId INTEGER NOT NULL REFERENCES Products(Id) ON DELETE
  CASCADE,
  CreatedAt DATE NOT NULL, ProductCount INTEGER DEFAULT 1, Price
  NUMERIC NOT NULL
);
SELECT Orders.CreatedAt, Orders.ProductCount, ProductName FROM
Products
JOIN Orders ON Products.Id = Orders.ProductId

```

Лекция 12.

Вопрос 51.

Базисные управляющие конструкции

Блок-схемы используются на этапе проектирования и записываются такой степенью детализации, чтобы на следующем этапе - этапе кодирования на выбранном языке программирования - можно было бы каждому блоку однозначно поставить в соответствие нужную последовательность инструкций выбранного языка программирования.

С использованием основных элементов строится набор базисных управляющих конструкций:

последовательность;
 ветвление и выбор
 циклы с предусловием и постусловием

Лекция 13.

Вопрос 52.

Пузырьковая сортировка массива

В пузырьковой сортировке каждый элемент сравнивается со следующим. Если два таких элемента не стоят в нужном порядке, то они меняются между собой местами. В конце каждого прохода наибольший или наименьший элемент ставится в конец списка. **Вопрос 53.**

Сортировка выбором.

Ищем наименьшее значение в массиве и ставим его на позицию, откуда начали проход. Потом двигаемся на следующую позицию. Зеленым отмечается наименьший элемент в подмассиве — он ставится в начало списка.

Вопрос 54.

Сортировка вставками

Начинаем со второй позиции

Число 12 больше 5 — элементы меняются местами

Вопрос 55.

Сортировка Шелла

Алгоритм включает в себя сортировку вставками. Исходный массив размером N разбивается на подмассивы с шагом $N/2$. Подмассивы сортируются вставками. Затем вновь разбиваются, но уже с шагом равным $N/4$. Цикл повторяется. Производим целочисленное деление шага на два каждую итерацию. Когда шаг становится равен 1, массив просто сортируется вставками.

Вопрос 56.

Быстрая сортировка

Массив разделяется на подмассивы, которые сортируются и затем сливаются в один.

В первую очередь выбирается опорный элемент. Опорным может быть любой элемент. Все значения больше опорного элемента ставятся после него, меньше — перед ним.

В полученных массивах также выбираем опорный элемент и разделяем по нему.

Чтобы расположить элементы большие — справа от опорного элемента, а меньшие — слева, двигаемся от начала списка. Если число будет больше опорного, то оно ставится на его место, а сам опорный на место перед ним.

Сложность в лучшем случае: $O(n \cdot \log n)$.

Сложность в худшем случае: $O(n^2)$.

Вопрос 57.

Сортировка кучей Алгоритм сортировки кучей:

Формируем бинарное дерево из массива. Расставляем узлы в дереве так, чтобы получилась куча. Верхний элемент помещаем в конец массива.

Возвращаемся на шаг 2, пока куча не опустеет.

Обращаться к дочерним узлам можно, зная, что дочерние элементы i -го элемента находятся на позициях $2 \cdot i + 1$ (левый узел) и $2 \cdot i + 2$ (правый узел).

Сложность алгоритма в любом случае: $O(n \cdot \log n)$.

Вопрос 58.

Сортировка слиянием

Разделяем исходный массив на два равных подмассива. Повторяем сортировку слиянием для этих двух подмассивов и объединяем обратно.

Цикл деления повторяется, пока не останется по одному элементу в массиве. Затем объединяем, пока не образуем полный список.

Алгоритм сортировки состоит из четырех этапов:

Найти середину массива.

Сортировать массив от начала до середины. Сортировать массив от середины до конца. Объединить массив.

Алгоритм объединения массивов:

Циклично проходим по двум массивам..

В объединяемый ставим тот элемент, что меньше.

Двигаемся дальше, пока не дойдем до конца обоих массивов.

Вопрос 59.

Бинарный поиск

Количество шагов поиска определится как $\log_2 n \uparrow$, где n -

количество элементов, \uparrow — округление в большую сторону до ближайшего целого числа.

На каждом шаге осуществляется поиск середины отрезка по формуле $mid = (left + right) / 2$

Если искомый элемент равен элементу с индексом mid , поиск

завершается.

В случае если искомый элемент меньше элемента с индексом mid , на место $mid - 1$ перемещается правая граница рассматриваемого отрезка, в противном случае — на место $mid + 1$ перемещается левая граница.

ВОПРОС 60 Программирование на Python. Функции ввода-вывода. Для ввода нужной информации используется функция `input()`, которая по умолчанию возвращает в программу введенную пользователем строку.

a, b, c — имена переменных

`=` — оператор присваивания

```
a = input() (5)
```

```
b = input() (6)
```

```
c = a + b
```

```
print(c) (56)
```

Чтобы преобразовать *строку из цифр* в целое число, воспользуемся функцией `int()`

```
a = int (input()) (5)
```

```
b = int (input()) (6)
```

```
d=2
```

```
c = (a + b ) *d (11*2)
```

```
print(c) (22)
```

Для ввода вещественного числа используем функцию `float(input())`

```
a = float (input()) (1.5)
```

```
b = float (input()) (3.05)
```

```
c=a+b (4.55)
```

```
print(c) (4.55)
```

Завывод данных в Python отвечают встроенная функция `print()`.

Спомощью функции вывода `print()` можно вывести на экран любую информацию, например:

```
print('y=',y)
```

Информация в кавычках `y=` выводится на экран в виде текста, `y`

показывает, что будет выведено значение переменной

`y`(информация любого вида в зависимости от типа переменной)

```
print(5 + 10)
```

На экран будет выведена сумма чисел 5 и 10 (15) `print(37 // 3)`

На экран будет выведена частное от деления нацело 37 на 3 (12)

ВОПРОС 61 Программирование на Python. Переменные, имена переменных

Переменная – это именованная область памяти, в которой во время выполнения программы хранятся данные определенного типа

Названия не должны начинаться с цифры, но могут заканчиваться цифрой. Например, назвать переменную `7up` – неправильно, а так – `seven11` – можно.

Названия могут состоять из комбинации строчных, заглавных букв, цифр и символов подчеркивания: `lower_case`, `mixedCase`, `CapitalizedCase`, `UPPER_CASE`, `lower123`.

Не следует давать переменным названия, совпадающие со служебными словами, названиями встроенных функций и методов, к примеру – `print`, `list`, `dict`, `set`, `pass`, `break`, `raise`.

Следует избегать использования отдельных букв, которые могут быть ошибочно приняты друг за друга – l (L в нижнем регистре), I (i в верхнем регистре) или зануль – 0.

В названиях не должно быть пробелов, дефисов и специальных символов, например, ' или \$.

ВОПРОС 62 Программирование на Python. Типы данных.

Преобразование типов.

Типы данных:

- Числовые – целые, вещественные, комплексные числа.

Примечание: для максимально точных расчетов с десятичными числами в Python используют модуль decimal (типа данных Decimal), а для операций с рациональными числами (дробями) – модуль fractions (типа данных Fraction).

- Булевы – логические значения True (истина) и False (ложь).
- Строковые – последовательности символов в кодировке Unicode.
- NoneType – нейтральное пустое значение, аналогичное null в других языках программирования.
 - Последовательности – списки, кортежи, диапазоны.
 - Словари – структура данных типа «ключ: значение».
 - Множества – контейнеры, содержащие уникальные значения. Подразделяются на изменяемые set и неизменяемые frozenset множества.
- Байтовые типы – bytes (байты), bytearray (изменяемая байтовая строка), memoryview (предоставление доступа к внутренним данным объекта).

Преобразование типов:

Округление вещественного числа:

```
a = float(input()) (5.123)
```

```
print(int(a)) (5)
```

Преобразование целого числа в вещественное:

```
a = 5 (5)
```



```
print(float(a)) (5.0)
```

Преобразование строки в число и вывод числа безведущих нулей:

```
a = '00032567'
```

```
print(int(a)) (32567) Сложение строки и числа:
```

```
a = 'Apollo'
```

```
b = 13
```

```
print(a + str(b)) (Apollo 13)
```

ВОПРОС 63 Программирование на Python. Операторы языка
(арифметические, логические, операции сравнения, операции со строками)
Операции со строками?????

Арифметические операторы

```
>>> 1 + 2 3
```

```
>>> 1 - 2 -1
```

```
>>> 1 * 2 2
```

```
>>> 1 / 2 0.5
```

```
>>> 10 // 3 3
```

```
>>> 10 % 3 1
```

(сумма) (разность)

(умножение) (деление)

(деление нацело) (остаток от деления) (возведение в степень)

```
>>> 3 ** 2
```

```
9
```

Логические операторы

or — логическое "ИЛИ";

and — логическое "И";

not — логическое отрицание. Пример: >>> (1 + 1 == 2) or (2 * 2 == 5)

```
True
```

```
>>> (1 + 1 == 2) and (2 * 2 == 5) False
```

```
>>> (1 + 1 == 2) and not (2 * 2 == 5)
```

```
True
```

Операторы сравнения == — равно;

!= — не равно;

> — больше;

< — меньше;

>= — больше или равно; **<=** — меньше или равно. `>>> 1 + 2 == 3`

`True`

ВОПРОС 64 Программирование на Python. Операторы присваивания.

= значение правого операнда присвоится левому операнду;

+= сумма левого и правого операнда присвоится левому операнду;

-= разность левого и правого операнда присвоится левому операнду;

***=** произведение левого и правого операнда присвоится левому операнду;

/= разделит левый операнд на правый и результат присвоится левому операнду;

//= результат целочисленного деления левого операнда на правый операнд присвоится левому операнду;

%= разделит левый операнд на правый по модулю и результат присвоится левому операнду;

****=** возведет левый операнд в степень правого и результат присвоится левому операнду.

Конкретные примеры работы операторов (они скорее всего не нужны) в презентации 15 слайды 8-10

ВОПРОС 65 Программирование на Python. Оператор выбора (if, else)

Синтаксически конструкция выглядит следующим образом:

- сначала записывается часть **if** с условным выражением, которое возвращает истину или ложь;
- затем может следовать одна или несколько **необязательных** частей **elif**;

- завершается запись этого составного оператора также **необязательной** частью **else**.

ВОПРОС 66 Программирование на Python. Цикл while.

Цикл **while** ("пока") позволяет выполнить одну и ту же последовательность действий, пока проверяемое условие истинно. Условие записывается до тела цикла и проверяется до выполнения

тела цикла. Как правило, цикл `while` используется, когда невозможно определить точное значение количества проходов исполнения цикла.

```
i=1
while i<10:

    print(i**2)

    i+=1
```

ВОПРОС 67 Программирование на Python. Модуль `math`. Функции этого модуля.

Встроенный модуль **`math`** в Python предоставляет набор функций для выполнения математических, тригонометрических и логарифмических операций.

Для использования этих функций в начале программы необходимо подключить модуль, что делается командой `import`: `import math`
Некоторые из основных функций модуля:

sqrt(X)
exp(X)
log(X), log2(X), log10(X) log(X, n)
sin(X), cos(X), tan(X) asin(X), acos(X), atan(X) atan2(X, Y)
sinh(X), cosh(X), tanh(X) asinh(X), acosh(X), atanh(X)

Квадратный корень из X

Экспонента числа X

Натуральный, двоичный и десятичный логарифм Логарифм X по основанию n

Синус, косинус и тангенс X, X указывается в радианах Арксинус, арккосинус и арктангенс X

арктангенс отношения X/y с учётом квадранта Гиперболические синус, косинус и тангенс X Обратный гиперболический синус, косинус и тангенс X



pi Выдаётся число π e Выдаётся число e

ВОПРОС 68 Программирование на Python. Применение цикла *for* когда в качестве множества значений используется список.

Цикл for позволяет проводить итерации —реализовывать набор инструкций нужное количество раз.

Его используют, когда количество итераций(повторов)известно заранее.

В цикле for указывается переменная цикла имножество значений, по которому будет пробегать переменная. Множество значенийможет быть задано *списком, кортежем, строкой* или *диапазоном*.

Список (list) — это упорядоченный наборэлементов, каждый из которых имеет свой номер, или индекс, позволяющий быстро получитьк нему доступ. Нумерация элементов в списке начинается с 0.В одном списке одновременно могут лежать данные разных типов —например, и строки, и числа. В один список можно положить другойсписок.

Списки называют динамическими структурами данных,потому что их можно менять на ходу: удалить один или несколькоэлементов, заменить или добавить новые.

Списки записываются с использованием квадратных скобок[].

ВОПРОС 69 Программирование на Python. Применение цикла for когда вкачестве множества значений используется кортеж

Кортежи (тип tuple) — это неизменяемый тип данныхв Python, который используется для хранения упорядоченной последовательностиэлементов.

У этих коллекций есть три свойства:

*Неизменяемость.*После того как кортеж создан, в него нельзя добавлять элементы, а такжеизменять их или удалять.

Упорядоченность. Элементыкортежа располагаются в определённом порядке, который тоже неизменяем. К

Vertical line on the left side of the page.

Vertical line on the right side of the page.





любому элементу можно обратиться по его индексу (порядковому номеру).

Элементами кортежа могут быть объекты разных типов данных: числа, строки, списки, другие кортежи и другие. Элементы-коллекции могут иметь неограниченную глубину вложенности. Например, кортеж может включать в себя список, который будет содержать другой список, который вновь будет содержать список и так далее.

Неизменяемость кортежей не абсолютна. У неё есть исключение — если внутри кортежа находятся изменяемые элементы, например списки, словари или множества, то их значения можно изменить.

Кортежи записываются с использованием круглых скобок ()

Кортежи занимают в памяти меньше места, чем списки **ВОПРОС 70**

Программирование на Python. Применение цикла for когда в качестве множества значений используется диапазон.

Диапазон (тип range) — это специальный тип данных в Python, который позволяет создавать последовательности чисел, не сохраняя их в памяти. Он часто используется в циклах for для выполнения повторяющихся операций. У диапазона есть три ключевых свойства:

1. Ленивое вычисление. Диапазон не создаёт все числа сразу, а генерирует их по мере необходимости, что делает его эффективным с точки зрения памяти.
2. Упорядоченность. Числа в диапазоне всегда идут в заданном порядке.
3. Гибкость. Диапазон можно настроить с учётом начального значения, конечного значения (не включается в последовательность) и шага.

Диапазон создаётся с использованием функции range() и может принимать от одного до трёх аргументов:

- range(stop) — генерирует числа от 0 до stop-1.
- range(start, stop) — генерирует числа от start до stop-1.
- range(start, stop, step) — генерирует числа от start до stop-1 с указанным шагом step.

Пример применения цикла for с диапазоном:

1. Простая итерация по числам:

```
for i in range(5): # Диапазон от 0 до 4
```

```
print(i)
```

```
# Вывод: 0, 1, 2, 3, 4
```

2. Указание начального и конечного значения: for i in range(2,7): # Диапазон от 2 до 6

```
print(i)
```

```
# Вывод: 2, 3, 4, 5, 6
```

3. Использование шага:

```
for i in range(1, 10, 2): # Диапазон от 1 до 9 с шагом 2
```

```
print(i)
```

```
# Вывод: 1, 3, 5, 7, 9
```

4. Обратный порядок:

```
for i in range(10, 0, -2): # Диапазон от 10 до 2 с шагом -2
```

```
print(i)
```

```
# Вывод: 10, 8, 6, 4, 2
```

Преимущества использования диапазона в цикле for:

- Экономия памяти за счёт ленивого вычисления значений.
- Удобство задания параметров последовательности (начало, конец, шаг).
- Возможность работы с большими последовательностями чисел.

Диапазоны применяются во множестве задач, например, для выполнения повторяющихся действий, обработки индексов в списках, построения последовательностей чисел или организации вложенных циклов.

ВОПРОС 71 Программирование на Python. Функция range. Ее аргументы.

Возможные операции с функцией.

Для получения последовательности чисел можно использовать функцию range, которая генерирует последовательность чисел.

итерация по числам с нуля до 10 не включительно

for i in range(0, 10):

print(i) 0

1 2

3

4

5

6

7

8

9

i – переменная цикла

range(0, 10) – последовательность чисел от 0 до 9, не включая 10

У функции 3 параметра:

- start — начало последовательности [включительно] (не обязательный параметр, по умолчанию равен 0).
- stop — задает точку остановки последовательности [значение не включено в последовательность] (обязательный параметр).
- step — шаг последовательности (не обязательный параметр, по умолчанию равен 1).

range(stop) # с одним параметром

range(start, stop, step) # с несколькими параметрами

При работе с функцией range() важно помнить следующее:

- Значение stop не входит в последовательность;
- Все аргументы функции должны быть целыми числами (положительными или отрицательными);
- При отрицательном шаге step нужно помнить, что значение start должно быть больше значения stop;
- Значение step не должно быть равно 0, иначе Python вызовет исключение "ValueError".

ВОПРОС 72 Программирование на Python. Модуль random. Функции random(), randint(), randrange().

Модуль random управляет генерацией случайных чисел. Его основные функции:

- `random()`: генерирует случайное число от 0.0 до 1.0
- `randint()`: возвращает случайное число из определенного

диапазона

- `randrange()`: возвращает случайное число из определенного набора чисел

Функция `randint(min, max)` возвращает случайное целое число в промежутке между двумя значениями `min` и `max`.

Функция `randrange()` возвращает случайное целое число из определенного набора чисел. Она имеет три формы:

- `randrange(stop)`: в качестве набора чисел, из которых происходит извлечение случайного значения, будет использоваться диапазон от 0 до числа `stop`
- `randrange(start, stop)`: набор чисел представляет диапазон от числа `start` до числа `stop`
- `randrange(start, stop, step)`: набор чисел представляет диапазон от числа `start` до числа `stop`, при этом каждое число в диапазоне отличается от предыдущего на шаг `step`