Problem 1

Create a class **RecIter** containing only public static functions:

- `public static int fiboR(int n)` calculating the n-th Fibonacci number
$$F_n = \begin{cases} n & \text{for } 0 \leqslant n < 2, \\ F_{n-1} + F_{n-2} & \text{for } n \geqslant 2 \end{cases}$$
using this recursive formula, which is rather unwise but enlightening — therefore, the function should be *recursive* (in particular, no loops are allowed in its implementation!);
- `public static int fiboI(int n)` calculating the n-th Fibonacci number without recursion (i.e., iteratively — using a loop);
- `public static int factR(int n)` calculating $n!$ recursively;
- `public static int factI(int n)` calculating $n!$ iteratively;
- `public static int gcdR(int a, int b)` calculating the GCD, i.e., the greatest common divisor, of a and b recursively;
- `public static int gcdI(int a, int b)` calculating the GCD of a and b iteratively;
- `public static int maxElem(int[] arr, int from, int to)` returning the largest element of the elements of array arr from element with index from *inclusive* to element with index to *exclusive*. It must be a **recursive** function. Function **max** from class **Math** may be useful;
- `public static void reverse(int[] arr, int from, int to)` reversing the order of elements of the array arr with indices from from *inclusive* to to *exclusive*. It must be a **recursive** function. Do *not* create any auxiliary arrays!.
- `public static boolean isPalindrom(String s)` returning **true** if, and only if, string s is a palindrom, i.e., a word which reads the same forward and backward, as, e.g., words *radar* or *madam*. Methods **charAt** and **substring** from class **String** may be useful. It must be a **recursive** function.

Then, in the **main** function *of a separate class* **Main** test all these functions.

Remark: according to Euclid (*Elements*, Book VII), the greatest common divisor of two positive integers, a and b, can be calculated as follows:

1. if $a = b$, then the result is a (or b, as they are equal);
2. from the larger of these two numbers subtract the smaller and go to 1.

Do not use any classes from packages other than basic *java.lang* (in particular, no collections are allowed).

You can assume that functions will be invoked with legal arguments (e.g., no negative argument of the factorial function).

For example, the following **main** function

download *FuncStat.java*

```
public static void main (String[] args) {
    System.out.println(RecIter.fiboR(45));
    System.out.println(RecIter.fiboI(45));
    System.out.println(RecIter.factR(10));
    System.out.println(RecIter.factI(10));
    System.out.println(RecIter.gcdR(12125,40643));
    System.out.println(RecIter.gcdI(12125,40643));
    int[] a = {3,8,2,9,7};
    System.out.println(RecIter.maxElem(a,0,a.length));
    RecIter.reverse(a,0,a.length);
    for (int i = 0; i < a.length; ++i)
        System.out.print(a[i] + " ");
    System.out.println();
    System.out.print("Is 'radar' a palindrom? ");
    System.out.println(RecIter.isPalindrom("radar"));
    System.out.print("Is 'rover' a palindrom? ");
    System.out.println(RecIter.isPalindrom("rover"));
}
```

should print (note that calculating Fibonacci number F_{45} recursively takes a while. . .)

```
1134903170
1134903170
3628800
3628800
97
97
9
7 9 2 8 3
Is 'radar' a palindrom? true
Is 'rover' a palindrom? false
```