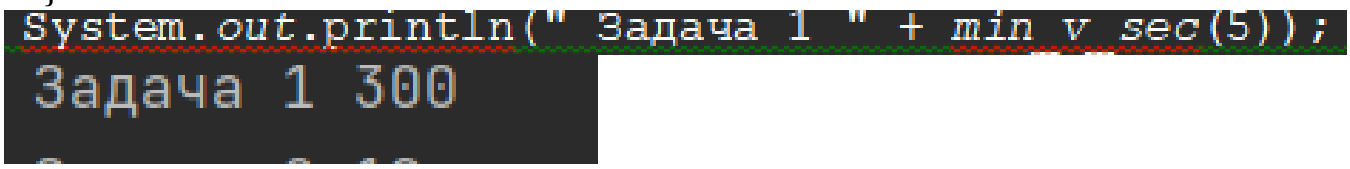
TASKS 1-10

```
public class tasks1 {
    public static void main(String[] args){
        System.out.println(" Задача 1 " + min_v_sec(5));
        System.out.println(" Задача 2 " + points(5,3));
        System.out.println(" Задача 3 " + footballpoints(5,0,2));
        System.out.println(" Задача 4 " + del_na5(-5));
        System.out.println(" Задача 5 " + II(true,false)+ "  "+ II(true,true));
        System.out.println(" Задача 6 " + howManyWalls(88,5,8));
        System.out.println(" Задача 7 " + squaed(7));
        System.out.println(" Задача 8 " + profitableGamble(0.9,3,2));
        System.out.println(" Задача 9 " + frames(10,34));
        System.out.println(" Задача 10 "+ mod(218,5));
    }
```

1. Напишите функцию, которая принимает целое число минут и преобразует его в секунды.

```
    //принимает мин и выдает секунды
    public static int min_v_sec(int min){
        int s = min*60;
        return s;
    }
```



2. Вы подсчитываете очки за баскетбольный матч, учитывая количество забитых 2-х и 3-х очков, находите окончательные очки для команды и возвращаете это значение.

```
    //баскетбольные очки 2-х,3-х броски, вывести общее кол-во
    public static int points(int dva, int tri){
        int k = dva*2 + tri*3;
        return k;
    }
```

3. Создайте функцию, которая принимает количество побед, ничьих и поражений и вычисляет количество очков, набранных футбольной командой на данный момент.

```
    //принимает кол-во побед(3 очка), ничьих(1), пораж(0).общее кол-во возвращает
    public static int footballpoints(int win,int draw,int lose){
        int k = win*3+draw;
```

```
    return k;
}
```

```
System.out.println(" Задача 3 " + footballpoints( win: 5, draw: 0, lose: 2));
```

```
Задача 3 15
```

4. Создайте функцию, которая возвращает true, если целое число равномерно делится на 5, и false в противном случае.

```
// тру, если число расномерно делится на 5. или false
public static boolean del_na5(double a){
    boolean c;
    if (a%5==0){
        c = true;
    }
    else {
        c = false;
    }
    return c;
}
```

```
System.out.println(" Задача 4 " + del_na5( a: -5));
```

```
Задача 4 true
```

5. В Java есть логический оператор &&. Оператор && принимает два логических значения и возвращает true, если оба значения истинны.

```
// оператор &&. принимает 2 лог значения
public static boolean II(boolean a, boolean b){
    boolean c;
    if (a&&b){
        c = true;
    }
    else {
        c = false;
    }
    return c;
}
```

```
System.out.println(" Задача 5 " + II( a: true, b: false)+ "  "+ II( a: true, b: true));
```

```
Задача 5 false  true
```

6. У меня есть ведро с большим количеством темно-синей краски, и я хотел бы покрасить как можно больше стен. Создайте функцию, которая возвращает количество полных стен, которые я могу покрасить, прежде чем мне нужно будет отправиться в магазины, чтобы купить еще.

```
//кол-во полных стен, которые можно покрасить
public static int howManyWalls(int n,int shirina, int visota){
    int k = n/(shirina*visota);
```

```
    return k;
}
```

```
System.out.println(" Задача 6 " + howManyWalls( n: 88, shirina: 5, visota: 8));
```

```
Задача 6 2
```

7. Исправьте код, чтобы решить эту задачу (только синтаксические ошибки). Посмотрите на приведенные ниже примеры, чтобы получить представление о том, что должна делать эта функция.

Код:

```
public class Challenge {
    public static int squaed(int b) {
        return a * a
    }
}
```

```
//исправить код(было int b)
public static int squaed(int a){
    return a*a;
}
```

```
System.out.println(" Задача 7 " + squaed( a: 7));
```

```
Задача 7 49
```

8. Создайте функцию, которая принимает три аргумента prob, prize, pay и возвращает true, если prob * prize > pay; в противном случае возвращает false.

```
// 3 аргумента.тру, если 1*2>3
public static boolean profitableGamble(double prob,double prize, double pay){
    boolean c;
    if (prob * prize > pay){
        c = true;
    }
    else {
        c = false;
    }
    return c;
}
```

```
System.out.println(" Задача 8 " + profitableGamble( prob: 0.9, prize: 3, pay: 2));
```

```
Задача 8 true
```

9. Создайте метод, который возвращает количество кадров, показанных за заданное количество минут для определенного FPS.

```
//кол-во кадров за опред кол-во минут для FPS
public static int frames(int min,int fps){
    return min*60 *fps;
}
```

```
System.out.println(" Задача 9 " + frames( min: 10, fps: 34));
```

```
Задача 9 20400
```

10. Создайте функцию, которая будет работать как оператор модуля % без использования оператора модуля. Оператор модуля-это способ определения остатка операции деления. Вместо того чтобы возвращать результат деления, операция по модулю возвращает остаток целого числа.

```
//оператор модуля % без его использования. возвращает остаток деления
public static int mod(int a, int b){
    return a -(a/b) *b;
}
```

```
System.out.println(" Задача 10 "+ mod( a: 218, b: 5));
```

```
Задача 10 3
```

}

```
public class task_2 {
            public static void main(String[] args) {
                System.out.println(" Задача 1 " + oppositeHouse(1,3));
                System.out.println(" Задача 2 "+ nameShuffle("Sasha
Samarina"));
                System.out.println(" Задача 3 " + discount(1500,50));
                int[] mas = {5,3,-8,0,44,-12};
                System.out.println(" Задача 4 "+ differenceMaxMin(mas));
                System.out.println(" Задача 5 "+ equal(3,4,3)+
equal(1,1,1)+equal(3,4,1));
                System.out.println(" Задача 6  thanks -"+
obratniPoryadok("thanks"));
                System.out.println(" Задача 7 "+ programmers(147,33,526));
                System.out.println(" Задача 8 ooXox - "+ getOX("ooXox")+
"_shcgojcbx - "+ getOX("shcgojcbx"));
                System.out.println(" Задача 9 "+ bomb("lyayluabombaaa")+ "_"+
bomb("vjyhh"));
                System.out.println(" Задача 10 "+ sameAscii("AA","B@") + "_"+
sameAscii("EdAbIt","EDABIT"));


            }
            //1которая принимает номер дома и длину улицы k и возвращает
номер дома на противоположной стороне
            public static int oppositeHouse(int n, int d){ //n- номер нашего
дома , d- длина улице
                                return d*2 -(n-1);
            }

            //2меняет порядок слов в строке
            public static String nameShuffle(String s){
                String[] mas = s.split(" ");//разделяет строку через пробелы
и добавляет в массив
                StringBuilder op = new StringBuilder();//создает объект
класса, где будет обратный порядрк
                for (int i = mas.length - 1; i >= 0; i--) {
                    op.append(mas[i]).append(" ");//добавляет в обратном
порядке
                }
                return op.toString();

            }
            //3исходная цена и процент скидки - возвращает цену после скидки
            public static double discount(int s,int p){
                return  s - s/100*p;
```

```
    }//4принимает массив и возвращает разницу макс и мин чисел
    public static int differenceMaxMin(int arr[]){
        int max = arr[0];//первый элемент исходного массива
        int min = arr[0];
        for (int i = 0; i < arr.length; i++){  //нахождение макс и
мин значений
            if (arr[i] > max ){
                max = arr[i];
            }
            if (arr[i] < min ){
                min = arr[i];
            }
        }
        return max - min;
    }
    //5возвращает кол-во одинаковых чисел
    public static int equal(int a,int b, int c){
        int k = 0;
        if (a == b) //для 3 и 0
            k++;
        if (a==c)
            k++;
        if (b==c)
            k++;
        if (((a == c) & (a != b)) || ((b == c) & (a != b)) || ((a ==
b) & (a != c))) k++;//чтобы учесть 2 числа
        return k;
    }
    //6строку в обратном порядке
    public static String obratniPoryadok (String a){
        StringBuilder aa = new StringBuilder(a);//создание объекта на
основе строки (работает быстрее)
        return  aa.reverse().toString(); // метод меняет порядок всех
символов на противоп.

        //создать объект StringBuilder с нашим параметром,
переворачивание строки
    }
    //7 3 зарплаты, найти разницу между самой высокопл и нет
    public static int programmers (int a,int b,int c){
        int[] mas = {a,b,c}; //носим значения в массив в порядке
возрастания
        Arrays.sort(mas);//сортируем массив
        int p = mas[2]-mas[0];//вычисляем разницу между бол и мен
        return  p;
    }
    /* 8 проверяет одинаково ли кол-во о и х
    тру, если кол-во одинаково
    если символов нет, то тру - т.к. изначально они равны,
    а в цикле не измен -  выполняется
     */
    public static boolean getOX(String s){
        int kolX = 0;
        int kol0 = 0;
        char[] mas= s.toCharArray(); //преобразует в посимвольный
массив, длина массива равна длине строки
        for (int a : mas){  //улучш цикл фор (объявление : выражение)
            if (a == 'x'||a =='X' )
                kolX++;
            if (a =='o'||a=='0')
                kol0++;
        }
        return kol0==kolX;
    }
```

```
            //9 если "бомба" в строке, ответить
            public static String bomb(String s){
                String text;
                if(s.contains("bomb") || s.contains("BOMB")) //метод чтобы
проверить, содержит ли String
                    // указанную последовательность символов
                    text = "ПРИГНИСЬ!";
                else
                    text = "Расслабься, бомбы нет";
                return text;
                }
                //10 сумма значений ASCII СОВПАДАЕТ ИЛИ НЕТ
            public static boolean sameAscii (String car1, String har2){
                int Sum1=0;
                int Sum2=0;
                char[] ascii1 = car1.toCharArray();//записывает в массив
посимвольно
                char[] ascii2 = har2.toCharArray();
                for (char c : ascii1) {
                    Sum1 += c;
                }
                for (char c : ascii2) {
                    Sum2 += c;
                }
                return Sum1==Sum2;
            }
```

```
import java.io.PrintStream;
import java.util.Arrays;

public class TASK_3 {
    public TASK_3() {
    }

    public static void main(String[] args) {
        System.out.println(" Задача 1 ");
        millionsRounding(new String[][]{{"Manila", "7323452"}, {"Kuala
Lumpur", "796"}, {"Jakarta", "770478"}});
        System.out.println(" Задача 2 " + Arrays.toString(otherSides(4)));
        PrintStream var10000 = System.out;
        String var10001 = rps("камень", "бумага");
        var10000.println(" Задача 3 " + var10001 + ";" + rps("камень",
"камень"));
        int[] var1 = new int[]{2, 8, 7, 5};
        System.out.println(" Задача 4 " + war(var1));
        System.out.println(" Задача 5 исходная: Fjf6b5bBBbbBTR, перевернутая:
" + revCase("Fjf6b5bBBbbBTR"));
        System.out.println(" Задача 6 " + inat("Mozz"));
        System.out.println(" Задача 7 " + doesBrickFit(1, 2, 2, 1, 1));
        System.out.println(" Задача 8 " + totalDistance(70.0D, 7.0D, 0,
false));
        var1 = new int[]{2, 3, 2, 3};
        System.out.println(" Задача 9 " + mean(var1));
        System.out.println(" Задача 10 " + parityAnalysis(243));
    }

    public static void millionsRounding(String[][] arr) {
        for(int i = 0; i < arr.length; ++i) {
            double chislo =
(double)Math.round((double)Integer.parseInt(arr[i][1]) / Math.pow(10.0D,
6.0D)) * Math.pow(10.0D, 6.0D);
```

```
            arr[i][1] = Double.toString(chislo);
        }

        System.out.println(Arrays.deepToString(arr));
    }

    public static double[] otherSides(int side1) {
        double side2 = Math.ceil((double)(side1 * 2 * 100)) / 100.0D;
        double side3 = Math.ceil((double)side1 * Math.sqrt(3.0D) * 100.0D) /
100.0D;
        return new double[]{side2, side3};
    }

    public static String rps(String a, String b) {
        if (a.equals("камень") && b.equals("ножницы")) {
            return "Игрок 1 выигрывает ";
        } else if (a.equals("камень") && b.equals("бумага")) {
            return "Игрок 2 выигрывает ";
        } else if (a.equals("ножницы") && b.equals("камень")) {
            return "Игрок 2 выигрывает ";
        } else if (a.equals("ножницы") && b.equals("бумага")) {
            return "Игрок 1 выигрывает ";
        } else if (a.equals("бумага") && b.equals("камень")) {
            return "Игрок 1 выигрывает ";
        } else {
            return a.equals("бумага") && b.equals("ножницы") ? "Игрок 2
выигрывает " : "НИЧЬЯ";
        }
    }

    public static int war(int[] arr) {
        int chet = 0;
        int nechet = 0;

        for(int i = 0; i < arr.length; ++i) {
            if (arr[i] % 2 == 0) {
                chet += arr[i];
            } else {
                nechet += arr[i];
            }
        }

        return Math.abs(chet - nechet);
    }

    public static String revCase(String str) {
        char[] chars = str.toCharArray();

        for(int i = 0; i < chars.length; ++i) {
            char c = chars[i];
            if (Character.isUpperCase(c)) {
                chars[i] = Character.toLowerCase(c);
            } else if (Character.isLowerCase(c)) {
                chars[i] = Character.toUpperCase(c);
            }
        }

        return new String(chars);
    }

    public static String inat(String st) {
        String s = null;
        char[] strToArray = st.toCharArray();
        int len = strToArray.length * 1000;
```

```
        char[] vowels = new char[]{'a', 'e', 'i', 'o', 'u', 'y'};
        char[] var5 = vowels;
        int var6 = vowels.length;

        for(int var7 = 0; var7 < var6; ++var7) {
            char i = var5[var7];
            if (i == strToArray[strToArray.length - 1]) {
                s = st + "-inator " + len;
            } else {
                s = st + "inator " + len;
            }
        }

        return s;
    }

    public static boolean doesBrickFit(int a, int b, int c, int w, int h) {
        return a * b <= w * h || a * c <= w * h || b * c <= w * h;
    }

    public static double totalDistance(double top, double ras, int pass,
boolean cond) {
        ras += (double)pass * ras * 0.05D;
        if (cond) {
            ras *= 1.1D;
        }

        return top / ras * 100.0D;
    }

    public static double mean(int[] mass) {
        return (double)Arrays.stream(mass).sum() / (double)mass.length;
    }

    public static boolean parityAnalysis(int a) {
        int sum = 0;

        int b;
        for(b = a; a > 0; a /= 10) {
            sum += a % 10;
        }

        return b % 2 == sum % 2;
    }
}
```

```
import java.util.Arrays;
public class Task4 {

        public static void main(String[] args) {
            System.out.println(" Задача 1 " + sevenBoom(new int[]
{2,3,5,6,7}));
            System.out.println(" Задача 2 " + cons(new int[] {5,4,6}));
            System.out.println(" Задача 3 " + unmix("emwo  emwo"));
            System.out.println(" Задача 4 " + noYelling("I just!!! can!!!
```

```
not!!! believe!!! it!!!"));
            System.out.println(" Задача 5 " + xPronounce("OMG x box unboxing
video x D"));
            System.out.println(" Задача 6 " + largestGap(new int[]
{2,56,43,6,77,1}));
            System.out.println(" Задача 7 " + obrfun(7977));
            System.out.print  (" Задача 8 "); commonLast("Watch the
characters dance!");
            System.out.println(" Задача 9 " + memeSum(25,77));
            System.out.println(" Задача 10 " + unrepeated("wwwee!!"));

        }
        //1 принимает массив чисел и возвращает "Бум!",
    // если в массиве появляется цифра 7.
        public static String sevenBoom(int[] seven) {
            int a = 0;
            String s;
            for (int i : seven) {
                s = Integer.toString(i);
                if (s.contains("7")) {
                    a += 1;
                }
            }
            if (a > 0)
                return "Boom!";
            else
                return "there is no 7 in the array";
        }
        // 2 функцию, которая определяет, могут ли элементы в массиве быть
переупорядочены,
    // чтобы сформировать  последовательный список чисел, где каждое
    // число появляется ровно один раз.
     public static boolean cons(int[] mas){
            Arrays.sort(mas);
            int a =0;
            for (int i =1; i < mas.length;i++){
                if (mas[i] - mas[i-1] ==1){
                                    }
                else a++;
            }
            if (a>0)
                return false;
            else
                return true;
     }
     // 3 все строки перепутались, каждая пара символов поменялась местами.
    public static String unmix(String s){
            char a;
            char[] mas = s.toCharArray();
            for (int i =1; i < mas.length;i++){
                if(i%2!=0){  //каждый 2 эл
                    a =mas[i];
                    mas[i] = mas[i-1];
                    mas[i-1]=a;
                }
            }
            return s=String.valueOf(mas);//возвращает строковое представление
int аргумента
    }
    // 4 преобразует предложения, заканчивающиеся несколькими ?? или !!!
    // в предложение, заканчивающееся только одним, без изменения пунктуации
    // в середине предложений.
    public static String noYelling(String yell){
        while (yell.endsWith("!!") || yell.endsWith("??")) {
```

```
            yell = yell.substring(0, yell.length() - 1); // удаление
последнего элемента строки
        }
        return yell;
    }
    /// 5 Замените все x на "cks", ЕСЛИ ТОЛЬКО:
    //Слово начинается с "x", поэтому замените его на "z".
    //Слово-это просто буква "x", поэтому замените ее на "ecks "
    public static String xPronounce(String x) {
        x = x.replaceAll("\sx\s", " ecks ");
        x = x.replaceAll("\sx", " z");
        x = x.replaceAll("x", "cks");
        return x;
    }
 // 6  Учитывая массив целых чисел,
    // верните наибольший разрыв между отсортированными элементами массива
    public static int largestGap(int[] mas){
        Arrays.sort(mas);
        int k = 0;
        for (int i =1; i < mas.length;i++){
            if(mas[i]-mas[i-1] >k){
                k =mas[i]-mas[i-1];
            }
        }
        return k;
    }
    // 7 создать функцию, которая при подаче
    // входных данных ниже производит показанные примеры выходных данных
из исходного числа вычитаем это же число отсортированное по возрастанию
    // 832 → 594
    // 51 → 36
    //7977 → 198
    //1 → 0
    //665 → 99
    //149 → 0
public static int obrfun(int a) {
            String d = String.valueOf(a); //пер в стр, в сим
            char[] s = d.toCharArray();
            Arrays.sort(s);
            int b = Integer.parseInt(String.valueOf(s));
            return a - b;
}
// 8 8.    Создайте функцию, которая принимает предложение в качестве
// входных данных и возвращает наиболее распространенную последнюю
// гласную в предложении в виде одной символьной строки.
public static void  commonLast(String str){
            String gl="aoeuyi";
            str = str.replace('.',' ');
    str = str.replace('.',' ');
    str = str.replace(',',' ');
    str = str.replace('!',' ');
    str = str.replace('?',' ');
    str = str.replaceAll("  ","");
    String[] words = str.split(" "); //отд сл по проб
    int[] mas = new int[6];
    int k=0;int kmax=0; int max=0;
       for (String w:words){
        k = gl.indexOf(w.charAt(w.length()-1));
        if(k!=-1){
            mas[k]+=1;
            if(mas[k]>max){
                max=mas[k];
                kmax=k;
```

```
            }
        }
    }
    if(max!=0)
        System.out.println(gl.charAt(kmax));

}


    // 9 складывать числа
    // 248
   // +208
    // 4416
   public static int memeSum(int a, int b) {
       String sum = "";
       int len = Math.max(String.valueOf(a).length(),
String.valueOf(b).length());//бол длина числа
       for (int i = len - 1; i >= 0; i--) {
           sum = Integer.toString(a % 10 + b % 10) + sum;
           a /= 10;
           b /= 10;
       }
       return Integer.parseInt(sum);
   }
   // 10 удаляет люб повторяющиеся символы
       public static String unrepeated(String s){
            String res="";
           for (int i =0; i < s.length();i++){
               if (!res.contains(String.valueOf(s.charAt(i)))) //добавляем
только новые символы
                   res +=s.charAt(i);
           }
           return res;

       }
}
```

```
import java.text.SimpleDateFormat;
import java.util.*;
import java.util.regex.Pattern;
public class Task5 {
    public static void main(String[] args){
        System.out.println("Задание 1 = " + sameLetterPattern("FFGG",
"CDCD"));
        System.out.println("Задание 2 = " + spiderVsFly("H3", "E2"));
        System.out.println("Задание 3 = " + digitsCount(677));
        System.out.println("Задание 4 = " + totalPoints(new String[] {"cat",
"create", "sat"}, "caster"));
        System.out.println("Задание 5 = " + longestRun(new int[] {1, 2, 3,72,
11, 15}));
        System.out.println("Задание 6 = " + takeDownAverage(new
String[]{"95%"}));
        System.out.println("Задание 7 = " + rearrange("Tesh3 th5e 1I lov2e
way6 she7 j4ust i8s."));
        System.out.println("Задание 8 = " + maxPossible(9328, 456));
        System.out.println("Задание 9 = " + timeDifference("Moscow", "July
31, 1983 23:01", "Rome"));
        System.out.println("Задание 10 = " + isNew(3));
    }
    //1  возвращает true, если две строки имеют один и тот же буквенный
шаблон
    public static boolean sameLetterPattern(String a, String b) {
```

```
        if (a.length() != b.length()) return false;
        HashMap<Character, Character> match = new HashMap<>();
        for (int i = 0; i < a.length(); i++) {
            if (!match.containsKey(a.charAt(i)))
                match.put(a.charAt(i), b.charAt(i));  ///заносим
            if (b.charAt(i) != match.get(a.charAt(i)))
                return false;
        }
        return true;
    }
//2
    public static String spiderVsFly(String s, String t) {
        //сумма номеров колец
        // разница между кольцами + разница между ветками * длина пути на
кольце
        // длина пути на кольце = 2 * x * sin(45 / 2) = x * 0.76536686473
        int sx = s.charAt(0) - 65;
        int sy = s.charAt(1) - 48;
        int fx = t.charAt(0) - 65;
        int fy = t.charAt(1) - 48;
        double strategyDist1 = sy + fy;
        double strategyDist2 = Math.abs(sy - fy) + ((sx + fx) % 8) * fy *
0.76536686473;
        String path = "";
        if (strategyDist1 <= strategyDist2) {
            for (int i = 0; i < sy; i++) {
                path += s.charAt(0);
                path += sy - i;
                path += '-';
            }
            path += "A0-";
            for (int i = 0; i < fy; i++) {
                path += t.charAt(0);
                path += i + 1;
                path += '-';
            }
        } else {
            for (int i = 0; i < Math.abs(sy - fy); i++) {
                path += s.charAt(0);
                if (sy > fy) path += sy - i;
                else path += sy + i;
                path += '-';
            }
            for (int i = 0; i <= (sx + fx) % 8; i++) {
                path += (char)(65 + (sx + i) % 8);
                path += t.charAt(1);
                path += '-';
            }
        }
        return path.substring(0, path.length() - 1);
    }
//3 рекурсивно подсчитывать количество цифр числа.
// Преобразование числа в строку не допускается

    public static int digitsCount2(long number) {
        if (number == 0) return 0;
        return 1 + digitsCount2(number / 10);
    }
    public static int digitsCount(long number) {
        return 1 + digitsCount2(number / 10);
    }

    //4  принимает в массив уже угаданных слов расшифрованное 6-буквенное
    // слово и возвращает общее количество очков, набранных игроком в
```

```
определенном раунде,
    public static int[] getCharset(String word) {
        int[] charset = new int[127];
        for (char c : word.toCharArray())
            charset[c]++;
        return charset;
    }
    public static int totalPoints(String[] words, String scramble) {
        int points = 0;
        int[] scrambleCharset = getCharset(scramble);
        for (int i = 0; i < words.length; i++) {
            int[] wordCharset = getCharset(words[i]);
            boolean good = true;
            for (int j = 0; j < 127; j++)
                if (wordCharset[j] > scrambleCharset[j]) {
                    good = false;
                    break;
                }
            if (good) {
                points += words[i].length() - 2;
                if (words[i].length() == 6) points += 50;
            }
        }
        return points;
    }
    // 5.  Последовательный прогон-это список соседних последовательных целых
чисел.
    // Этот список может быть как увеличивающимся, так и уменьшающимся.
    // Создайте функцию, которая принимает массив чисел и возвращает длину
    // самого длинного последовательного запуска

    public static int longestRun(int[] arr) {
        int max = 1;
        int cur = 1;
        for (int i = 0; i < arr.length - 1; i++)
            if (arr[i+1] - arr[i] == 1 || arr[i+1] - arr[i] == -1) {
                cur++;
                if (max < cur)
                    max = cur;
            } else cur = 1;
        return max;
    }
    // 6   Какой процент вы можете набрать на тесте, который в одиночку
снижает
    // средний балл по классу на 5%? Учитывая массив оценок ваших
одноклассников,
    // создайте функцию, которая возвращает ответ. Округлите до ближайшего
процента.

    public static String takeDownAverage(String[] percents) {
        int avg = 0;
        for (String s : percents)
            avg += Integer.parseInt(s.substring(0, s.length() - 1));
        return (avg / percents.length - percents.length*5 - 5) + "%";
    }
    // 7   Учитывая предложение с числами, представляющими расположение
слова,
    // встроенного в каждое слово, верните отсортированное предложение

    public static String rearrange(String str) {
        String[] words = str.split(" ");
        String[] res = new String[words.length];
        for (String word : words) {
            for (int i = 0; i < word.length(); i++) {
```

```
                if (Character.isDigit(word.charAt(i))) {
                    res[word.charAt(i) - 48 - 1] = word.substring(0, i) +
word.substring(i+1);
                    break;
                }
            }
        }
        return String.join(" ", res);
    }
    //8   функцию, которая делает первое число как можно больше,
    // меняя его цифры на цифры во втором числе.

    public static Integer[] splitNumber(int n) {
        ArrayList<Integer> res = new ArrayList<>();
        while (n > 0) {
            res.add(n % 10);
            n /= 10;
        }
        return res.toArray(new Integer[res.size()]);
    }
    public static int maxPossible(int n, int r) {
        Integer[] num = splitNumber(n);
        Integer[] rnum = splitNumber(r);
        Arrays.sort(rnum, Collections.reverseOrder());
        int rnumindex = 0;
        for (int i = num.length - 1; i >= 0; i--) {
            if (num[i] < rnum[rnumindex]) {
                num[i] = rnum[rnumindex];
                rnumindex++;
            }
            if (rnumindex == rnum.length)
                break;
        }
        n = 0;
        int dec = 1;
        for (int i : num) {
            n += i * dec;
            dec *= 10;
        }
        return n;
    }
    // 9 вычислить, сколько времени сейчас в двух разных городах. Вам дается
строка cityA
    // и связанная с ней строка timestamp (time in cityA) с датой,
отформатированной
    // в полной нотации США, как в этом примере:
        //"July 21, 1983 23:01"
    //Вы должны вернуть новую метку времени с датой и соответствующим
временем в cityB, отформатированную как в этом примере:
       //"1983-7-22 23:01"


    public static SimpleDateFormat parseDate = new SimpleDateFormat("MMMM d,
yyyy HH:mm");
    public static SimpleDateFormat formatDate = new SimpleDateFormat("yyyy-M-
d HH:mm");
    public static String getGMT(String city) {
        if (city == "Los Angeles") return "GMT-08:00";
        if (city == "New York") return "GMT-05:00";
        if (city == "Caracas") return "GMT- 04:30";
        if (city == "Buenos Aires") return "GMT-03:00";
        if (city == "London") return "GMT00:00";
        if (city == "Rome") return "GMT+01:00";
        if (city == "Moscow") return "GMT+03:00";
```

```
        if (city == "Tehran") return "GMT+03:30";
        if (city == "New Delhi") return "GMT+05:30";
        if (city == "Beijing") return "GMT+08:00";
        if (city == "Canberra") return "GMT+10:00";
        return "GMT";
    }
    public static String timeDifference(
            String cityA, String timestamp, String cityB
    ) {
        try {
            parseDate.setTimeZone(TimeZone.getTimeZone(getGMT(cityA)));
            formatDate.setTimeZone(TimeZone.getTimeZone(getGMT(cityB)));
            Date date = parseDate.parse(timestamp);
            return formatDate.format(date);
        } catch(Exception e) {}
        return null;
    }

    // 10 Новое число-это число, которое не является перестановкой любого
меньшего числа.
//509-это новое число, потому что оно не может быть образовано перестановкой
любого меньшего числа
// (ведущие нули не допускаются
    public static boolean isNew(int n) {
        Integer[] num = splitNumber(n); //из 8 задачи
        for (int i = 0; i < num.length - 1; i++)
            if (num[i] > 0 && num[i] < num[num.length - 1])
                return false;
        return true;
    }
}
```

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashSet;
import java.util.List;
import java.util.stream.Collectors;
public class Task6 {
    public static void main(String[] args) {

        System.out.println("Answer 1 = " + hiddenAnagram("My world evolves in
a beautiful space called Tesh.",
                "sworn love lived"));
        System.out.println("Answer 2 = " +
collect("intercontinentalisationalism", 6));
        System.out.println("Answer 3 = " + nicoCipher("myworldevolvesinhers",
"tesh"));
        System.out.println("Answer 4 = " + Arrays.toString(twoProduct(new
int[]{1, 2, 3, 9, 4, 15, 3, 5}, 45)));
        System.out.println("Answer 5 = " + Arrays.toString(isExact(6)));
        System.out.println("Answer 6 = " + fractions("0.(6)"));
        System.out.println("Answer 7 = " + pilish_string("33314444"));
        System.out.println("Answer 8 = " + generateNonconsecutive(1));
        System.out.println("Answer 9 = " + isValid("aabbcd"));
        int[] var10000 = new int[]{1, 6, 5, 4, 8, 2, 3, 7};
        System.out.println("Answer 10 = " +
Arrays.deepToString(sumsUp(var10000)));

    }
```

```
    /* 1. Создайте функцию, которая принимает две строки. Первая строка
содержит
       предложение, содержащее буквы второй строки в последовательной
       последовательности, но в другом порядке. Скрытая анаграмма должна
содержать
       все буквы, включая дубликаты, из второй строки в любом порядке и не
должна
       содержать никаких других букв алфавита.
       Напишите функцию, чтобы найти анаграмму второй строки, вложенную где-
то в
       первую строку. Вы должны игнорировать регистр символов, любые пробелы
и
       знаки препинания и возвращать анаграмму в виде строчной строки без
пробелов
       или знаков препинания.
       Примечание:
       - Совершенная анаграмма содержит все буквы оригинала в любом порядке,
ни
       больше, ни меньше.
       - Если в предложении нет скрытой анаграммы, верните "noutfond".
       - Как и в приведенных выше примерах, скрытая анаграмма может
начинаться или
       заканчиваться частично через слово и/или охватывать несколько слов и
может
       содержать знаки препинания и другие не-альфа-символы.
       - Игнорируйте регистр символов и любые встроенные не-альфа-символы.
       - Если в предложении больше 1 результата, верните ближайший к началу.
*/

    public static int[] getLetterSet(String str) {
        int[] set = new int[26];
        for (char c : str.toCharArray())
            set[c - 97]++;
        return set;
    }

    public static String onlyLetters(String str) {
        str = str.toLowerCase();
        String res = "";
        for (char c : str.toCharArray())
            if (97 <= c && c <= 122)
                res += c;
        return res;
    }

    public static String hiddenAnagram(String a, String b) {
        a = onlyLetters(a);
        b = onlyLetters(b);
        int[] setB = getLetterSet(b);
        for (int i = 0; i <= a.length() - b.length(); i++) {
            String substr = a.substring(i, i+b.length());
            int[] setA = getLetterSet(substr);
            if (Arrays.equals(setA, setB)) {
                String res = "";
                for (char c : substr.toCharArray())
                    if (97 <= c && c <= 122)
                        res += c;
                return res;
            }
        }
        return "noutfond";
    }

    /* 2. Напишите функцию, которая возвращает массив строк, заполненных из
```

```
срезов
       символов n-длины данного слова (срез за другим, в то время как n-длина
       применяется к слову).
       Примечания:
       - Убедитесь, что результирующий массив лексикографически упорядочен.
       - Возвращает пустой массив, если заданная строка меньше n.
       - Ожидается, что вы решите эту задачу с помощью рекурсивного подхода.
*/


    public static List<Object> collect(String str, int n) {
        List<Object> lst = new ArrayList();
        if (str.length() < n) {
            return lst;
        } else {
            lst.add(str.substring(0, n));
            lst.addAll(collect(str.substring(n), n));
            List<Object> list =
(List)lst.stream().sorted().collect(Collectors.toList());
            return list;
        }
    }

    /* 3. В шифре Nico кодирование осуществляется путем создания цифрового
ключа и
       присвоения каждой буквенной позиции сообщения с помощью
предоставленного
       ключа.
       Создайте функцию, которая принимает два аргумента, message и key, и
возвращает
       закодированное сообщение.
       Существуют некоторые вариации правил шифрования. Одна из версий правил
       шифрования изложена ниже:
       message = "mubashirhassan"
       key = "crazy"
       nicoCipher(message, key) → "bmusarhiahass n"
       Шаг 1: Назначьте числа отсортированным буквам из ключа:
       "crazy" = 23154
       Шаг 2: Назначьте номера буквам данного сообщения:
        2 3 1 5 4
        ---------
        m u b a s
        h i r h a
        s s a n
       Шаг 3: Сортировка столбцов по назначенным номерам:
        1 2 3 4 5
        ---------
        b m u s a
        r h i a h
        a s s   n
       Шаг 4: Верните закодированное сообщение по строкам:
       eMessage = "bmusarhiahass n" */

    public static int[] getCharset(String word) {
        int[] charset = new int[127];
        for (char c : word.toCharArray()) charset[c]++;
        return charset;
    }

    public static String nicoCipher(String message, String key) {
        for (int i = 0; i < (message.length()+key.length()) % key.length();
i++)
            message += ' ';
        int[] set = getCharset(key);
```

```
        int[] setCount = set.clone();
        int counter = 1;
        for (int i = 0; i < set.length; i++)
            if (set[i] != 0) {
                if (set[i] > 1)
                    counter += set[i] - 1;
                set[i] = counter++;
            }
        int[] offsets = new int[key.length()];
        for (int i = 0; i < key.length(); i++)
            offsets[set[key.charAt(i)]-setCount[key.charAt(i)]--] = i;
        String res = "";
        for (int i = 0; i < message.length(); i++) {
            int index = (i / offsets.length) * offsets.length + offsets[i %
offsets.length];
            res += message.charAt(index);
        }

        return res;
    }

    /* 4. Создайте метод, который принимает массив arr и число n и возвращает
массив из
       двух целых чисел из arr, произведение которых равно числу n следующего
вида:
       [value_at_lower_index, value_at_higher_index]
       Убедитесь, что вы возвращаете массив из двух целых чисел, который
точно делит n
       (произведение n) и что индексы совпадают с указанным выше форматом.
Таким образом,
       сортировка значений основана на восходящих индексах.
       Примечание:
       - Дубликатов не будет.
       - Возвращает пустой массив, если совпадение не найдено.
       - Всегда считайте, что пара рассматриваемого элемента (текущий
элемент, на который
       указывали во время поиска) находится слева от него.
       - Массив может иметь несколько решений (произведений n), поэтому
возвращайте первое
       найденное полное совпадение (как описано выше). */

    public static int[] twoProduct(int[] arr, int n) {
        HashSet<Integer> set = new HashSet<>();
        for (int m : arr) {
            if (n % m == 0 && set.contains(n / m))
                return new int[] {n/m, m};
            set.add(m);
        }
        return new int[] {};
    }

    /* 5. Создайте рекурсивную функцию, которая проверяет, является ли число
точной
       верхней границей факториала n. Если это так, верните массив точной
       факториальной границы и n, или иначе, пустой массив.
       Примечание:
       - Ожидается, что вы решите эту задачу с помощью рекурсии. */

    public static int[] isExact(int f, int m, int n) {
        if (f < n)
            return isExact(f*(m+1), m+1, n);
        return new int[] {f, m};
    }
```

```
    public static int[] isExact(int n) {
        int[] res = isExact(1, 1, n);
        if (res[0] == n) return res;
        return new int[] {};
    }

    /* 6. Деление на дробь часто приводит к бесконечно повторяющейся
десятичной дроби.
       1/3=.3333333... 1/7=.142857142857...
       Создайте функцию, которая принимает десятичную дробь в строковой форме
с
       повторяющейся частью в круглых скобках и возвращает эквивалентную
дробь в
       строковой форме и в наименьших членах. */

    public static String fractions(String frac) {
        int startBracket = frac.indexOf('(');
        if (startBracket != -1) {
            String f = frac.substring(startBracket+1, frac.length()-1);
            frac = frac.substring(0, startBracket);
            for (int i = 0; i < 9 - f.length(); i++)
                frac += f;
        }
        double a = Double.parseDouble(frac);
        int div = 2;
        while (Math.ceil(a * div) - a * div > 0.000001) div++;
        return "" + (int)Math.ceil(a * div) + "/" + div;
    }


    /* 7. В этой задаче преобразуйте строку в серию слов (или
последовательности
       символов), разделенных одним пробелом, причем каждое слово имеет
одинаковую
       длину, заданную первыми 15 цифрами десятичного представления числа Пи:
       3.14159265358979
       Если строка содержит больше символов, чем общее количество, заданное
суммой цифр
       Пи, неиспользуемые символы отбрасываются, и вы будете использовать
только те,
       которые необходимы для формирования 15 слов.
       String =

"HOWINEEDADRINKALCOHOLICINNATUREAFTERTHEHEAVYLECTURESINVOLVINGQUANTUMME
       CHANICSANDALLTHESECRETSOFTHEUNIVERSE"
       Pi String = "HOW I NEED A DRINK ALCOHOLIC IN NATURE AFTER THE HEAVY
       LECTURES INVOLVING QUANTUM MECHANICS"
       // Every word has the same length of the digit of Pi found at the same
       index ?
       // "HOW" = 3, "I" = 1, "NEED" = 4, "A" = 1, "DRINK" = 5
       // "ALCOHOLIC" = 9, "IN" = 2, "NATURE" = 6, "AFTER" = 5
       // "THE" = 3, "HEAVY" = 5, "LECTURES" = 8, "INVOLVING" = 9
       // "QUANTUM" = 7, "MECHANICS" = 9
       // 3.14159265358979
       Также, если строка содержит меньше символов, чем общее количество,
заданное суммой
       цифр Пи, в любом случае вы должны соблюдать последовательность цифр
для получения
       слов.
       String = "FORALOOP"
       Pi String = "FOR A LOOP"
       // Every word has the same length of the digit of Pi found at the same
       index ?
       // "FOR" = 3, "A" = 1, "LOOP" = 4
```

```
       // 3.14
       Если буквы, содержащиеся в строке, не совпадают в точности с цифрами,
в этом случае
       вы будете повторять последнюю букву, пока слово не будет иметь
правильную длину.
       String = "CANIMAKEAGUESSNOW"
       Pi String = "CAN I MAKE A GUESS NOWWWWWWW"
       // Every word has the same length of the digit of Pi found at the same
       index ?
       // "CAN" = 3, "I" = 1, "MAKE" = 4, "A" = 1, "GUESS" = 5, "NOW" = 3
       // 3.14153 (Wrong!)
       // The length of the sixth word "NOW" (3)...
       // ...doesn't match the sixth digit of Pi (9)
       // The last letter "W" will be repeated...
       // ...until the length of the word will match the digit
       // "CAN" = 3, "I" = 1, "MAKE" = 4, "A" = 1, "GUESS" = 5, "NOWWWWWWW" =
       9
       // 3.14159 (Correct!)
       Если данная строка пуста, должна быть возвращена пустая строка.
       Учитывая строку txt, реализуйте функцию, которая возвращает ту же
строку,
       отформатированную в соответствии с приведенными выше инструкциями.
       Примечание:
       - Эта задача представляет собой упрощенную концепцию, вдохновленную
Пилишем,
       своеобразным типом ограниченного письма, в котором используются все
известные
       возможные цифры Пи. Потенциально бесконечный текст может быть написан
с
       использованием знаков препинания и набора дополнительных правил,
которые позволяют
       избежать длинных последовательностей маленьких цифр, заменяя их
словами больше 9
       букв и делая так, чтобы композиция выглядела более похожей на
стихотворение вольным
       стихом.
       - Точка, отделяющая целую часть числа Пи от десятичной, не должна
учитываться в
       функции: она присутствует в инструкциях и примерах только для удобства
чтения. */


    public static String pilish_string(String str) {
        // тут должна быть функция для высчитывания PI
        String res = "";
        String pi = String.valueOf(Math.PI).replace(".", "");
        int piIndex = 0;
        while (str.length() > 0) {
            int p = pi.charAt(piIndex) - 48;
            int n = Math.min(p, str.length());
            res += str.substring(0, n);
            str = str.substring(n);
            piIndex++;
            if (str.length() > 0) res += ' ';
            else if (p > n)
                for (int i = 0; i < p - n; i++)
                    res += res.charAt(res.length() - 1);
        }
        return res;
    }

    /* 8. Создайте функцию для генерации всех непоследовательных двоичных
строк, где
       непоследовательные определяется как строка, в которой нет
```

```
последовательных
       строк, и где n определяет длину каждой двоичной строки. */

    public static String generateNonconsecutive(int num) {
        StringBuilder sb = new StringBuilder();
        StringBuilder res = new StringBuilder();

        for(int i = 0; (double)i < Math.pow(2.0D, (double)num); ++i) {
            sb.delete(0, sb.length());
            sb.append(Integer.toBinaryString(i));

            while(sb.length() < num) {
                sb.insert(0, '0');
            }

            if (!sb.toString().contains("11")) {
                res.append(sb).append(" ");
            }
        }

        return res.toString();
    }

    /* 9. Шерлок считает строку действительной, если все символы строки
встречаются
       одинаковое количество раз. Также допустимо, если он может удалить
только 1
       символ из 1 индекса в строке, а остальные символы будут встречаться
одинаковое
       количество раз. Для данной строки str определите, действительна ли
она. Если да,
       верните «ДА», в противном случае верните «НЕТ».
       Например, если str = "abc", строка действительна, потому что частота
символов у всех
       одинакова. Если str = "abcc", строка также действительна, потому что
мы можем
       удалить 1 "c" и оставить по одному символу каждого символа в строке.
Однако, если
       str = "abccc", строка недействительна, поскольку удаление одного
символа не приводит
       к одинаковой частоте символов. */

    public static String isValid(String str) {
        int[] set = getLetterSet(str);
        int[] medium = new int[str.length()];
        for (int i = 0; i < set.length; i++)
            if (set[i] != 0) medium[set[i]]++;
        int cur = 0;
        int max = 0;
        for (int i = 0; i < medium.length; i++)
            if (medium[i] > cur) {
                cur = medium[i];
                max = i;
            }
        boolean index = false;
        for (int i = 0; i < set.length; i++)
            if (set[i] != 0 && max - set[i] != 0) {
                if (index) return "NO";
                index = true;
            }
        return "YES";
    }

    /* 10. Создайте функцию, которая получает каждую пару чисел из массива,
```

```
который
       суммирует до восьми, и возвращает его как массив пар (отсортированный
по
       возрастанию). */

    public static int[][] sumsUp(int[] arr) {
        int[][] res = new int[arr.length / 2 + 1][2];
        int index = 0;

        for(int i = 0; i < arr.length; ++i) {
            for(int j = i+1; j < arr.length; ++j) {
                if (arr[i] + arr[j] == 8) {
                    res[index][0] = Math.min(arr[i], arr[j]);
                    res[index][1] = Math.max(arr[i], arr[j]);
                    ++index;
                }
            }
        }

        return res;
    }

}
```