ClassMath

Класс Math располагается в пакете java.lang и предоставляет набор статических методов для осуществления ряда различных математических вычислений.Ниже приведены примеры вычислений, для которых класс Math может оказаться полезным:

* Вычисление абсолютных значений (значений по модулю)
* Вычисление значений тригонометрических функций (синусов, косинусов и т.д.)
* Возведение в различные степени
* Извлечение корней различных степеней
* Генерация случайных чисел
* Округления
* И пр.

Ниже мы попробуем рассмотреть как класс Java Math помогает решать задачи, перечисленные выше. Начнем разбор класса с методов, которые позволяют вычислить значение по модулю. За это отвечает метод abs. Данный метод перегружен и в классе Math имеются следующие его различия:

* static double abs(double a)
* static float abs(float a)
* static int abs(int a)
* static long abs(long a)

Пример использования:

public static void main(String[] args) {

System.out.println(Math.abs(-1)); // 1

System.out.println(Math.abs(-21.8d)); // 21.8

System.out.println(Math.abs(4532L)); // 4532

System.out.println(Math.abs(5.341f)); // 5.341

}

## Вычисление значений тригонометрических функций

Класс Math позволяет вычислять различные тригонометрические функции - синусы, косинусы, тангенсы и пр. С полным перечнем методов можно познакомиться [на сайте официальной документации](https://docs.oracle.com/javase/8/docs/api/java/lang/Math.html). Ниже приведен некоторый перечень данных методов:

* static double sin(double a)
* static double cos(double a)
* static double tan(double a)
* static double asin(double a)
* static double acos(double a)
* static double atan(double a)

Методы вычисляют: синус, косинус, тангенс, арксинус, арккосинус, арктангенс. Каждый метод вычисляет значение для угла `a`. Этот параметр передается в каждый метод и в каждом случае измеряется в радианах (а не в градусах, как мы привыкли). Тут есть две новости, хорошая и плохая. Начнем с хорошей. В классе Math есть методы для перевода радианов в градусы и градусов в радианы:

* static double toDegrees(double angrad)
* static double toRadians(double angdeg)

Здесь метод toDegrees переведет угол angrad, измеряемый в радианах в градусы. А метод toRadians, наоборот, переводит угол angdeg, измеряемый в градусах - в радианы. А плохая новость в том, что происходит это с некоторой погрешностью. Приведем пример вычисления синусов и косинусов:

public static void main(String[] args) {

System.out.println(Math.sin(Math.toRadians(0)));

System.out.println(Math.sin(Math.toRadians(30)));

System.out.println(Math.sin(Math.toRadians(90)));

System.out.println(Math.cos(Math.toRadians(0)));

System.out.println(Math.cos(Math.toRadians(30)));

System.out.println(Math.cos(Math.toRadians(90)));

}

Программа выведет:

0.0

0.49999999999999994

1.0

1.0

0.8660254037844387

6.123233995736766E-17

Что не совсем соответствует таблицам синусов и косинусов, отчасти благодаря погрешностям при переводе из градусов в радианы.

## Возведение в степень

Для возведения числа в степень класс Math предоставляет метод pow, который имеет следующую сигнатуру:

static double pow(double a, double b)

Данный метод возводит параметр `a` в степень `b`. Примеры:

public static void main(String[] args) {

System.out.println(Math.pow(1,2)); // 1.0

System.out.println(Math.pow(2,2)); // 4.0

System.out.println(Math.pow(3,2)); // 9.0

System.out.println(Math.pow(4,2)); // 16.0

System.out.println(Math.pow(5,2)); // 25.0

System.out.println(Math.pow(1,3)); // 1.0

System.out.println(Math.pow(2,3)); // 8.0

System.out.println(Math.pow(3,3)); // 27.0

System.out.println(Math.pow(4,3)); // 64.0

System.out.println(Math.pow(5,3)); // 125.0

}

## Извлечение корней

Класс Math предоставляет методы для извлечения корней: квадратного и кубического. За данную процедуру отвечают следующие методы:

* static double sqrt(double a)
* static double cbrt(double a)

Метод sqrt - извлекает квадратный корень (square root), а метод cbrt - кубический (cube root). Примеры:

public static void main(String[] args) {

System.out.println(Math.sqrt(4)); // 2.0

System.out.println(Math.sqrt(9)); // 3.0

System.out.println(Math.sqrt(16)); // 4.0

System.out.println(Math.cbrt(8)); // 2.0

System.out.println(Math.cbrt(27)); // 3.0

System.out.println(Math.cbrt(125)); // 5.0

}

## Генерация случайных чисел

Для генерации случайных чисел класс Math предоставляет метод random. Данный метод генерирует случайное позитивное вещественное (double) число в промежутке от 0.0 до 1.0. Сигнатура метода имеет следующий вид:

public static double random()

Взглянем на примеры:

public static void main(String[] args) {

for (int i = 0; i < 5; i++) {

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

}

}

После выполнения метода мейн на консоль вывелось:

0.37057465028778513

0.2516253742011597

0.9315649439611121

0.6346725713527239

0.7442959932755443

С помощью небольших манипуляций, можно использовать метод random класса Math для получения целочисленных случайных чисел лежащих в определенном диапазоне. Приведем пример функции которая принимает два аргумента min и max и возвращает случайное целое число, которое лежит в промежутке от min (включительно) до max (включительно):

static int randomInARange(int min, int max) {

return (int) (Math.random() \* ((max - min) + 1)) + min;

}

Напишем метод Main, в котором протестируем метод randomInARange:

public class MathExample {

public static void main(String[] args) {

// Карта, в которой мы будем хранить количество выпадений какого-то числа

Map<Integer, Integer> map = new TreeMap<>();

// За 10000 операций

for (int i = 0; i < 10000; i++) {

// Сгенерируем рандомное число от -10 включительно до 10 включительно

final Integer randomNumber = randomInARange(-10, 10);

if (!map.containsKey(randomNumber)) {

// Если карта еще не содержит "выпавшего случайного числа"

// Положим его в карту с кол-вом выпадений = 1

map.put(randomNumber, 1);

} else {

// Иначе, увеличим количество выпадений данного числа на 1

map.put(randomNumber, map.get(randomNumber) + 1);

}

}

// Выведем на экран содержимое карты в формате ключ=[значение]

for (Map.Entry<Integer, Integer> entry : map.entrySet()){

System.out.println(String.format("%d=[%d]", entry.getKey(), entry.getValue()));

}

}

static int randomInARange(int min, int max) {

return (int) (Math.random() \* ((max - min) + 1)) + min;

}

}

После запуска метода main вывод может быть таким:

-10=[482]

-9=[495]

-8=[472]

-7=[514]

-6=[457]

-5=[465]

-4=[486]

-3=[500]

-2=[490]

-1=[466]

0=[458]

1=[488]

2=[461]

3=[470]

4=[464]

5=[463]

6=[484]

7=[479]

8=[459]

9=[503]

10=[444]

Process finished with exit code 0

## Округление

Для округления чисел в Java одним из инструментов являются методы класса Math. А точнее методы round, ceil и floor:

* static long round(double a)
* static int round(float a)
* static double floor(double a)
* static double ceil(double a)

Метод round - округляет так, как привычно обывателю. Если дробная часть числа больше либо равна 0.5, то число будет округлено в большую сторону, иначе в меньшую. Метод floor всегда, независимо от значений дробной части, округляет число в меньшую сторону (в сторону отрицательной бесконечности). Метод ceil, напротив, вне зависимости от значений дробной части, округляет числа в большую сторону (в сторону положительной бесконечности). Взглянем на примеры:

public static void main(String[] args) {

System.out.println(Math.round(1.3)); // 1

System.out.println(Math.round(1.4)); // 1

System.out.println(Math.round(1.5)); // 2

System.out.println(Math.round(1.6)); // 2

System.out.println(Math.floor(1.3)); // 1.0

System.out.println(Math.floor(1.4)); // 1.0

System.out.println(Math.floor(1.5)); // 1.0

System.out.println(Math.floor(1.6)); // 1.0

System.out.println(Math.ceil(1.3)); // 2.0

System.out.println(Math.ceil(1.4)); // 2.0

System.out.println(Math.ceil(1.5)); // 2.0

System.out.println(Math.ceil(1.6)); // 2.0

}