



# Модуль 1

Введение в язык программирования Java





# 1. Вступление

- История и этапы развития языка Java.
- Сравнительный анализ языка Java с другими языками программирования.
- Что такое виртуальная машина?
- Что такое байт-код?



## 2. Алгоритм

- Понятие алгоритма.
- Примеры использования алгоритмов в реальной жизни.
- Типы алгоритмов. Линейный, разветвленный, циклический.



### 3. Понятие блок-схемы.

- Базовые обозначения в блок-схемах.
- Блок начала алгоритма.
- Блок завершения алгоритма.
- Блок ввода данных.
- Блок вывода данных.
- Блок вычислений.
- Простейшие примеры использования блок-схем.



## 4. Программная среда IntelliJ IDEA.

- Основы работы с IDE IntelliJ IDEA.
- Создание проекта.
- Добавление файла к проекту.
- Обзор альтернативных средств разработки.
- Запуск простейшего приложения.



## 5. Git

- **Что такое Git?**
- **Цели и задачи Git?**
- **Основные термины:**
  - репозиторий
  - коммит
  - ветка
  - рабочий каталог



## 6. Практическая часть

- **Что такое Unit test**
- **Обзор ресурсов для самостоятельной практики**
  - [codewars.com](https://codewars.com)
  - [codingame.com](https://codingame.com)
  - [exercism.org](https://exercism.org)
- **Практическое задание**



## 1. История и этапы развития языка Java.

Java — строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией Sun Microsystems (<https://ru.wikipedia.org/wiki/Java>).

Дата официального выпуска — 23 мая 1995 года.





## Версии языка

- **JDK 1.0** (21 января 1996 года)
- **JDK 1.1**
- **J2SE 1.2**
- **J2SE 1.3**
- **J2SE 1.4**
- **J2SE 5.0** (30 сентября 2004)
- **Java SE 6**
- **Java SE 7**
- **Java SE 8** (19 марта 2014 года)
- **Java SE 9**
- **Java SE 10**
- **Java SE 11** (25 сентября 2018 года)
- **Java SE 12**
- **Java SE 13**
- **Java SE 14**
- **Java SE 15**
- **Java SE 16**
- **Java SE 17** (14 сентября 2021)




# Java EE

**Java Enterprise Edition** представляет платформу для создания корпоративных приложений на языке Java. Прежде всего это сфера веб-приложений и веб-сервисов.



## Java ME

**Java Micro Edition (Java ME, ранее — Java 2 Micro Edition, J2ME)** — подмножество платформы **Java** для устройств, ограниченных в ресурсах, например: **сотовых телефонов, карманных персональных компьютеров, ресиверов цифрового телевидения, проигрывателей дисков Blu-ray.** ([https://ru.wikipedia.org/wiki/Java\\_Platform,\\_Micro\\_Edition](https://ru.wikipedia.org/wiki/Java_Platform,_Micro_Edition))



## Сравнительный анализ языка Java с другими языками программирования

- Вопрос: Что можно написать на Java?

Ответ: Всё!

- Вопрос: Что можно написать на C++?

Ответ: Java!



## Что такое виртуальная машина

**Java Virtual Machine** (сокращенно **Java VM**, **JVM**) — виртуальная машина Java — основная часть исполняющей системы **Java**, так называемой *Java Runtime Environment* (**JRE**). Виртуальная машина Java исполняет **байт-код Java**, предварительно созданный из **исходного текста** Java-программы **компилятором** Java ([https://ru.wikipedia.org/wiki/Java\\_Virtual\\_Machine](https://ru.wikipedia.org/wiki/Java_Virtual_Machine))



# Что такое байт-код?

```
class Hello {  
    public static void main(String[] args) {  
        System.out.println("Максим, ты уволен!");  
    }  
}
```



```
public class Main {
```

```
// compiled from: Main.java
```

```
// access flags 0x1
```

```
public <init>()V
```

```
L0
```

```
LINENUMBER 1 L0
```

```
ALOAD 0
```

```
INVOKESPECIAL java/lang/Object.<init> ()V
```

```
RETURN
```

```
L1
```

```
LOCALVARIABLE this LMain; L0 L1 0
```

```
MAXSTACK = 1
```

```
MAXLOCALS = 1
```

```
public static main([Ljava/lang/String;)V
```

```
L0
```

```
LINENUMBER 4 L0
```

```
GETSTATIC java/lang/System.out : Ljava/io/PrintStream;
```

```
LDC "Hello world"
```

```
INVOKEVIRTUAL java/io/PrintStream.println (Ljava/lang/String;)V
```

```
L1
```

```
LINENUMBER 5 L1
```

```
RETURN
```

```
L2
```

```
LOCALVARIABLE args [Ljava/lang/String; L0 L2 0
```

```
MAXSTACK = 2
```

```
MAXLOCALS = 1
```



# Что такое компилятор

**Компилятор** — программа, переводящая написанный на языке программирования текст в набор машинных кодов (<https://ru.wikipedia.org/wiki/Компилятор>)





Если компилятор - это программа, то на каком языке  
пишут компиляторы?



Компилятор Java написан на





## Понятие алгоритма

**Алгоритм** (лат. *algorithmi* — от имени среднеазиатского математика *Аль-Хорезми*) — конечная совокупность точно заданных правил решения некоторого класса задач или набор *инструкций*, описывающих порядок действий исполнителя для решения определённой задачи. (<https://ru.wikipedia.org/wiki/Алгоритм>)

## Примеры использования алгоритмов в реальной жизни.





## Типы алгоритмов

- Линейный: Приготовление яичницы:
  - зажечь газ
  - разогреть на нём сковороду
  - разбить 2 яйца
  - содержимое вылить на горячую сковороду
  - посолить
  - накрыть крышкой
  - ждать 5 минут
  - выключить газ
  - выложить яичницу на тарелку



# Разветвленный алгоритм

## Покупки

- пойти в магазин
- взять хлеб
- если есть пирожки, взять два
  - иначе один
- заплатить за покупки
- вернуться домой



## Циклический

### Посадка картофеля

1. Вскопать ряд грунта
2. Удобрить место высадки
3. Положить картофель с интервалом 25см
4. Закопать картофель
5. Перейти к п.1



## Базовые обозначения в блок-схемах

Begin/End

Данный символ, который иногда также именуют «Терминатором», применяется для обозначения начальной или конечной точки схемы или возможного результата того или иного пути развития процесса. Внутри блока, как правило, располагается слово «Начало» или «Конец»

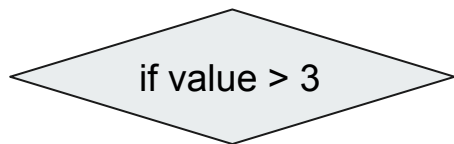
int value = 5

Этот символ, также известный под названием «Действие», используется для обозначения процесса, действия или функции. Это самый распространенный символ в блок-схемах.





## Базовые обозначения в блок-схемах



if value > 3

Символизирует вопрос, на который требуется ответ (как правило, «да/нет» или «истина/ложь»). На этом этапе блок-схема разветвляется в разных направлениях в зависимости от выбранного ответа и последующих блоков.



print(value)

Эта фигура, также известная под названием «Данные», символизирует данные, доступные для ввода или вывода, а также затраченные или полученные ресурсы.

# Блок-схема алгоритма расчета линейной функции

Формула линейной функции:

$$y = kx + b$$

