Практика 3

1. Ввести с клавиатуры целое число. Вывести на экран его строку-описание следующего вида:
"отрицательное четное число" - если число отрицательное и четное,
"отрицательное нечетное число" - если число отрицательное и нечетное,
"**ноль**" - если число **равно 0**,
"положительное четное число" - если число положительное и четное,
"положительное нечетное число" - если число положительное и нечетное.
Пример для числа 100: положительное четное число
Пример для числа -51: отрицательное нечетное число

2. Ввести с клавиатуры целое число в диапазоне **1** - **999**. Вывести его строку-описание следующего вида:
"четное однозначное число" - если число четное и имеет одну цифру,
"нечетное однозначное число" - если число нечетное и имеет одну цифру,
"четное двузначное число" - если число четное и имеет две цифры,
"нечетное двузначное число" - если число нечетное и имеет две цифры,
"четное трехзначное число" - если число четное и имеет три цифры,
"нечетное трехзначное число" - если число нечетное и имеет три цифры.
Если введенное число не попадает в диапазон **1** - **999**, в таком случае ничего не выводить на экран.
Пример для числа 100: четное трехзначное число
Пример для числа 51: нечетное двузначное число

3. Ввести с клавиатуры **три целых числа**. Вывести на экран количество положительных чисел среди этих трех.
   Примеры:
   а) при вводе чисел -4 6 6
   получим вывод 2
   б) при вводе чисел -6 -6 -3
   получим вывод 0
   в) при вводе чисел 0 1 2
   получим вывод 2

4. Ввести с клавиатуры **три целых числа**. Вывести на экран количество положительных и количество отрицательных чисел в исходном наборе, в следующем виде: "количество отрицательных чисел: а", "количество положительных чисел: б", где а, б - искомые значения. Пример:
   а) при вводе чисел 2 5 6
   получим вывод
   количество отрицательных чисел: 0
   количество положительных чисел: 3
   б) при вводе чисел -2 -5 6
   получим вывод
   количество отрицательных чисел: 2
   количество положительных чисел: 1

5. Ввести с клавиатуры строку и число N больше 0. Вывести на экран строку N раз используя цикл while. Каждое значение с новой строки.
   Пример ввода: абв 2
   Пример вывода:
   абв
   абв

6. Вывести на экран квадрат из **10x10** букв **S** используя цикл while. Буквы в каждой строке не разделять. Пример вывода на экран:
   SSSSSSSSS
   SSSSSSSSS
   SSSSSSSSS
   SSSSSSSSS
   SSSSSSSSS
   SSSSSSSSS
   SSSSSSSSS
   SSSSSSSSS
   SSSSSSSSS
   SSSSSSSSS

7. Вывести на экран таблицу умножения **10x10** используя цикл while. Числа разделить пробелом. Пример вывода на экран:
   1 2 3 4 5 6 7 8 9 10
   2 4 6 8 10 12 14 16 18 20
   3 6 9 12 15 18 21 24 27 30
   4 8 12 16 20 24 28 32 36 40
   5 10 15 20 25 30 35 40 45 50
   6 12 18 24 30 36 42 48 54 60
   7 14 21 28 35 42 49 56 63 70
   8 16 24 32 40 48 56 64 72 80
   9 18 27 36 45 54 63 72 81 90
   10 20 30 40 50 60 70 80 90 100

8. Используя цикл for вывести на экран:
   - горизонтальную линию из **10** восьмёрок
   - вертикальную линию из **10** восьмёрок (символ из горизонтальной строки не учитывается).

9. Ввести с клавиатуры два числа m и n. Используя цикл for вывести на экран прямоугольник размером m на n из восьмёрок.
   Пример: m=2, n=4
   8888
   8888

10. Используя цикл for вывести на экран прямоугольный треугольник из восьмёрок со сторонами **10** и **10**. Пример вывода на экран:
    8
    88
    888
    8888
    88888
    888888
    8888888
    88888888
    888888888
    8888888888

11. Добавьте метод public static void printString(String **s**), в нем напишите код, который будет выводить переданную строку на экран.

```
public class Solution {
   // напишите тут ваш код

   public static void main(String[] args) {
      printString("Hello, Sirius IS!");
   }
}
```

12. Расставить правильно (по другому) **скобки**, чтобы на экран вывелось число **382**

    Последовательность цифр и арифметических операций изменять нельзя.

    Количество круглых скобок должно остаться прежним (2 открывающие и 2 закрывающие).

```
public class Solution {
   public static void main(String[] args) {
      System.out.println((2 * 3) + 4 * 5 + (6 * 7));
   }
}
```

13. Напишите код метода convertEurToUsd, который переводит **евро** в **доллары** по **заданному курсу**. Для возврата результата из метода convertEurToUsd используйте оператор return. **Пример:** return 123*435; Вызовите метод convertEurToUsd дважды в методе main с любыми параметрами. Результаты выведите на экран, каждый раз с новой строки. Подсказка: Расчет выполняется по формуле: долларСША = евро * курс

```
public class Solution {
   public static void main(String[] args) {
      //напишите тут ваш код
   }

   public static double convertEurToUsd(int eur, double course) {
      //напишите тут ваш код
   }
}
```

14. Напишите код addTenPercent, который увеличивает переданное целое число на **10%**. Для возврата результата из метода addTenPercent используйте оператор return. Пример: return 123 * 435;

```
public class Solution {
   public static double addTenPercent(int i) {
      //напишите тут ваш код
   }

   public static void main(String[] args) {
      System.out.println(addTenPercent(9));
   }
}
```

15. Нужно посчитать, сколько литров воды нужно для заполнения бассейна до бортов. Известно, что бассейн имеет линейные размеры a × b × c, заданные в метрах. Эти размеры передаются в метод getVolume. Метод должен вернуть количество литров воды, которое нужно для наполнения бассейна.
    Пример: Метод getVolume вызывается с параметрами **25**, **5**, **2**.
    Пример вывода: 250000

```
public class Solution {
   public static void main(String[] args) {
      System.out.println(getVolume(25, 5, 2));
   }

   public static long getVolume(int a, int b, int c) {
      //напишите тут ваш код
   }
}
```

16. Реализуйте метод public static void writeToConsole(String **s**), который добавляет к началу строки выражение "**printing:** " и выводит измененную строку на экран. Пример вывода для "Hello world!": printing: Hello world!

```
public class Solution {
   public static void main(String[] args) {
      writeToConsole("Hello world!");
   }

   public static void writeToConsole(String s) {
      //напишите тут ваш код

   }
}
```

17. Добавьте метод `public static int convertToSeconds(int hour)` который будет конвертировать часы в секунды. Вызовите его дважды в методе `main` с **любыми параметрами**. Результаты выведите на экран, каждый раз с новой строки.

```
public class Solution {
   //напишите тут ваш код

   public static void main(String[] args) {
      //напишите тут ваш код
   }
}
```

18. Вывести на экран следующий текст - две строки:
    It's Windows path: "C:\Program Files\Java\jdk1.14.0\bin"
    It's Java string: \"C:\\Program Files\\Java\\jdk1.14.0\\bin\"
    Подсказка:
    \" – экранирование двойной кавычки;
    \\ – экранирование обратной косой черты (\).

```
public class Solution {
   public static void main(String[] args) {
      //напишите тут ваш код
   }
}
```

19. Создать **7 объектов**, чтобы на экран вывелись **7 цветов** радуги. Пример вывода: Red Orange Yellow Green Blue Indigo Violet. Каждый объект при создании выводит на экран определенный цвет.

```
public class Solution {
  public static void main(String[] args) {
    //напишите тут ваш код

  }

  public static class Red {
    public Red() {
      System.out.println("Red");
    }
  }

  public static class Orange {
    public Orange() {
      System.out.println("Orange");
    }
  }

  public static class Yellow {
    public Yellow() {
      System.out.println("Yellow");
    }
  }

  public static class Green {
    public Green() {
      System.out.println("Green");
    }
  }

  public static class Blue {
    public Blue() {
      System.out.println("Blue");
    }
  }

  public static class Indigo {
    public Indigo() {
      System.out.println("Indigo");
    }
  }

  public static class Violet {
    public Violet() {
      System.out.println("Violet");
    }
  }
}
```

20. Напишите код метода sumDigitsInNumber(int number). Метод на вход принимает целое трехзначное число. Нужно посчитать сумму цифр этого числа, и вернуть эту сумму.
    Пример: Метод sumDigitsInNumber вызывается с параметром **546**.
    Пример вывода:15

```
public class Solution {
   public static void main(String[] args) {
      System.out.println(sumDigitsInNumber(546));
   }

   public static int sumDigitsInNumber(int number) {
      //напишите тут ваш код

   }
}
```

21. Вводить с клавиатуры **числа**. Если пользователь ввел **-1**, вывести на экран сумму всех введенных чисел и завершить программу. **-1** должно учитываться в сумме.
    Подсказка: один из вариантов решения этой задачи, использовать конструкцию:

```
while (true) {
    int number = считываем число;
    if (проверяем, что number -1)
    break;
    }
```

22. Ввести с клавиатуры **три числа**, вывести на экран **среднее** из них. Т.е. не самое большое и не самое маленькое. Если все числа **равны**, вывести любое из них.

23. Создать класс **Cat**. У кота должно быть имя (name, String), возраст (age, int), вес (weight, int), сила (strength, int).

```
public class Cat {
   //напишите тут ваш код

   public static void main(String[] args) {

   }
}
```

24. Создать 3 объекта класса **Cat**.

```
public class Solution {
   public static void main(String[] args) {
      //напишите тут ваш код
   }

   public static class Cat {
      private String name;
      private int age;
      private int weight;
      private int strength;

      public Cat(String name, int age, int weight, int strength) {
         this.name = name;
         this.age = age;
         this.weight = weight;
         this.strength = strength;
      }
   }
}
```

25. Создать **class Person**. У человека должно быть имя String name, возраст int age, адрес String address, пол char sex.

26. Создать **class Person**. У человека должно быть имя String name, возраст int age, пол char sex. Создайте **геттеры** и **сеттеры** для всех переменных класса **Person**.

27. Реализовать метод boolean fight(Cat **anotherCat**): реализовать механизм драки котов в зависимости от их **веса**, **возраста** и **силы**. Нужно сравнить каждый критерий по отдельности, и победителем будет тот, у которого больше "победивших" критериев. Метод должен определять, выиграли ли мы (this) бой или нет, т.е. возвращать **true**, если выиграли и **false** - если нет.
Если ничья и никто не выиграл, возвращаем либо **true** либо **false**, но **должно выполняться условие:** если cat1.fight(cat2) возвращает true, то cat2.fight(cat1) должен возвращать false.

```
public class Cat {
   public int age;
   public int weight;
   public int strength;

   public Cat() {
   }

   public boolean fight(Cat anotherCat) {
      //напишите тут ваш код
   }

   public static void main(String[] args) {

   }
}
```

28. Создать три кота используя класс **Cat**. Провести **три боя** попарно между котами. Класс **Cat** создавать не надо. Для боя использовать метод boolean fight(Cat **anotherCat**). Результат каждого боя вывести на экран с новой строки.

```
public class Solution {

   public static void main(String[] args) {
      //напишите тут ваш код
   }

   public static class Cat {

      protected String name;
      protected int age;
      protected int weight;
      protected int strength;

      public Cat(String name, int age, int weight, int strength) {
         this.name = name;
         this.age = age;
         this.weight = weight;
         this.strength = strength;
      }

      public boolean fight(Cat anotherCat) {
         int ageScore = Integer.compare(this.age, anotherCat.age);
         int weightScore = Integer.compare(this.weight, anotherCat.weight);
         int strengthScore = Integer.compare(this.strength, anotherCat.strength);

         int score = ageScore + weightScore + strengthScore;
         return score > 0; // return score > 0 ? true : false;
      }
   }
}
```

29. Создать класс **Dog** (собака) с тремя инициализаторами:
    - Имя
    - Имя, рост
    - Имя, рост, цвет

30. Создать класс (**Circle**) **круг**, с тремя инициализаторами:
    - centerX, centerY, radius
    - centerX, centerY, radius, width
    - centerX, centerY, radius, width, color

31. Создать **class Person**. У человека должно быть имя String name, возраст int age. Добавьте метод initialize(String **name**, int **age**), в котором проинициализируйте переменные name и age. В методе main создайте объект Person, занесите его ссылку в переменную person. Вызовите метод initialize с любыми значениями.

```
public class Solution {
   public static void main(String[] args) {
      //напишите тут ваш код
   }

   static class Person {
      //напишите тут ваш код
   }
}
```

32. Изучите внимательно класс **Person**. Исправьте класс так, чтобы только один метод initialize инициализировал все переменные класса **Person**.

```
public class Person {
   String name;
   char sex;
   int money;
   int weight;
   double size;

   public void initialize(String name) {
      this.name = name;
   }

   public void initialize(String name, char sex) {
      this.name = name;
      this.sex = sex;
   }

   public void initialize(String name, int money, char sex) {
      this.name = name;
      this.money = money;
      this.sex = sex;
   }

   public static void main(String[] args) {

   }
}
```

33. Создать класс прямоугольник (**Rectangle**). Его данными будут top, left, width, height (верхняя координата, левая, ширина и высота). Создать для него как можно больше (минимум 4) методов initialize(...)
Примеры:
- заданы **4** параметра: left, top, width, height
- ширина/высота не задана (оба равны 0)
- высота не задана (**равно ширине**) создаём квадрат
- создаём копию другого прямоугольника (он и передаётся в параметрах)
- + свой вариант

34. Создать класс **Cat** (кот) с пятью инициализаторами:
- Имя,
- Имя, вес, возраст
- **Имя, возраст (**вес стандартный)
- **вес, цвет** (имя, адрес и возраст неизвестны, это бездомный кот)
- **вес, цвет, адрес** (чужой домашний кот)

Задача инициализатора - сделать объект **валидным**. Например, если вес неизвестен, то нужно указать какой-нибудь **средний вес**. Кот не может ничего не весить. То же касается возраста и цвета. А вот имени может и не быть (**null**). То же касается адреса: **null**.

35. Создать класс **Friend** (друг) с тремя конструкторами:
    - Имя
    - Имя, возраст
    - Имя, возраст, пол

36. Создать класс **Cat** (кот) с пятью конструкторами:
    - Имя,
    - Имя, вес, возраст
    - **Имя, возраст** (вес стандартный)
    - **вес, цвет**, (имя, адрес и возраст - неизвестные. Кот - бездомный)
    - **вес, цвет, адрес** (чужой домашний кот)

Задача конструктора - сделать объект **валидным**. Например, если вес не известен, то нужно указать какой-нибудь средний вес. Кот не может ничего не весить. То же касательно возраста. А вот имени может и не быть (**null**). То же касается адреса: **null**.

37. Создать класс (**Circle**) круг, с тремя конструкторами:
    - centerX, centerY, radius
    - centerX, centerY, radius, width
    - centerX, centerY, radius, width, color

38. Создать класс прямоугольник (**Rectangle**). Его данными будут top, left, width, height (верхняя координата, левая, ширина и высота). Создать для него как можно больше (минимум 4) методов конструкторов Примеры:
    - заданы **4** параметра: left, top, width, height
    - ширина/высота не задана (оба равны 0)
    - высота не задана (**равно ширине**) создаём квадрат
    - создаём копию другого прямоугольника (он и передаётся в параметрах)
    - + свой вариант

39. В классе **Circle** создать конструктор который проинициализирует все переменные класса. В конструкторе должно быть **три аргумента**.

```
public class Circle {
   public double x;
   public double y;
   public double r;

   //напишите тут ваш код
   public static void main(String[] args) {

   }
}
```

40. Разберитесь, что делает программа. Исправьте **конструктор с двумя параметрами** так, чтобы он вызывал другой конструктор с радиусом **10**. Подумайте, какой конструктор нужно вызвать. Подсказка: внимательно изучите реализацию конструктора по умолчанию.

```
public class Circle {

    public double x;
    public double y;
    public double radius;

    public Circle(double x, double y, double radius) {
        this.x = x;
        this.y = y;
        this.radius = radius;
    }

    public Circle(double x, double y) {
        //напишите тут ваш код
    }

    public Circle() {
        this(5, 5, 1);
    }

    public static void main(String[] args) {
        Circle circle = new Circle();
        System.out.println(circle.x + " " + circle.y + " " + circle.radius);
        Circle anotherCircle = new Circle(10, 5);
        System.out.println(anotherCircle.x + " " + anotherCircle.y + " " + anotherCircle.radius);
    }
}
```

41. Изучите класс **Circle**. Напишите максимальное количество конструкторов с разными аргументами (**минимум 4**). Подсказка: не забудьте про конструктор по умолчанию.

```
public class Circle {
   public double x;
   public double y;
   public double radius;

   //напишите тут ваш код

   public static void main(String[] args) {

   }
}
```

42. Разберитесь, что делает программа. Найдите и исправьте одну ошибку в классе **Circle**. Метод main изменять нельзя. Подсказка: изучите конструктор по умолчанию.

```
public class Circle {
   public Color color;

   public static void main(String[] args) {
      Circle circle = new Circle();
      circle.color.setDescription("Red");
      System.out.println(circle.color.getDescription());
   }

   public void Circle() {
      color = new Color();
   }

   public class Color {
      String description;

      public String getDescription() {
         return description;
      }

      public void setDescription(String description) {
         this.description = description;
      }
   }
}
```

43. По аналогии с классом **Duck** (утка) создайте классы **Cat** (кошка) и **Dog** (собака). Подумайте, что должен возвращать метод toString в классах **Cat** и **Dog**? В методе main создайте по два объекта каждого класса и выведите их на экран. Объекты класса **Duck** уже созданы и выводятся на экран.

```
public class Solution {

   public static void main(String[] args) {
      Duck duck1 = new Duck();
      Duck duck2 = new Duck();
      System.out.println(duck1);
      System.out.println(duck2);

      //напишите тут ваш код
   }

   public static class Duck {
      public String toString() {
         return "Duck";
      }
   }

   //напишите тут ваш код
}
```

44. 1. Внутри класса **Solution** создайте **public static** классы **Man** и **Woman**.
    2. У классов должны быть поля: name (String), age (int), address (String).
    3. Создайте **конструкторы**, в которые передаются все возможные параметры.
    4. Создайте по два **объекта** каждого класса со всеми данными используя конструктор.
    5. Объекты выведите на экран в таком формате: **name + " " + age + " " + address**

45. Создайте классы **Dog**, **Cat**, **Mouse**. Добавьте по **три поля** в каждый класс, на свой выбор. Создайте объекты для героев мультика Том и Джерри. Так много, как только сможете.
    Пример: Mouse jerryMouse = new Mouse("Jerry", 12 , 5), где **12** - высота в см, **5** - длина хвоста в см.

```
public class Solution {
   public static void main(String[] args) {
      Mouse jerryMouse = new Mouse("Jerry", 12, 5);
      //напишите тут ваш код
   }

   public static class Mouse {
      String name;
      int height;
      int tail;

      public Mouse(String name, int height, int tail) {
         this.name = name;
         this.height = height;
         this.tail = tail;
      }
   }
  //напишите тут ваш код
}
```

46. Вводить с клавиатуры **числа** и считать их **сумму**, пока пользователь не введет слово "**сумма**". Вывести на экран полученную сумму. **Подсказка:** реализовать считывание с клавиатуры, пока не будет введена определенная строка, например "exit", можно с помощью следующей конструкции:

```
BufferedReader buffer = new BufferedReader(new InputStreamReader(System.in));
while (true)
{
	String s = buffer.readLine();
	if (s.equals("exit"))
		break;
}
```

47. Текущая реализация: Программа **вводит два числа** с клавиатуры и **выводит минимальное из них** на экран.
Новая задача: Программа вводит пять чисел с клавиатуры и выводит минимальное из них на экран. (В классе должен быть метод public static min, принимающий 5 параметров типа int.)

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
public class Solution {

   public static void main(String[] args) throws Exception {
      BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
      int a = Integer.parseInt(reader.readLine());
      int b = Integer.parseInt(reader.readLine());
      int minimum = min(a, b);
      System.out.println("Minimum = " + minimum);
   }

   public static int min(int a, int b) {
      return a < b ? a : b;
   }
}
```

48. Написать программу, которая:
    1. считывает с консоли число N, которое должно быть больше **0** (Программа не должна ничего выводить на экран, если N меньше либо равно 0.)
    2. потом считывает N чисел с консоли
    3. выводит на экран максимальное из введенных N чисел.

```
public class Solution {
   public static void main(String[] args) throws Exception {
      BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
      int maximum = ;

      //напишите тут ваш код
      System.out.println(maximum);
   }
}
```