Практическая работа № 3. Классы Math и Random. Классы оболочки

Цель данной практической работы - изучить работу с классами Math и Random основные концепции объектно-ориентированного программирования, научиться программировать математические вычисления с использованием этих классов, а также познакомиться с классами оболочками и их использованием в Джава программах и научиться форматировать вывод строк.

Теоретические сведения

Класс Math

Класс Java Math предоставляет ряд методов для работы выполнения математических вычислений, например таких как как min (), max (), sqrt(), pow(), sin(), cos(), tan(), round (), abs ( ) так далее. В классе также ест константа число PI. Все методы класса публичные и статические, поэтому вы можете вызывать их, обратившись напрямую через точку к классу, не создавая объект типа класс. Если размер равен int или long и результаты выходят за пределы диапазона значений, методы addExact (), subtractExact (), multiplyExact () и toIntExact () вызывают исключение ArithmeticException.

Для других арифметических операций, таких как увеличение, уменьшение, деление, абсолютное значение и отрицание, переполнение происходит только с определенным минимальным или максимальным значением. При необходимости его следует сравнивать с максимальным и минимальным значением. Класс Java Math имеет множество методов, которые позволяют решать на Джава математические задачи.

```
public class JavaMathExample1{
public static void main(String[] args){
double x = 28;
double y = 4;
// вернет maximum из двух чисел
System.out.println("Maximum number of x and y is: "
+Math.max(x, y));
// вернет квадратный корень из y
System.out.println("Square root of y is: " +
Math.sqrt(y));
//вергнет 28 в четвертой степени 28*28*28*28
System.out.println("Power of x and y is: " +
Math.pow(x, y));
// вернет логарифм заданного значения
```

```
    System.out.println("Logarithm of x is: " +
Math.log(x));
    System.out.println("Logarithm of y is: " +
Math.log(y));
    // вернет логарифм из десяти по основанию 10
    System.out.println("log10 of x is: " + Math.log10(x));
    System.out.println("log10 of y is: " + Math.log10(y));

    // вернет log из x + 1
    System.out.println("log1p of x is: " +Math.log1p(x));
    /* Метод Math.exp() — возвращает натуральный логарифм
по основанию e и аргументу — показателю степени, где e —
иррациональная константа, равная приблизительно 2,71828182
*/
    System.out.println("exp of a is: " +Math.exp(x));
    System.out.println("expm1 of a is: " +Math.expm1(x));
        }
    }
```

Генерация случайных чисел в Джава

Язык Джава предоставляет три способа генерации случайных чисел с использованием некоторых встроенных методов и классов, перечисленных ниже:

1. Класс Random
2. Метод random ()
3. класс ThreadLocalRandom

Класс Random находится в пакете java.util, соответственно для работы с ним вы должны импортировать этот пакет. Метод random() это метод класса Math, он может генерировать случайные числа типа double. Рассмотрим первый подход, с помощью класса Random. Этот класс имеет в своём составе различные методы для создания случайных чисел. Все они публичные и статические. Чтобы использовать этот класс для генерации случайных чисел, мы должны сначала создать экземпляр этого класса, а затем уже вызвать его методы, например такие как nextInt (), nextDouble (), nextLong () и тому подобные. С помощью этого класса мы можем генерировать случайные числа различных типов: integer, float, double, long, boolean. Мы можем передать аргументы методам для определения верхней границы диапазона генерируемых чисел. Например, если вызвать метод

nextInt (6) с аргументом 6, то будет генерироваться число в диапазоне от 0 до 5 включительно.

Листинг 3.1 – Пример генерации случайного числа

```
import java.util.Random;
public class generateRandom{
public static void main(String args[]){
// Создаем экземпляр класса Random
Random rand = new Random();

// Генерируем рандомно целые числа в диапазоне 0 to 999
int rand_int1 = rand.nextInt(1000);
int rand_int2 = rand.nextInt(1000);

// Печатаем полученные числа тип - int
System.out.println("Random Integers: "+rand_int1);
System.out.println("Random Integers: "+rand_int2);

// Генерируем с помощью Random тип double
double rand_dub1 = rand.nextDouble();
double rand_dub2 = rand.nextDouble();

// Печатаем полученные числа - тип double
System.out.println("Random Doubles: "+rand_dub1);
System.out.println("Random Doubles: "+rand_dub2);
    }
}
```

Результат работы программы на листинге 3.1 представлен ниже:

```
Random Integers: 547

Random Integers: 126

Random Doubles: 0.8369779739988428

Random Doubles: 0.5497554388209912
```

Рассмотрим второй способ с помощью метода Math.random (). Класс Math содержит различные методы для выполнения различных числовых операций, таких как вычисление возведения в степень, логарифмы и т. Д. Один из этих методов - random(), возвращает значение типа double с положительным знаком, больше или равно 0,0 и меньше 1,0. Возвращаемые значения выбираются

псевдослучайно. Этот метод может генерировать только случайные числа типа Double. Программа на листинге 3.2 демонстрирует как использовать этот метод при написании программ:

Листинг 3.2 – Пример использования метода random()

```
import java.util.*;
public class generateRandom{
public static void main(String args[]){
// Сгенерируем рандомно double
System.out.println("Random doubles: " +
Math.random());
System.out.println("Random doubles: " +
Math.random());
    }
}
```

Результат работы программы на листинге 3.2 представлен ниже:

```
Random doubles: 0.13077348615666562
Random doubles: 0.09247016928442775
```

Теперь рассмотрим третий подход для генерации случайного числа, с помощью класса ThreadLocalRandom. Этот класс представлен в java 1.7 для генерации случайных чисел типа целых, двойных, логических и т. Д. В программе ниже объясняется, как использовать этот класс для генерации случайных чисел:

Листинг 3.3 – Пример использования класса ThreadLocalRandom.

```
// сгенерируем рандомно числа
import java.util.concurrent.ThreadLocalRandom;
public class generateRandom{
public static void main(String args[]){
// создаем случ. целые числа в диапазоне 0 to 999
int rand_int1 =
ThreadLocalRandom.current().nextInt();
int rand_int2 =
ThreadLocalRandom.current().nextInt();
// печатаем случайные целые числа
System.out.println("Random Integers: " + rand_int1);
System.out.println("Random Integers: " + rand_int2);

// Генерируем рандомно double
```

```
    double rand_dub1 =
ThreadLocalRandom.current().nextDouble();
    double rand_dub2 =
ThreadLocalRandom.current().nextDouble();
    // печатаем случайные целые числа double
    System.out.println("Random Doubles: " + rand_dub1);
    System.out.println("Random Doubles: " + rand_dub2);

    // Генерируем рандомно boolean
    boolean rand_bool1 =
ThreadLocalRandom.current().nextBoolean();
    boolean rand_bool2 =
ThreadLocalRandom.current().nextBoolean();

    // печатаем случайные числа Boolean
    System.out.println("Random Booleans: " + rand_bool1);
    System.out.println("Random Booleans: " + rand_bool2);
        }
    }
```

Результат работы программы на листинге 3.3 представлен ниже:

```
Maximum number of x and y is: 28.0
Square root of y is: 2.0
Power of x and y is: 614656.0
Logarithm of x is: 3.332204510175204
Logarithm of y is: 1.3862943611198906
log10 of x is: 1.4471580313422192
log10 of y is: 0.6020599913279624
log1p of x is: 3.367295829986474
exp of a is: 1.4462570642914 75E12
expm1 of a is: 1.4462570642904 75E12
```

Для того, чтобы более подробно узнать про генерацию случайных чисел обратитесь к документации по языку Джава на сайте компании Oracle.[3]

Классы оболочки

В языке Джава у каждого примитивного типа есть соответствующий этому типу ссылочный или объектный тип класс – оболочка или обертка.

[3] https://docs.oracle.com

Ссылочный тип данных — это объект. В табл. 3.1 приведены примитивные типы данных и соответствующие им классы обертки:

Таблица 3.1 Соответствие примитивных и ссылочных типов

Примитивный тип	**Тип оболочка**
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
boolean	Boolean
char	Character

Типы оболочки используются в Джава, поскольку контейнеры не могут хранить примитивные типы данных. Поэтому необходимо выполнять преобразование примитивного типа к ссылочному. Это называется автоупаковка. Обратное преобразование называется автораспаковка.

Рассмотрим на примере класса Integer работу методов класса оболочки. Класс Integer — это класс оболочка для примитивного типа int. У класса Integer есть одно только поле типа int. Так как Integer это оболочка, то основгная его задача предоставить различные методы для работы с int, а также ряд методов для преобразования int в String и String в int. Пример работы с классом:

```
Integer num = 30;
int i = 15;
Integer a = i; //это автоупаковка
System.out.println(a); // 15
Integer x = new Integer(10); //конструктор Integer
System.out.println(x);
int b = x;//автораспаковка
```

Методы класса Integer

У класса Integer есть метод Integer.parseInt(String s). метод возвращает целое число, а именно возвращают примитивный тип данных int. Также у него есть переопределений метод Integer.toString() который возвращает строку (String) то есть символьное представление числа.

```
System.out.println(Integer.parseInt("0011", 2)); // 3
System.out.println(Integer.parseInt("10", 8)); // 8
System.out.println(Integer.parseInt("F", 16)); // 15
```

У данных методов есть аналог — метод valueOf().Отличие от parseInt в том, что результатом работы valueOf () будет объект Integer, а не int.

Форматирование вывода

Язык Джава предоставляет возможность использовать форматирование вывод с помощью использования форматно строки функции `printf` () функции. Для задания спецификаторов вывода используются те же самые управляющие символы, что и в языке Си. Пример использования спецификаторов представлен на листинге 3.4

Листинг 3.4 – Пример использования функции printf ()для форматирования вывода

```
public class Test{
public static void main(String args[]){
double x = 11.635;
System.out.printf("Значение e = %.3f%n", Math.E);
System.out.printf("exp(%.3f) = %.3f%n", x,
Math.exp(x));
    }
}
```

Спецификаторы формата вывода

Рассмотрим общий синтаксис спецификаторов выводы для его форматирования для общих типов, символьных и числовых:

`%[индекс_аргумента$][флаги][ширина][.точность]спецификатор`

Спецификаторы индекс аргумента, флаг, ширина и точность не являются обязательными.

- Индекс аргумента— это целое число i, указывающее, что здесь следует использовать i-й аргумент из списка аргументов
- Флаги– это набор символов, используемых для изменения формата вывода
- Ширина поля вывода — это положительное целое число, которое указывает минимальное количество символов, которое должно быть записано на выходе
- Точность — это целое число, обычно используемое для ограничения количества символов, конкретное поведение которых зависит от преобразования.
- спецификатор является обязательной частью. Это символ, указывающий, как должен быть отформатирован аргумент,

форматирование зависит от типа выводимого значения. Набор допустимых преобразований для данного аргумента зависит от типа данных аргумента.

Спецификатор символа - %c, спецификатор целого числа %d, %i, спецификатор вещественного числа %lf, f%, спецификатор строки %s.

В нашем примере выше, если мы хотим указать номер аргумента явно, мы можем записать его, используя индексы аргументов 1 $ и 2 $. Оба они являются первым и вторым аргументом соответственно:

```
String greetings = String.format( "Hello %2$s, welcome to %1$s !", "MIREA", "students");
```

Вы можете использовать спецификаторы вывода без аргументов, тогда следует использовать такой шаблон формы записи:

```
%[флаги][ширина]спецификатор
```

Форматирование вывода. Класс Formatter

В языке Джава для форматирования строкового вывода часто используется класс Formatter. Для его использования нужно подключить пакет java.util. Этот класс обеспечивает преобразование формата вывода (спецификаторы формата) позволяющие выводить числа, строки, время и даты практически в любом понравившемся вам формате. В классе Formatter есть метод format(), который преобразует переданные в него параметры в строку заданного формата и сохраняет в объекте типа Formatter. Рассмотрим использование класса Formatter. Для начала необходимо создать объект класса Formatter, а потом уже вызывать метод format() для форматирования вывода.

Листинг 3.5 – Пример использования Formatter

```
import java.util.Formatter;
public class public class FormatDemo1 {
public static void main(String[] args) {
Formatter fmt = new Formatter();
fmt.format("This %s is about %s %c ", "course", "java", '8');
//вместо первого %s подставится строка course
//вместо второго %s подставится строка java
//вместо %c подставится символ 8
System.out.print(f);
}}
```

Рассмотрим еще один пример, на листинге 3.6 приведен код, демонстрирующий работу с форматированием вывода вещественных чисел

Листинг 3.6 – Пример использования Formatter

```
import java.util.Formatter;
public class FormatDemo2 {
public static void main(String[] args) {
double x = 1000.0 / 3.0;
System.out.println("Строка без форматирования: " +
x);
Formatter formatter = new Formatter();
formatter.format("Строка с форматированием: %.2f%n",
x);
formatter.format("Строка с форматированием: %8.2f%n",
x);
formatter.format("Строка с форматированием:
%16.2f%n", x);
System.out.println(formatter);
    }
}
```

Форматирование чисел. Классы NumberFormat и DecimalFormat

Классы NumberFormat и DecimalFormat находятся в пакете java.text. В языке Джава класс NumberFormat языка Java используется для форматирования чисел. Чтобы получить объект класса для форматирования в национальном стандарте по умолчанию, используются следующие методы:

- NumberFormat.getlnstance()
- NumberFormat.getNumberlnstance()- идентичен getInstance()
- NumberFormat.getCurrencylnstance()
- NumberFormat.getPercentlnstance()

Чтобы получить объект класса для форматирования в других национальных стандартах используются следующие методы:

- NumberFormat.getlnstance(Locale locale)
- NumberFormat.getNumberlnstance(Locale locale) – идентичен методу getInstance(Locale locale)

Для значков валюты и процентов используются два метода класса NumberFormat:

- getCurrencylnstance(Locale locale)
- getPercentlnstance(Locale locale)

Метод getCurrencylnstance() задает добавление в вывод значка валюты, метод getPercentlnstance() значка процентов. Используется локаль для страны по умолчанию – locale. Вы в любой момент можете изменить локализацию для вывода значка валюты. Для работы с локализацией вам нужен класс

Листинг 3.7 – Пример использования NumberForma NumberForma и Locale

```
import java.text.NumberFormat;
import java.util.Locale;
public class NumberFormatDemo1 {
public static void main(String[] args) {
double number = 123.4567;
//определяем локализацию
Locale locFR = new Locale("fr");
Locale.setDefault(Locale.CHINA);
//определяем форматировщик 1
NumberFormat numberFormat1 =
NumberFormat.getInstance();
System.out.println("Число в текущей локали: " +
numberFormat1.format(number));
//определяем форматировщик 2
NumberFormat numberFormat2 =
NumberFormat.getInstance(Locale.US);
System.out.println("Число в китайской локали: " +
numberFormat2.format(number));
//определяем форматировщик 3
NumberFormat numberFormat3 =
NumberFormat.getCurrencyInstance();
System.out.println("Денежная единица в текущей
локали: " + numberFormat3.format(number));
//определяем форматировщик 4
NumberFormat numberFormat4 =
NumberFormat.getCurrencyInstance(Locale.FRANCE);
System.out.println("Денежная единица во французской
локали: " + numberFormat4.format(number));
//определяем форматировщик 5
NumberFormat numberFormat5 =
NumberFormat.getPercentInstance();
```

```
    System.out.println("Процент в текущей локали: " +
numberFormat5.format(number));
    //определяем форматировщик 6
NumberFormat numberFormat6 =
NumberFormat.getPercentInstance(locFR);
    System.out.println("Процент во французской локали: "
+ numberFormat6.format(number));
    }
}
```

Результат работы программы на листинге 3.8 представлен ниже:

```
Число в текущей локали: 123.457
Число в китайской локали: 123.457
Денежная единица в текущей локали: ¥123.46
Денежная единица во французской локали: 123,46 €
Процент в текущей локали: 12,346%
Процент во французской локали: 12 346 %
```

Задания на практическую работу № 3

Задания на Math и Random

1. Создать массив вещественных чисел случайным образом, вывести его на экран, отсортировать его, и снова вывести на экран (использовать два подхода к генерации случайных чисел – метод random() класса Math и класс Random)
2. Создать класс точка Point, описывающий точку на плоскости. Создать Circle класс, в котором одно поле представляет точку – центр окружности, и добавить другие свойства, позволяющие задать точку на плоскости. Создать третий класс Tester который использует для хранения объектов массив объектов Circle и второе поле количество элементов в массиве. Добавить в класс методы, позволяющие найти самую маленькую и самую большую окружность. Добавить в класс метод, упорядочивающий хранение окружностей в массив. Для инициализации полей радиуса и длины окружности используйте класс Random или метод random() класса Math по желанию
3. Создайте массив из 4 случайных целых чисел из отрезка [10;99], выведите его на экран в строку, далее определите и выведите на экран сообщение о том, является ли массив строго возрастающей последовательностью.
4. Пользователь должен ввести с клавиатуры размер массива - натуральное число больше, так чтобы введенное пользователем число сохранялось в переменную n. Если пользователь ввел не подходящее число, то

программа должна просить пользователя повторить ввод. Создать массив из n случайных целых чисел из отрезка [0; n] и вывести его на экран. Создать второй массив только из четных элементов первого массива, если они там есть, и вывести его на экран.

5. Пользователь должен ввести с клавиатуры размер массива - натуральное число больше, так чтобы введенное пользователем число сохранялось в переменную n. Если пользователь ввел не подходящее число, то программа должна просить пользователя повторить ввод. Создать массив из n случайных целых чисел из отрезка [0; n] и вывести его на экран. Создать второй массив только из четных элементов первого массива, если они там есть, и вывести его на экран

Задания на классы Оболочки

Задание 1

1. Создайте объекты класса Double, используя методы valueOf().
2. Преобразовать значение типа String к типу double. Используем метод Double.parseDouble().
3. Преобразовать объект класса Double ко всем примитивным типам.
4. Вывести значение объекта Double на консоль.
5. Преобразовать литерал типа double к строке: String d = Double.toString(3.14);

Задание 2

Заполнить таблицу табл. 3.2 Методы классов оболочек - на пересечении указать x, если данный метод существует у соответствующего класса оболочки.

Таблица 3.2 Методы классов оболочек

	Boolean	Byte	Character	Double	Float	Integer	Long	Short	isStatic
byteValue									
doubleValue									
floatValue									
intValue									
longValue									
shortValue									

Окончание табл. 3.2

	Boolean	Byte	Character	Double	Float	Integer	Long	Short	isStatic
parseXxx									

parseXxx with radix									
valueOf with radix									
toString									
toString(primitive)									
toString(primitive,radix)									

Задание на форматирование строк вывода

1. Создать класс конвертор валют.
2. Создать мини-Приложение интернет-магазин. В приложении рассчитывается стоимость покупки товара, где пользователь может выбрать валюту для оплаты товара
3. Разработать класс Отчет о сотрудниках

   1) Создать класс Employee, у которого есть переменные класса - fullname, salary.
   2) Создать массив, содержащий несколько объектов этого типа.
   3) Создать класс Report со статическим методом generateReport, в котором выводится информация о зарплате всех сотрудников.
   4) Используйте форматирование строк. Пусть salary будет выровнено по правому краю, десятичное значение имеет 2 знака после запятой и можете добавить что-нибудь свое.