Java

Первая программа.

Типы данных. Константы

Лекция #1

Пустовалова О.Г.
доцент. каф. мат.мод.
ИММиКН ЮФУ

Содержание

Какое ПО нужно установить

Установить JDK 8 для вашей ОС

http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

- Java Development Kit (сокращенно JDK) — бесплатно распространяемый компанией Oracle Corporation (ранее Sun Microsystems) комплект разработчика приложений на языке Java, включающий в себя компилятор Java (javac), стандартные библиотеки классов Java, примеры, документацию, различные утилиты и исполнительную систему Java (JRE).
- В состав JDK не входит интегрированная среда разработки на Java, поэтому разработчик, использующий только JDK, вынужден использовать внешний текстовый редактор и компилировать свои программы, используя утилиты командной строки.

Какое ПО нужно установить

Установить среду разработки IntelliJ Idea Community Edition
http://www.jetbrains.com/idea/download/

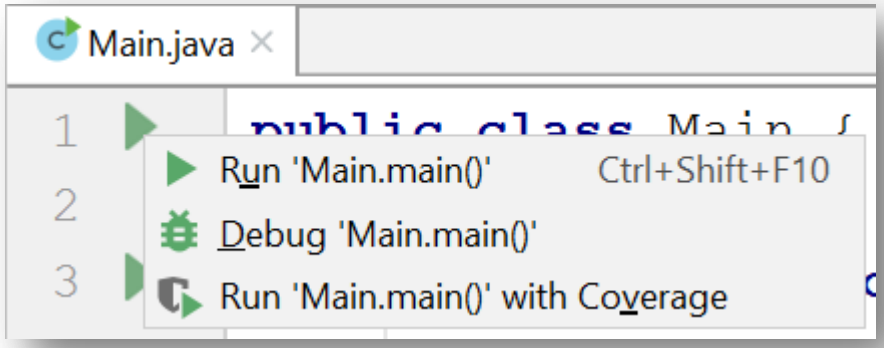
- IntelliJ IDEA — интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python, разработанная компанией JetBrains. IntelliJ IDEA — интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python, разработанная компанией JetBrains.

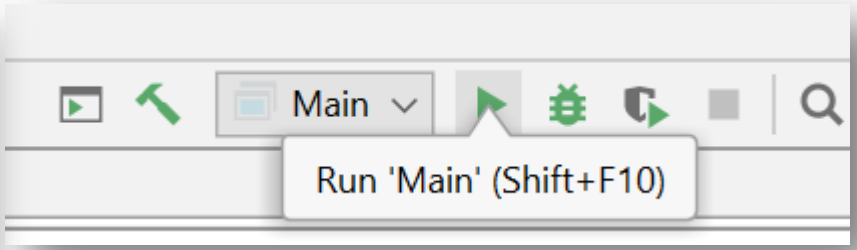
Первая программа

Первая программа

Main.java

```
1  public class Main {
2
3      public static void main(String[] args) {
4
5          System.out.println("Hello!");
6
7      }
8  }
```





Пример 1. First program. Открытость main()

```
public class FirstProgram {
 public static void main(String[] args) {

        System.out.println("We will not use 'Hello, World!'");
    }
}
```

модификатор доступа

Загрузчик Java в версии Java SE 1.4 требует, чтобы метод **main()** был открытым (**public**).

Пример 1. First program. Программа = класс

все элементы программ на Java находятся в составе классов

```
public class FirstProgram {
```

CamelCase

```
 public static void main(String[] args) {

        System.out.println("We will not use 'Hello, World!'");
    }
}
```

в языке Java **учитывается регистр символов**

Пример 1. О статичности main()

```
public class FirstProgram {

    public static void main(String[] args) {

        System.out.println("We will not use 'Hello, World!'");
    }
}
```

Метод **main ()** в Java всегда является **статическим**

Статический метод не принадлежит ни одному из объектов класса

Пример 1. First program. Метод main обязателен

```
public class FirstProgram {

 public static void main(String[] args) {

        System.out.println("We will not use 'Hello, World!'");
    }
}
```

для нормального выполнения программы в классе должен присутствовать метод **main()**

в класс можно добавить и **другие методы** и вызывать их из метода main ()

Пример 1. First program. Вызов другого метода из main()

```
public class FirstProgram {

    public static void anothermethod() {
        System.out.println("***Hello, World!***");
    }

    public static void main(String[] args) {
        System.out.println("We will not use 'Hello, World!'");

        anothermethod();
    }
}
```

```
We will not use 'Hello, World!'
***Hello, World!***
```

Пример 1. First program. System.out

```
public class FirstProgram {



 public static void main(String[] args) {

        System.out.println("We will not use 'Hello, World!'");
    }
}
```

System.***out*** **–** объект

println – его метод

Общий вид приложения на Java

```
public class ИмяКласса {

    public static void main(String[] args) {

        операторы программы;
        объект.метод(параметры);

    }
}
```

в языке Java каждый оператор должен оканчиваться
точкой с запятой ;

Общий вид приложения на Java. Комментарии

```
// Однострочный комментарий
public class ClassName {
    public static void main(String[] args) {
        /*
        Многострочный комментарий.

        Вызов метода без параметров.
        Переход на следующую строку.
         */
        System.out.println();
    }
}
```

Комментарии для документации

```
/**
 * так формировать
 * комментарии для
 * документации
 */
public class Main {
    public static void main(String[] args) {
        System.out.println();
    }
}
```

Code Conventions for the Java Programming Language

http://java.sun.com/docs/codeconv/index.html

- ✓ имена классов начинаются с прописной буквы; если имя содержит несколько слов, то каждое слово начинается с прописной буквы;

```
public class FirstProgram
```

- ✓ имена методов и переменных начинаются со строчной буквы; если имя содержит несколько слов, то каждое следующее слово начинается со строчной буквы;

```
int m, carCount;
```

- ✓ имена констант записываются полностью прописными буквами; если имя состоит из нескольких слов, то между ними ставится знак подчеркивания.

```
System.out.println("PI = "+Math.PI);
```

Java – строго типизированный язык

Строгая типизация Java

Тип каждой переменной должен быть *непременно* объявлен.

В Java имеются
восемь *простых* или *примитивных* **типов данных**.

4 типа — целые числа,

2 типа — действительные числа с плавающей точкой,

1 тип — символы в Юникоде

1 тип — логические значения

Переменные в Java

Переменные в Java

Имя переменной должно начинаться с буквы и представлять собой сочетание букв и цифр.

Термины *буквы* и *цифры* в Java имеют более широкое значение, чем в большинстве других языков программирования.

Буквами считаются символы **A - Z**, **a - z**, **_** (символ подчёркивания), и любой другой символ кодировки в Юникоде, соответствующий букве.

Регистр учитывается.

Длина имени переменной не ограничивается.

Несмотря на то что знак **$** считается достоверным в Java, пользоваться им для именования элементов прикладного кода не рекомендуется. Ведь он служит для обозначения имен, формируемых компилятором Java и другими инструментальными средствами.

Инициализация переменных в Java

Объявление переменной можно размещать в любом месте кода Java

Рекомендуется объявлять переменную как можно ближе к тому месту кода, где предполагается ее использовать.

В Java объявления и определения переменных не различаются

```
double salary = 100.0;
System.out.println(salary);
int vacationDays = 14;
```

ПРИМИТИВНЫЕ ТИПЫ ДАННЫХ

Примитивные типы данных

1. Целые числа - **byte, short, char, int, long**
2. Числа с плавающей точкой - **float, double**
3. Логический – **boolean** (true, false)

Тип	**Размер (бит)**	**Диапазон**
float	32	от -1.4e-45f до 3.4e+38f
double	64	от -4.9e-324 до 1.7e+308

Примитивные типы данных

Тип	Размер (бит)	Диапазон
byte	8 бит	от -128 до 127
short	16 бит	от -32768 до 32767
char	16 бит	беззнаковое целое число, представляющее собой символ UTF-16 (буквы и цифры)
int	32 бит	от -2147483648 до 2147483647
long	64 бит	от -9223372036854775808L до 9223372036854775807L

Пример. Тип char

```
public class MyChar1 {
    public static void main(String args[]) {

        char a = 'a', b, c = 'c';

        b = (char) ((a + c) / 2);

        // Можно складывать, вычитать, делить и умножать
        // Но из-за особенностей арифметики Java
        // результат приходится приводить к типу char явно
        System.out.println("b = " + b);
        }
    }
```

```
b = b
```

Пример. Тип char

```
public class MyChar2 {

    public static void main(String args[]) {
        char ch1, ch2;

        ch1 = 88;  // code for X
        ch2 = 'Y';

        System.out.print("ch1 and ch2: ");
        System.out.println(ch1 + " " + ch2);
        }
    }
```

```
ch1 and ch2: X Y
```

Пример. Тип char

```
public class MyChar3 {

   public static void main(String args[]) {
      char ch1;

      ch1 = 'X';
      System.out.println("ch1 contains " + ch1);

      ch1++; // increment ch1
      System.out.println("ch1 is now " + ch1);
      }
   }
```

```
ch1 contains X
ch1 is now Y
```

Пример. Тип char. Unicode в шестнадцатеричном формате

```
public class MyUCode1 {

    public static void main(String[] args) {

        System.out.print('\u0048');
        System.out.print('\u0045');
        System.out.print('\u004C');
        System.out.print('\u004C');
        System.out.print('\u004F');
        System.out.println('\u0021');

    }
```

HELLO!

Пример. Тип char. Unicode в шестнадцатеричном формате

```
public class MyUCode2 {
    public static void main(String[] args) {
        System.out.println("\u0048" +
                           "\u0045" +
                           "\u004C" +
                           "\u004C" +
                           "\u004F" +
                           "\u0021");
    }
}
```

HELLO!

Тип boolean

Тип boolean

Тип данных boolean имеет два логических значения:

false и **true**,

которые служат для вычисления логических выражений.

Преобразование значений типа boolean в целочисленные значения и наоборот невозможно.

```
int x;
if (x=0)
```

Incompatible types.
Required: boolean
Found: int

```
int x;
if (x==0)
```

В C++ возникновение такой ошибки на этапе выполнения возможно, а в Java - нет!

Пример . Тип boolean

```
public class MyBool1 {

   public static void main(String args[]) {
      boolean b;

      b = false;
      System.out.println("b is " + b);
      b = true;
      System.out.println("b is " + b);

      // a boolean value can control the if statement
      if(b) System.out.println("This is executed.");

      b = false;
      if(b) System.out.println("This is not executed.");

      // outcome of a relational operator is a boolean value
      System.out.println("10 > 9 is " + (10 > 9));
      }    }
```

```
b is false
b is true
This is executed.
10 > 9 is true
```

Тип double

Пример. Тип double

```
public class MyDooouble1 {

    public static void main(String args[]) {
        double pi, r, a;

        r = 10.8; // radius of circle
        pi = 3.1416; // pi, approximately
        a = pi * r * r; // compute area

        System.out.println("Area of circle is " + a);
        }
    }
```

```
Area of circle is 366.436224
```

Пример. Тип double

```
public class MyDouble2 {

    public static void main(String args[]) {
        double a = 3.0, b = 4.0;

        // c is dynamically initialized
        double c = Math.sqrt(a * a + b * b);

        System.out.println("Hypotenuse is " + c);
        }
    }
```

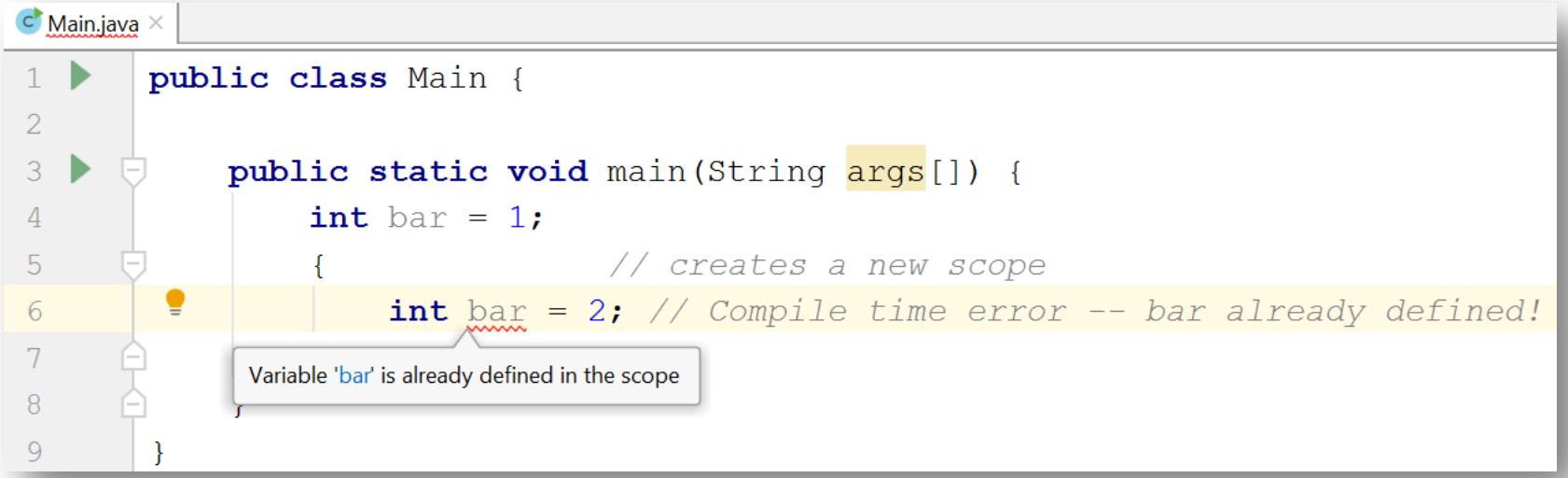
```
Hypotenuse is 5.0
```

Пример. Про скобки!

```
public class MyBracket1{

   public static void main(String args[]) {
      int x; // known to all code within main

      x = 10;
      if(x == 10) { // start new scope
         int y = 20; // known only to this block

         // x and y both known here.
         System.out.println("x and y: " + x + " " + y);
         x = y * 2;
      }
      // y = 100; // Error! y not known here

      // x is still known here.
      System.out.println("x is " + x);
   }
}
```

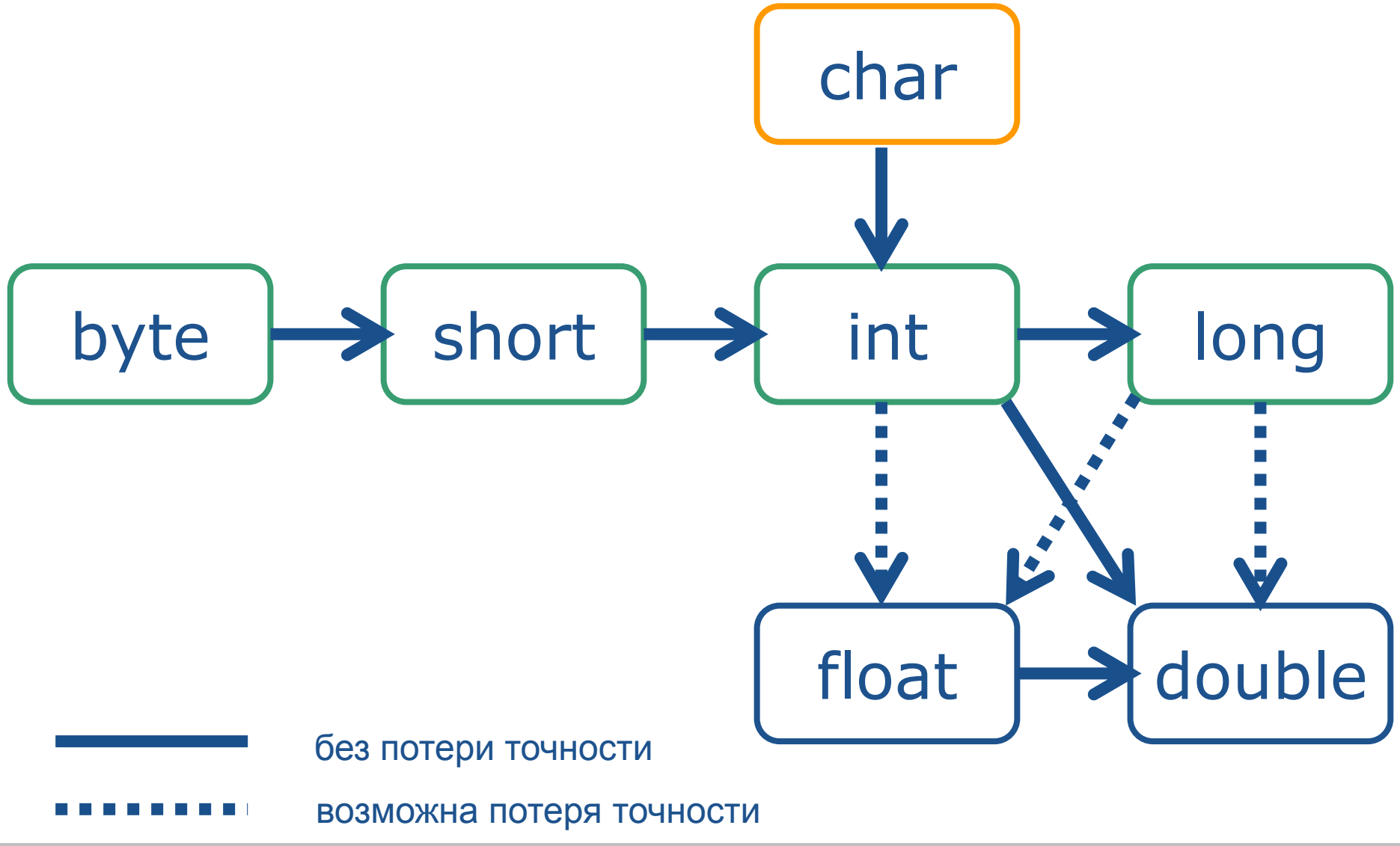
```
x and y: 10 20
x is 40
```

Пример. Про скобки!



Преобразование типов

Преобразование типов



Правила приведения типов

- ✓ Если хотя бы один из операндов относится к типу **double**, то и второй операнд преобразуется в тип double.
- ✓ Иначе, если хотя бы один из операндов относится к типу **float**, то и второй операнд преобразуется в тип float.
- ✓ Иначе, если хотя бы один из операндов относится к типу **long**, то и второй операнд преобразуется в тип long.
- ✓ Иначе оба операнда преобразуются в тип **int**.

```
double x = 9.997;

int nx = (int)Math.round(x);
```

Правила приведения типов

При попытке приведения типов результат может выйти за пределы диапазона допустимых значений.

И в этом случае произойдет усечение.

Например, при вычислении выражения
(byte) 300 получается значение 44.

Приведение логических значений к целым и наоборот **невозможно**.

Такое ограничение предотвращает появление ошибок.

Пример. Преобразование типов

```
public class MyToType1 {

    public static void main(String args[]) {
        byte b;
        int i = 257;
        double d = 323.142;

        System.out.println("\nConversion of int to byte.");
        b = (byte) i;
        System.out.println("i and b " + i + " " + b);
        System.out.println("\nConversion of double to int.");
        i = (int) d;
        System.out.println("d and i " + d + " " + i);
        System.out.println("\nConversion of double to byte.");
        b = (byte) d;
        System.out.println("d and b " + d + " " + b);
    }}
```

```
Conversion of int to byte.
i and b 257 1

Conversion of double to int.
d and i 323.142 323

Conversion of double to byte.
d and b 323.142 67
```

Константы

Константы

Для обозначения констант служит ключевое слово

final

Ключевое слово **final** означает, что присвоить данной переменной какое-нибудь значение можно лишь один раз, после чего изменить его уже нельзя.

Принято использовать в именах констант только прописные буквы.

Константы. Пример

```
public class UseConstExample {

    public static final double MY_PI = 3.14;

    public static void main(String[] args) {

        System.out.println(MY_PI);

        System.out.println(Math.PI);
    }
}
```

```
3.14

3.141592653589793
```

Константы. Использование константы в другом методе

```
public class UseConstExample {

    public static final double MY_PI = 3.14;

    public static double circlesquare(double r) {
        return MY_PI*r*r;
    }


    public static void main(String[] args) {

        System.out.println(circlesquare(1));
    }}
```

3.14

константа класса задается **за пределами метода main** (), поэтому ее можно использовать в других методах того же класса.

Вещественные константы в Java

```
public class MyConst1 {

    public static void main(String args[]) {
// экспоненциальая форма представления числа
        System.out.println(2.5e3);
        System.out.println(2.5E+3);
        System.out.println(2.5E-3);

// формат float
        System.out.println(2.5e-3F);
        System.out.println(2.5E-3f);

// формат double
        System.out.println(2.5e-3D);
        System.out.println(2.5E-3d);
    }
}
```

```
2500.0
2500.0
0.0025

0.0025
0.0025

0.0025
0.0025
```

Целочисленные константы в Java

```
public class MyConst2 {

    public static void main(String args[]) {
        // признак восьмеричной константы - 0
        System.out.println(017);

        // признак шестнадцатиричной константы - 0x или 0X
        System.out.println(0x1f);
        System.out.println(0X1F);

        // признак хранения в long формате L или l
        System.out.println(25L);
        System.out.println(0x1fL);
        System.out.println(0X1FL);
    }
}
```

```
15

31
31

25
31
31
```

Управляющие символы

```
public class MySymb1 {

    public static void main(String args[]) {

        System.out.println("A"+"\n"+"B");
/*
 '\n' - символ перевода строки newline с кодом ASCII 10;
 '\r' - символ возврата каретки CR с кодом 13;
 '\f' - символ перевода страницы FF с кодом 12;
 '\b' - символ возврата на шаг BS с кодом 8;
 '\t' - символ горизонтальной табуляции HT с кодом 9;  //
 '\\' - обратная наклонная черта;
 '\"' - кавычка;
 '\'' - апостроф.
*/  }
}
```

```
A
B
```

Вывести таблицу значений x sin(x), используя управляющие символы.

Символьные константы в Java

```
public class MySymb2 {

    public static void main(String args[]) {
        // код в кодировке 1251 символ A => \101
        System.out.println("\101");

        // код в кодировке Unicode символ A => \u0041
        System.out.println("\u0041");
    }
}
```

```
A

A
```

https://unicode-table.com/ru

Компилятор и исполняющая система Java работают только с кодировкой Unicode.

Вывести таблицу кодов и символов английского алфавита.

Математические функции и константы

Математические функции и константы

Math.pow(x,a) = x^a

Math.sqrt(x) = $x^{1/2}$

Math.sin
Math.cos
Math.tan
Math.atan
Math.atan2

```
System.out.println(Math.floorMod(13,5));
System.out.println(Math.floorDiv(13,5));
```

```
3
2
```

```
System.out.println(Math.exp(1));

System.out.println(Math.log(Math.E));
System.out.println(Math.log10(100));

System.out.println(Math.PI);
```

```
2.718281828459045

1.0

2.0

3.141592653589793
```

Математические функции и константы

```
// подключение библиотеки Math
import static java.lang.Math.*;


public class Example {

    public static void main(String args[])
    {

    System.out.println("The square root of \u03C0 is " + sqrt(PI));


    }
}
```

The square root of π is 1.7724538509055159

Горячие клавиши IntelliJ IDEA

Горячие клавиши IntelliJ IDEA

Ctr + Alt + L

- приведение кода в соответствие code style

Shift + F10

- запустить

Ctrl + S

- Сохранить все

Горячие клавиши IntelliJ IDEA

Ctrl + наведение мышью на фрагмент кода

- краткая информация

Ctrl + Shift + /

- многострочное комментирование / раскомментирование

Ctrl + /

- однострочное комментирование /раскомментирование

Welcome to Core Java!

Пример. Welcome to Core Java!

```
/**
 * @author Cay Horstmann
 */
public class Main {

   public static void main(String[] args) {

      String greeting = "Welcome to Core Java!";

      System.out.println(greeting);

      for (int i = 0; i < greeting.length(); i++)
         System.out.print("=");

      System.out.println();
   }
}
```

Спасибо за внимание!