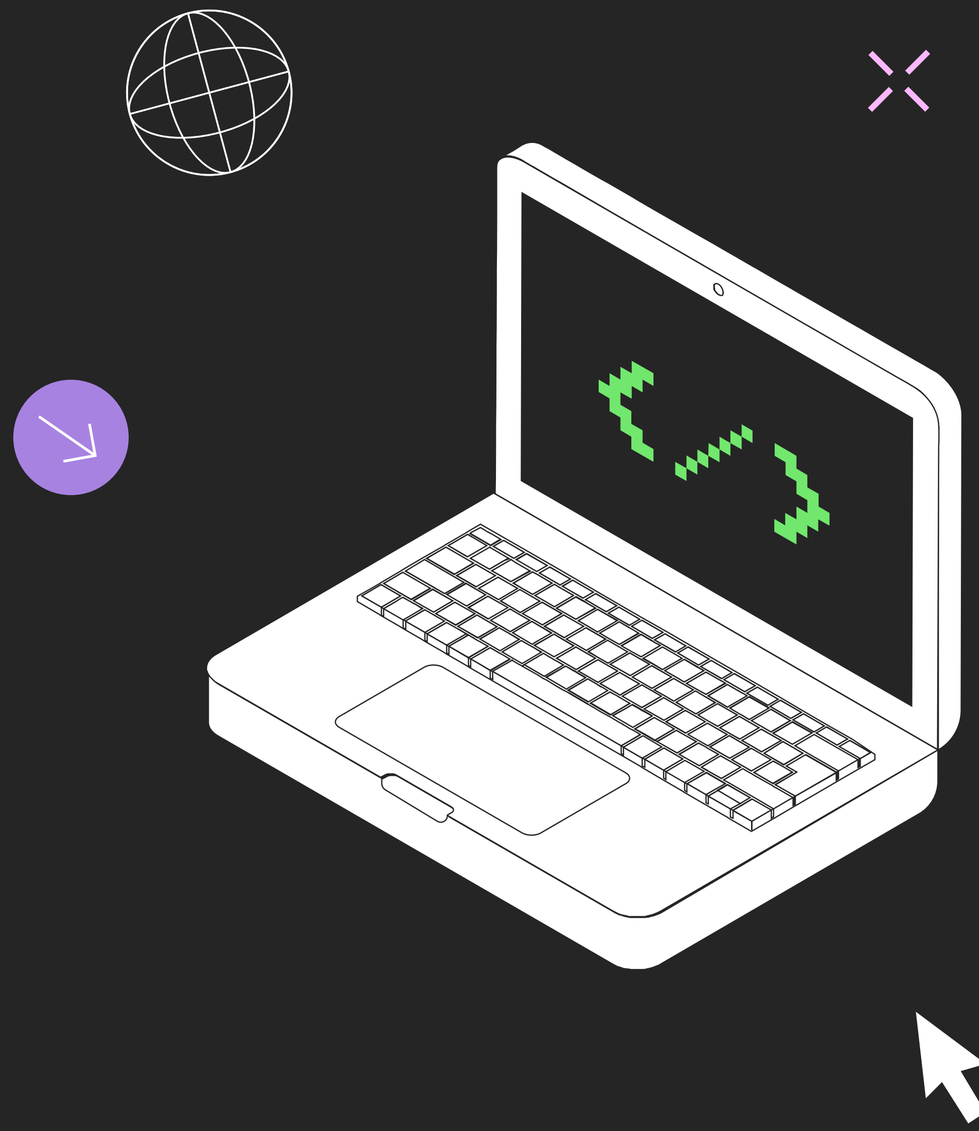
GeekBrains



Введение
в программирование

Языки программирования и подходы к обучению

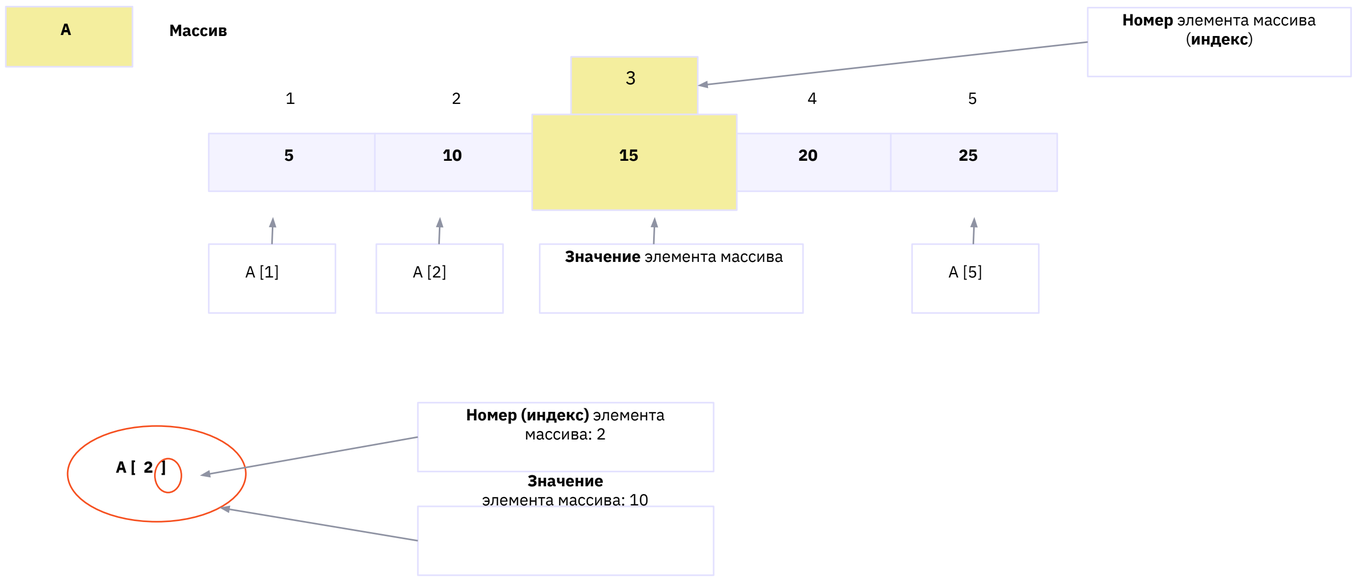
💡 Программирование и язык программирования — это разные вещи.

- У каждого языка программирования есть уникальные свои синтаксические особенности.
- Если учить только один язык, то познания в программировании сузятся только до этого языка.
- Тогда переход с одного языка на другой может быть достаточно сложным.

💡 **Язык программирования** — это средство донесения смысла, а **программирование** — сам смысл (какую задачу выполняют программа)

Массив как структура данных

Массив — это структура данных, хранящая набор значений (элементов массива), идентифицируемых по индексу или набору индексов.



- ✔ все элементы имеют один тип
- ✔ весь массив имеет одно имя
- ✔ все элементы расположены в памяти рядом

Параметры массива:

1. **Начало массива**

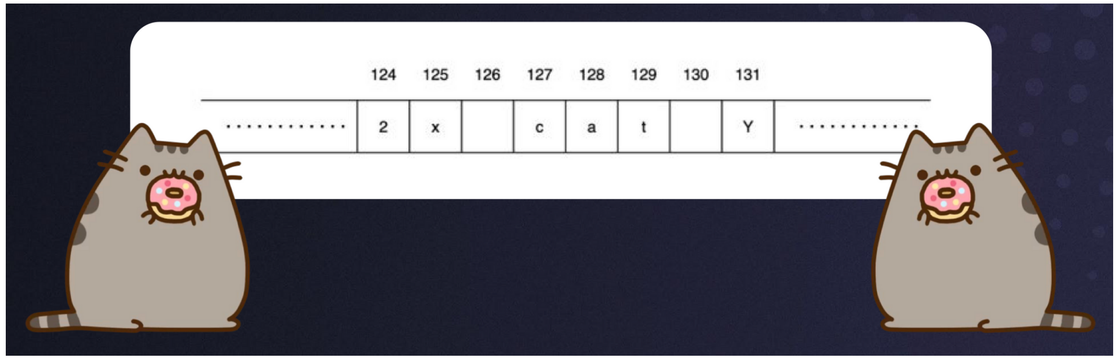
   Адрес первой ячейки с элементами массива.

2. **Размер каждого элемента массива**

   Сколько ячеек памяти занимает каждый элемент массива.

3. **Количество элементов массива**

💡 Память компьютера — это массив



Формула поиска n-ого элемента массива

$a_n = start + (n-1) * cellSize$

a_n — искомый адрес в памяти n-ного элемента,

start — адрес начала массива

n-1 — количество элементов, на которое нам необходимо отступить от начала массива

cellSize — размер одного элемента (сколько ячеек памяти он занимает)

Как компьютер считывает наш код

Языки высокого уровня — есть огромное количество посредников; максимально похожи на человеческую речь

Языки низкого уровня — небольшое количество посредников; команды ближе к командам, которые будет выполнять компьютер

Трансляторы (программы-переводчики) — переводят написанный человеком файл и преобразуют в код для машины.

Трансляторы бывают двух типов:

1. Интерпретаторы
   - Python
   - Выполняет программу построчно
2. Компиляторы
   - Java
   - Сначала переводит всю программу, потом выполняет

Сейчас практически нет чёткого деления среди программ, программы на шкале «Интерпретатор-Компилятор» ближе к одному из полюсов.

Типы ошибок:

- **Синтаксические ошибки**

  Например, неправильно написанные команды, пропущенные знаки или несоблюдение отступов и тд.

- **Ошибки выполнения**

  Например, деление на 0

- **Логические ошибки**

  Когда программа работает, но выводит не то, что от неё требуется. Или работает с одним количеством чисел, а с другим уже не работает.

Наименование переменных

💡 Стили наименования переменных:

- snake case: max_number_index
- CamelCase: MaxNumberIndex

Самое сложное в программировании — именование переменных.

- Значения, которые хранятся в переменных должны быть отражены в названии переменных.
- Читатель должен понимать по названию, что хранится в переменной.

Структуры в программировании

Программу любой сложности можно представить в виде комбинации из трёх структур:

1. Следование

   Последовательное выполнение инструкций.

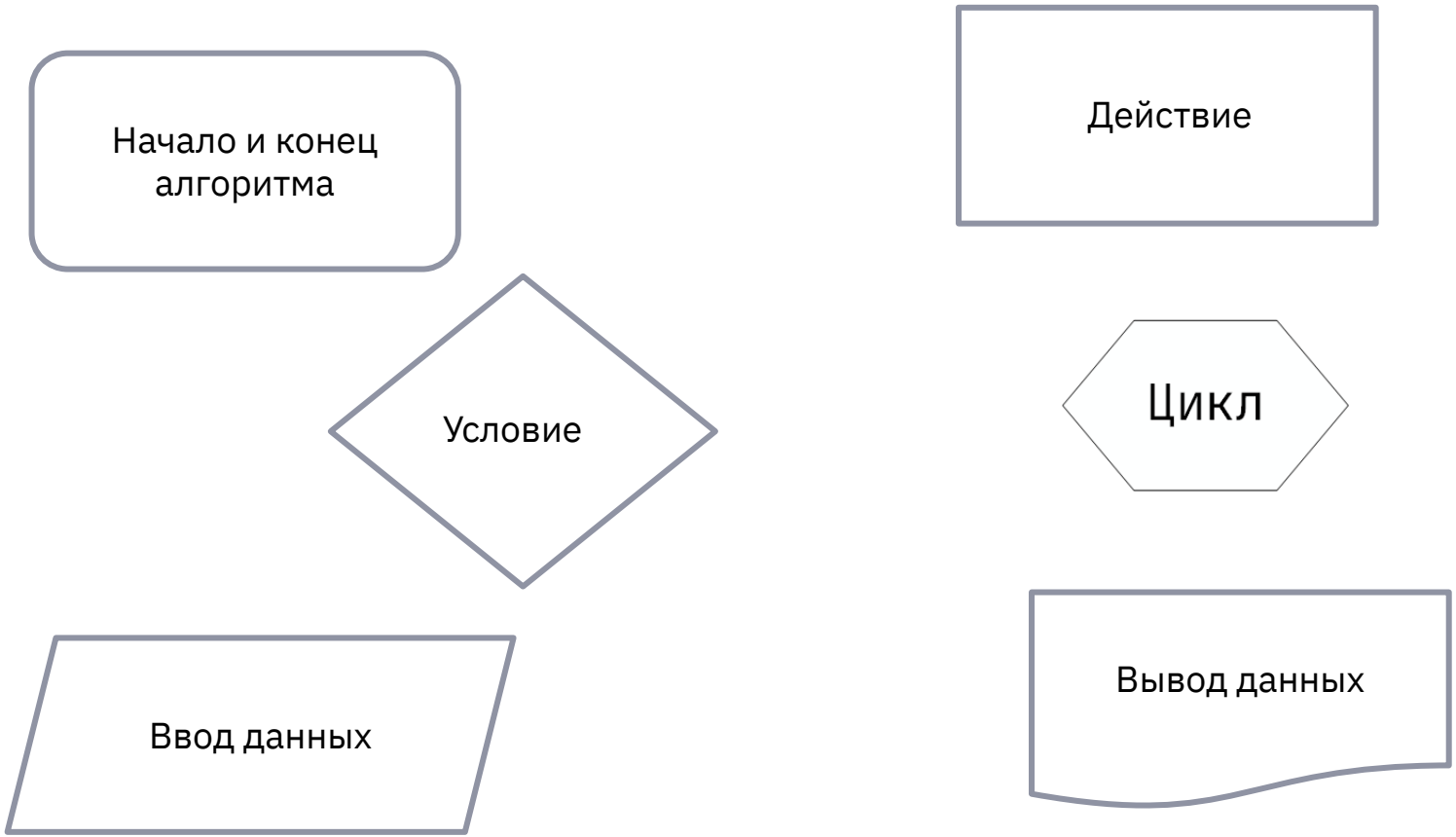
2. Ветвление

   Условия, которые позволяют перенаправить выполнение нашей программы по одной из имеющихся веток.

3. Цикл

   Конструкция, которая позволяет задавать многократное повторение операторов.

Блоки Составления блок-схем

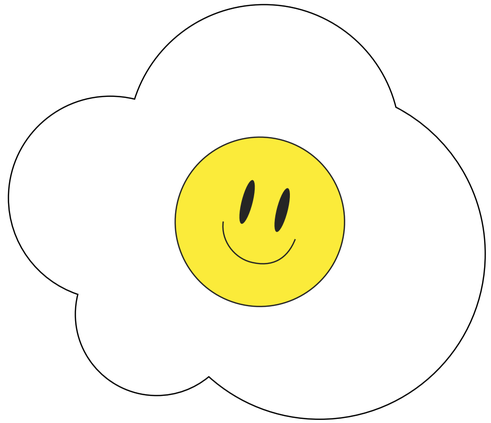


Функция

Бытовое представление функций: подпрограммы

Инструкция готовки яичницы.

1. Необходимо взять сковороду.
2. Поставить сковороду на плиту и включить огонь.
3. Смазать сковороду маслом.
4. Аккуратно разбить несколько яиц и вылить содержимое на горячую, смазанную маслом сковороду.
5. Посолить
6. Накрыть крышкой и следить, чтобы яичница не подгорела.
7. Когда яичница будет готова, выключить плиту.



- Слово «несколько» выделено цветом — функции могут быть с параметром, а могут быть без них.
- Параметр (аргумент) функции — числа, которые влияют на функцию.

Чтобы в реальной жизни пользоваться такими функциями, необходимо дать подробную инструкцию.

- Разбор примера с функцией find_max.
  - В примере ошибка, задача студентам дома найти ошибку.
- Решение задачи с поиском второго по величине значения.

```
function find_max(array):
 size = array.length
 index = 0
 max = array[0]
 while (index < size) do
    if (array[index] > max) then
    max = array[index]
    index = index + 1
 return max
```

```
numbers = [1, 8, 3, 2, 6]
another_numbers = [15, 2, 74, 3, 8, 16, 24]
second_max_number = find_second_max(numbers)
another_second_max_number = find_second_max(another_numbers)
```