

Программирование на языке Java

9. Типы с плавающей точкой
10. Методы класса **Методы класса**
Math

Программирование на языке Java

Тема 9. Типы с плавающей точкой

Типы с плавающей точкой – 1

Числа с плавающей точкой, называемые **действительными (вещественными)**, используются при вычислениях, которые требуют получения результата с точностью до определенного десятичного знака.

Пример. Вычисление квадратного корня, трансцендентных функций (`sin()`, `cos()`, ...).

В Java существует два типа с плавающей точкой: `float` и `double` (числа одинарной и двойной точности).

Типы с плавающей точкой – 2

Стандарт **IEEE754**

Число представлено в виде $\pm m \cdot 2^e$,

где m – мантисса, e – порядок (экспонента)

Тип	Бит	Знак	Мантисса	Порядок	Min	Max
float	32	1	23	8	1.4e-045	3.4e038
double	64	1	52	11	4.9e-324	1.8e308

Типы с плавающей точкой – 3

Тип `float` используется, когда требуется дробная часть без **особой точности**, например, для представления денежных сумм в рублях и копейках.

Применение типа `double` наиболее рационально, когда требуется сохранение точности множества последовательных вычислений или манипулирование большими числами.

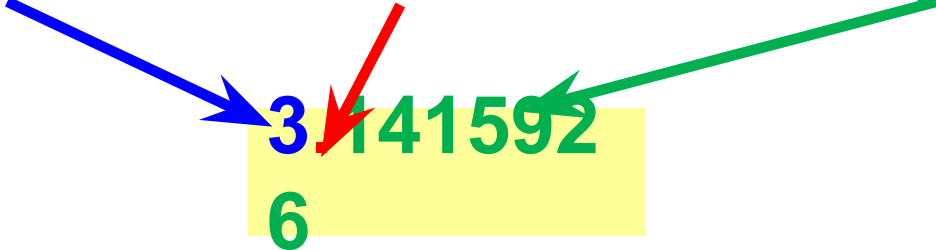
Все трансцендентные математические функции (`sin()`, `cos()`, `sqrt()`, ...) возвращают значения типа `double`.

Константы с плавающей точкой – 1

Числа с плавающей точкой представляют десятичные значения с дробной частью.

Стандартная форма записи десятичного числа состоит из:

целого числа; десятичной точки; дробной части.



3.1415926

Константы с плавающей точкой – 2

Научная форма записи десятичного числа
СОСТОИТ ИЗ:

мантиссы; символа E, суффикса,
указывающего степенью функцию числа

10



3.14159e0

Константы с плавающей точкой – 3

Задача. Записать в стандартной форме

$$1.44e-6 = 0.00000144$$

$$0.832e8 = 83200000.0$$

$$0.000034e7 =$$

340.

$$0.00524e-1 = 0.000524$$

Константы с плавающей точкой – 4

По умолчанию в **Java** константам с плавающей точкой присвоен тип **double**.

Для указания константы типа **float**, к ней нужно дописать символ **F** или **f**.

```
float x;  
x = 23.48f;
```

Также существует суффикс **d** или **D**

```
double y = 3D;
```

Особые случаи: бесконечность

- Деление положительного числа на 0.0 дает $+\infty$
- Деление отрицательного числа на 0.0 дает $-\infty$
- Переполнение дает $+\infty$ или $-\infty$, в зависимости от направления

```
double posInfinity = 1.0 / 0.0;  
double negInfinity = -1.0 / 0.0;  
double x = posInfinity + 1;
```

Какое значение примет x?

Особые случаи: NaN

- Деление 0.0 на 0.0 дает **NaN** (Not a Number – не число)
- Любая арифметическая операция с NaN дает NaN
- NaN != NaN

```
double nan = 0.0 / 0.0;  
nan = posInfinity + negInfinity ;
```

Значение NaN

К получению **NaN** приводит:

- все математические операции с NaN;
- деление нуля на ноль;
- деление бесконечности на бесконечность;
- умножение нуля на бесконечность;
- сложение бесконечностей с противоположными знаками;
- вычисление квадратного корня отрицательного числа;
- логарифмирование отрицательного числа.

Точность вычислений

- Для любого a вещественного типа существует $\varepsilon > 0$: $a + \varepsilon == a$

```
double t = 0.1 + 0.1 + 0.1 + 0.1 + 0.1 +
           0.1 + 0.1 + 0.1 + 0.1 + 0.1;
System.out.println(t);
```

0.99999999999999999999

Почему?

- Из-за погрешности вычислений, поэтому при сравнении вещественных чисел проверяют не равенство ($a == b$), а близость этих чисел

$$|a - b| < \varepsilon$$

Почему по модулю?

Модификатор `strictfp`

- Java использует математический сопроцессор (FPU – Floating Point Unit) для вычислений с плавающей точкой;
- Регистры FPU могут быть шире 64 бит
- Результаты вычислений могут отличаться
- Модификатор `strictfp` включает режим строгой совместимости, результаты будут идентичны на любом процессоре

Ввод с клавиатуры

Аналогично вводу целых чисел

```
Scanner in = new Scanner(System.in);  
System.out.println("Введите x");  
float x = in.nextFloat();  
System.out.println("Введите y");  
double y = in.nextDouble();
```

```
Введите x  
12,567  
Введите y  
0,00034
```

Внимание! При считывании с клавиатуры используется разделитель запятой

**Почему
запятая?**

Форматный вывод – 1

Спецификаторы формата `%f`, `%e`, `%g`.

```
double x = 12345.6789;  
System.out.printf ("%f",  
x);
```

12345,678900

минимальное число
позиций, **6 цифр** в
дробной части

```
System.out.printf ("%e",  
x);
```

1.234568e+04

Научная нотация:
 $1,23456 \cdot 10^4$

Форматный вывод. Указание точности

```
double x = 12345.6789;  
System.out.printf ("%10.3f", x);
```

12345,679

всего 10 позиций,
3 цифры в дробной
части

```
System.out.printf ("%10.2e",  
x);
```

1.23e+04

всего 10 позиций,
2 цифры в дробной
части мантииссы

Задача

Задача. Вычислить площадь круга заданного радиуса.

```
double pi, s, r;  
r = in.nextDouble();  
pi = 3.1415926;  
s = pi * r * r;
```

Считываем с
клавиатуры

Приблизительное
значение π

Вычисление площади
круга

```
System.out.printf("Площадь = %f", s);
```

Программирование на языке Java

**Тема 10. Методы класса
Math**

Класс `Math`

Разработчику на `Java` доступно множество готовых (или библиотечных) классов и методов, полезных для использования в собственных программах.

Наличие библиотечных решений позволяет изящно решать множество типовых задач.

Класс `Math` содержит методы, которые используются в геометрии и тригонометрии, а также некоторые методы общего назначения.

Константы класса Math

~3,14

2 константы типа `double`:

`Math.PI` – число π с точностью в 15 десятичных знаков.

~2,72

`Math.E` – основание натурального логарифма с точностью в 15 десятичных знаков.

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

```
3.141592653589793  
2.718281828459045
```

Прямые трансцендентные функции

Метод	Описание
<code>double sin(double arg)</code>	Возвращает синус угла <code>arg</code> , переданного в радианах
<code>double cos(double arg)</code>	Возвращает косинус угла <code>arg</code> , переданного в радианах
<code>double tan(double arg)</code>	Возвращает тангенс угла <code>arg</code> , переданного в радианах

Прямые трансцендентные функции. Пример

```
System.out.println(Math.sin(Math.PI/2));  
System.out.println(Math.cos(Math.PI/2));  
System.out.println(Math.tan(Math.PI/4));
```

```
1.0  
6.123233995736766E-17  
0.9999999999999999
```

Значение близкое
к нулю

Значение близкое
к единице

Почему не 0 и 1 ?

Обратные трансцендентные функции

Метод	Описание
<code>double asin(double arg)</code>	Возвращает угол, синус которого равен <code>arg</code> .
<code>double acos(double arg)</code>	Возвращает угол, косинус которого равен <code>arg</code> .
<code>double atan(double arg)</code>	Возвращает угол, тангенс которого равен <code>arg</code> .
<code>double atan2(double x, double y)</code>	Возвращает угол, тангенс которого равен x/y .

Обратные трансцендентные функции. Пример

```
System.out.println(Math.asin(1)*2);  
System.out.println(Math.acos(1));  
System.out.println(Math.atan(0));  
System.out.println(Math.atan2(1,1)*4);
```

```
3.141592653589793  
0.0  
0.0  
3.141592653589793
```

Гиперболические функции

Метод	Описание
<code>double sinh (double arg)</code>	Возвращает гиперболический синус угла <code>arg</code> , переданного в радианах .
<code>double cosh (double arg)</code>	Возвращает гиперболический косинус угла <code>arg</code> , переданного в радианах .
<code>double tanh (double arg)</code>	Возвращает гиперболический тангенс угла <code>arg</code> , переданного в радианах .

Экспоненциальные функции

Метод	Описание
<code>double exp(double arg)</code>	Возвращает экспоненту <code>arg</code> .
<code>double log(double arg)</code>	Возвращает натуральный логарифм <code>arg</code> .
<code>double log10(double arg)</code>	Возвращает логарифм по основанию 10 от <code>arg</code> .
<code>double pow(double y, double x)</code>	Возвращает <code>y</code> в степени <code>x</code> .
<code>double sqrt(double arg)</code>	Возвращает квадратный корень из <code>arg</code> .

Экспоненциальные функции. Пример

```
System.out.println(Math.exp(1));  
System.out.println(Math.exp(2));  
System.out.println(Math.log(1));  
System.out.println(Math.log(Math.E));  
System.out.println(Math.log10(1000));  
System.out.println(Math.pow(2, 3));  
System.out.println(Math.sqrt(25));
```

```
2.7182818284590455
```

```
7.38905609893065
```

```
0.0
```

```
1.0
```

```
3.0
```

```
8.0
```

```
5.0
```

Функции округления – 1

Метод	Описание
<code>int abs(int arg)</code>	Возвращает абсолютное значение <code>arg</code> .
<code>long abs(long arg)</code>	Возвращает абсолютное значение <code>arg</code> .
<code>float abs(float arg)</code>	Возвращает абсолютное значение <code>arg</code> .
<code>double abs(double arg)</code>	Возвращает абсолютное значение <code>arg</code> .



Чем эти методы отличаются?

Функции округления – 1. Пример

```
System.out.println(Math.abs(5));  
System.out.println(Math.abs(-5));  
System.out.println(Math.abs(10.3));  
System.out.println(Math.abs(-10.3));
```

```
5  
5  
10.3  
10.3
```

Функции округления – 2

Метод	Описание
<code>double ceil(double arg)</code> 	Возвращает наименьшее целое число, которое больше <code>arg</code> .
<code>double floor(double arg)</code> 	Возвращает наибольшее целое число, которое меньше или равно <code>arg</code> .
<code>int round(float arg)</code>	Возвращает <code>arg</code> , округленное до ближайшего <code>int</code> .
<code>long round(double arg)</code>	Возвращает <code>arg</code> , округленное до ближайшего <code>long</code> .

Функции округления – 2. Пример

ПОТОЛОК

```
System.out.println(Math.ceil(5.4));  
System.out.println(Math.floor(5.4));  
System.out.println(Math.round(5.4));  
System.out.println(Math.round(5.6));  
System.out.println(Math.round(5.5));
```

```
6.0  
5.0  
5  
6  
6
```

ПОЛ

Функции округления – 3

Метод	Описание
<code>int max(int x, int y)</code>	Возвращает большее из двух чисел x и y .
<code>long max(long x, long y)</code>	Возвращает большее из двух чисел x и y .
<code>float max(float x, float y)</code>	Возвращает большее из двух чисел x и y .
<code>double max(double x, double y)</code>	Возвращает большее из двух чисел x и y .

Функции округления – 3

Метод	Описание
<code>int min(int x, int y)</code>	Возвращает меньшее из двух чисел x и y .
<code>long min(long x, long y)</code>	Возвращает меньшее из двух чисел x и y .
<code>float min(float x, float y)</code>	Возвращает меньшее из двух чисел x и y .
<code>double min(double x, double y)</code>	Возвращает меньшее из двух чисел x и y .

Функции округления – 3. Пример

```
System.out.println(Math.max(2, 4));  
System.out.println(Math.min(2, 4));  
System.out.println(Math.max(10.3, 4));  
System.out.println(Math.min(10.3, 4));
```

```
4  
2  
10.3  
4.0
```

Почему 4.0, а не 4?



Как вычислить максимум из трех чисел?

Прочие функции

Метод	Описание
<code>double toDegrees (double angle)</code>	Преобразует радианы в градусы. Переданный в <code>angle</code> угол должен быть указан в радианах. Возвращается результат в градусах.
<code>double toRadians (double angle)</code>	Преобразует градусы в радианы. Переданный в <code>angle</code> угол должен быть указан в градусах. Возвращается результат в радианах.

Прочие функции. Пример

```
System.out.println(Math.toDegrees(Math.PI));  
System.out.println(Math.toDegrees(Math.PI/4));  
;  
System.out.println(Math.toRadians(180));  
System.out.println(Math.toRadians(90));
```

180.0

45.0

3.141592653589793

1.5707963267948966

Псевдослучайные числа

Метод `Math.random()` возвращает псевдослучайное вещественное число из промежутка $[0;1)$.

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

```
0.8701659383706429  
0.5194884184661862  
0.3324845299964946
```

Случайные
числа

Целые числа в заданном интервале – 1

Целые числа в интервале $[0, n-1]$:

```
(int) (Math.random() * n);
```

Примеры:

```
x = (int) (Math.random () * 100); // [0,99]
```

```
x = (int) (Math.random () * z); // [0,z-1]
```

Целые числа в интервале $[a, b]$:

```
x = (int) (Math.random () * (b - a + 1)) + a;  
// [a,b]
```

Целые числа в заданном интервале – 2

Задача. Получить случайное число в интервале от -10 до 10.

```
int x = random () * 21 - 10;
```


Методы класса Math. Задача – 1

```
System.out.println(Math.abs(-2.33));
System.out.println(Math.round(Math.PI));
System.out.println(Math.round(9.5));
System.out.println(Math.round(9.5-0.001));
;
System.out.println(Math.ceil(9.4));
double c = Math.sqrt(3*3 + 4*4);
System.out.println(c);
double s1 = Math.cos(Math.toRadians(60));
System.out.println(s1);
```

2.33

3

10

9

10.0

5.0

0.5

Методы класса Math. Задача – 2

Записать в стандартной форме записи числа

$$-12.3\text{E}+2 = \mathbf{-1230.0}$$

$$-0.8\text{E}-6 = \mathbf{-0.0000008}$$

$$1\text{E}+3 = \mathbf{1000.0}$$

$$+1\text{E}-6 = \mathbf{0.000001}$$

Методы класса Math. Задача – 3

Какие круглые скобки можно убрать, не изменив порядка вычисления выражений

$$(a+b) / c$$

$$a+b/c$$

$$a / (b*c)$$

$$x1/x2*y$$

$$\text{Math.sqrt}(p) * q/r$$

$$a-b-c-d-e$$

$$(a-b) - (c-d) - e$$

Методы класса Math. Задача – 4

Записать следующие выражения на Java

x^5

```
Math.pow(x, 5)
```

$\cos^8 x^4$

```
Math.pow(Math.cos(Math.pow(x, 4)), 8)
```

$\log_{10}(x/5)$

```
Math.log10(x/5)
```

$|x^{-3}|$

```
Math.abs(Math.pow(x, -3))
```

2^{x+1}

```
Math.pow(2, x+1)
```

$\sin 8^\circ$

```
Math.sin(Math.toRadians(8))
```

Методы класса Math. Задача – 5

Определить типы выражений

```
double x, y, z;
```

```
int i, j, k;
```

```
x+y*i;
```

```
i+j-k;
```

```
i/j+x;
```

```
i*x+j*y;
```

double

int

double

double

Задача

Задача. Дано целое число x . Вывести количество цифр данного числа.

```
int x;  
x = in.nextInt();  
double count = Math.floor(Math.log10(x)) +  
1;  
System.out.println(count);
```

Что плохо?