

# Занятие 12

Курс "Основы программирования"

# План занятия

## Вопросы-ответы:

1. Программная архитектура ПК
2. Языки программирования
3. Операции, операторы
4. Переменные, массивы, объекты
5. Функции

# Программная архитектура ПК

1. Каковы две наиболее значимые в работе программ сущности?

# Программная архитектура ПК

- Память
- Процессор

# Программная архитектура ПК

**2. Какие основные виды памяти компьютера нам доступны?**

# Программная архитектура ПК

- **Оперативная память**
- **Постоянная / внешняя память**

# Программная архитектура ПК

## 3. Стек и куча (Stack & Heap).

# Программная архитектура ПК

- **Стек – буфер для временного хранения данных внутри процесса. Память выделяется автоматически по мере необходимости.**
- **Куча – вся доступная оперативная память. ОС выделяет участки (сегменты) по требованию программы.**



# Программная архитектура ПК

4. Для чего предназначен  
процессор компьютера?

# Программная архитектура ПК

Процессор выполняет действия:

- Копирование данных между участками памяти
- Математические операции

# Программная архитектура ПК

**5. Как мы можем управлять  
деятельностью процессора?**

# Программная архитектура ПК

- Процессор производит действия в порядке, предписанном программой.
- Создавая ту или иную программу, мы определяем набор действий, который процессор последовательно произведёт.

# Программная архитектура ПК

**6. Что нужно предпринять для создания программы?**

# Программная архитектура ПК

- Записать текст программы на одном из языков программирования.
- При необходимости, преобразовать её текст в понятный компьютеру код.
- Запустить полученную программу.

# Языки программирования

**1. Какие типы языков бывают?**

# Языки программирования

- Императивные (C, JS, C#, PHP ...)
- Декларативные (HTML, JSON ...)
- Функциональные (Lisp, F#, Haskell ...)
- Логические (Prolog ...)



# Языки программирования

2. В чём разница между декларативными и императивными языками?

# Языки программирования

- **Императивные языки обеспечивают возможность задать список действий, которые, затем, машина неукоснительно выполнит.**
- **Декларативные языки позволяют описать различные сущности и их взаимосвязи, независимо от действий, которые машина предпримет для их обеспечения.**

# Языки программирования

3. В чём разница между компилятором и интерпретатором?

# Языки программирования

- Компилятор лишь конвертирует предоставленные вами тексты в вид понятный «исполнителю», не исполняя её.
- Интерпретатор же непосредственно занимается исполнением команд, причём на вход ему подаётся изначальный текст программы, а не скомпилированный.

# Языки программирования

4. Для чего применяется язык С?

# Языки программирования

**Язык С – быстрый и простой, находит применение в несложных но ресурсоёмких системах.**

**Например, вычисление факториалов, поиск и сортировка в массивах.**

# Языки программирования

5. Для чего применяется язык JS?

# Языки программирования

Программы на JS гораздо медленней программ на C, но они позволяют молниеносно создавать системы высокой сложности с большим количеством разнородных взаимосвязанных компонентов.



# Операции, операторы

1. Каково значение операций в императивных языках программирования?

# Операции, операторы

**Программы на императивных языках  
состоят в основном из операций.**

**В том числе, как исполняемых  
процессором, так и нет.**

# Операции, операторы

**2. Перечислите основные типы операций.**

# Операции, операторы

- Присвоение
- Арифметика
- Логика
- Доступ
- Управление программой
- Управление средой

# Операции, операторы

**3. Какие операторы формирования выражений вы знаете?**

# Операции, операторы

+ - \* / % ++ --

|| && !

| & ^ ~

> < >= <= == !=

+= -= \*= /= %= |= &= ~= ^=

( )

# Операции, операторы

4. Перечислите операторы управляющие ходом работы программы.

# Операции, операторы

`if, else, switch`

`for, while, do while`

`break, continue, return`

`()`

`{ }`

`try, catch`



# Переменные, массивы, объекты

## 1. Что такое переменная?

# **Переменные, массивы, объекты**

**Переменная – это программная ячейка памяти.**

**В отличие от аппаратной памяти, ячейки  
представленные переменными могут иметь  
разный размер и тип содержимого, определяемые  
языком или самим программистом.**

# Переменные, массивы, объекты

2. Что может храниться в переменной?

# Переменные, массивы, объекты

Данные любого рода:

- Числа
- Строки
- Другие переменные (объекты, массивы)
- Программный код (функции)
- И т.д.

# Переменные, массивы, объекты

3. Что представляют собой массивы?

# Переменные, массивы, объекты

**Массив – это переменная, внутри которой лежит пронумерованное множество других переменных.**

- **Массив может быть и пустым.**
- **Максимальный размер массива ограничен возможностями компьютера.**

# Переменные, массивы, объекты

4. Для чего нужны массивы?

# Переменные, массивы, объекты

**Массивы принято использовать для хранения наборов однородных данных.**

- Например, список покупок, список учащихся.

**В массивах удобно искать, сортировать, обрабатывать, добавлять, удалять данные.**



# Переменные, массивы, объекты

5. Что представляют собой объекты?

# Переменные, массивы, объекты

**Объект – это переменная, внутри которой множество других именованных переменных.**

**Объект может быть пустым.**

**Объект для внешнего мира представляется единой сущностью, работающей по определённому принципу.**

# Переменные, массивы, объекты

6. Для чего нужны объекты?

# **Переменные, массивы, объекты**

**В виде объектов представляют модели реального или вымышленного мира.**

**В объект включают свойства и задают поведение в соответствии с моделируемой сущностью.**

**Объекты используются когда сущность не удаётся представить в виде простого числа или строки: например она имеет несколько важных характеристик или сложный закон поведения.**

# Функции

## 1. Что такое функция?

# Функции

**Функция – это именованный или иначе обозначенный кусок программного кода, который может быть многократно выполнен.**

**Как правило, функции способны принимать несколько входных параметров перед выполнением, и оставлять один результат после своего выполнения.**

# Функции

2. **Сигнатура функции: возвращаемое значение, принимаемые аргументы, имя.**

# Функции

**Сигнатура – то что отличает одну функцию от другой.**

- **Имя – название функции.**
- **Аргументы – входные данные, их значения задаются при каждой операции вызова.**
- **Возвращаемое значение – результат работы функции.**  
**Обычно устанавливается в процессе завершения работы функции.**



# Функции

## 3. Хранение функции в переменной.

# Функции

**Написав код функции, мы можем:**

- **Хранить его в переменных.**
- **Передавать его между переменными.**
- **Вызывать выполнение этого кода оператором вызова на любой из переменных где он хранится.**

**Ваши вопросы**