Given is a program that compiles. What will be the results of its execution in the console?

Example 1:

```java
public class TestCodeReadEx1 {
        public static void main(String[] args) {
            int[][] multi = {
                    {0, 1, 2},
                    {3, 4, 5},
                    {6, 7, 8, 9}
            };
            for (int[] j : multi) {
                int idx = j.length - 1;
                while (idx > 0) {
                    System.out.print(j[--idx] + "\t");
                }
            }
        }
}
```

Output:

Example 2:

```java
public class TestCodeReadEx2 {
        public static void main(String[] args) {
            String result = rec(12, 8);
            System.out.println();
            System.out.println(result);
        }
        public static String rec(int in, int i) {
            if (in == i) return "";
            System.out.print(i + " ");
            String result = in + " " + rec(in, i + 2);
            System.out.print(i + " ");
            return result;
        }
}
```

Output:

Example 3:

```
public class TestCodeReadEx3 {
    public static void main(String[] args) {
        String s = "There is";
        methodA(s);
        System.out.println(s);
    }
    public static void methodA(String s){
        s += "hope";
    }
}
```

Output:

Example 4:

```
public class TestCodeReadEx4 {
    public static void main(String[] args) {
        method(5);
    }
    public static void method(int n){
        if(n<=0)
            System.out.print(n+"\t");
        else{
            method(--n);
            System.out.print(n +"\t");
        }
    }
}
```

Output:

Example 5:

```java
public class TestCodeReadEx5 {
    public static void main(String[] args) {
        int[] tab = {1, 2, 3};
        int[] tabX = tab;
        tab[0] = 9;
        tab[tab.length-1] = 8;
        tabX[0] = 7;
        tabX[tab.length-1] = 6;
        int[][] tabY ={tab,tabX};
        for (int[] ints : tabY) {
            for (int i = ints.length-1; i >= 0; i--) {
                System.out.print(ints[i] + "\t");
            }
            System.out.println();
        }
    }
}
```

Output:

1) Implement the **byte[] uniqueDigits(int value)** method, which will return an array containing the digits that make up the number value. The array cannot contain duplicates.

E.g. for 421314 -> {4, 2, 1, 3}

2) Implement a **boolean isExponent(byte value)** method that returns **true** if value is a power of two, **false** otherwise. **ATTENTION: it is forbidden to use ready-made methods from the Math class**. For example: for 2 -> true, 3 -> false, 64 ->true

3) Using the above-mentioned methods, write a program that displays the digits of the number in descending order, and skips powers of 2 and digit repetitions contained in the number. **ATTENTION: it is forbidden to use ready-made sorting methods (e.g. Arrays.sort())**

E.g.: 421314 -> 3,1

4) Implement a recursive method for exponentiation of positive numbers.

int pow(int a, int to)

E.g.: pow(2,2) -> 4; pow(5,0) -> 1;