Problems about recursion

1. Show the evolution of the call stack for the following method with input $n = 5$. We recommend you to use as a reference the lines with `//****`

```
/** n>=0 */
public static int fact( int n )
{
    int p;
    //****
    if ( n == 0 )
        p = 1;
    else
        p = n * fact(n-1);
    //****
    return p;
}
```

2. Write a Java method, static and recursive, that shows on screen the natural numbers from 1 up to n, where $n > 0$ is the input argument. The method could be invoked from another one, such as `main()`, as follows:

```
int n = ...;
naturalNumbers( n );
```

3. Write a Java method, static and recursive, that shows on screen the natural numbers from n to 1, where $n > 0$ is the input argument. The method could be invoked from another one, such as `main()`, as follows:

```
int n = ...;
reverseNaturalNumbers( n );
```

4. Write a Java method, static and recursive, that given two natural numbers $a \geq 0$ and $b > 0$ computes the quotient of the integer division $\lfloor \frac{a}{b} \rfloor$ by means of successive subtractions. You can use the following recurrence relation:

   - $a/b = 0$, if $a < b$,
   - $a/b = (a - b)/b + 1$, if $a \geq b$.

5. Given two integer numbers a and b, where $b \geq 0$, it can be recursively defined their product $a \cdot b$ as follows:

   - $a \cdot b = 0$, if $b = 0$,
   - $a \cdot b = a \cdot (b - 1) + a$, if $b > 0$.

   Please, notice that for this definition the product is reduced to a sequence of sums.

   Whereas it is only possible to use sums, write a Java method, static and recursive, for performing the operation previously defined.

6. Given two integer numbers a and b, where $b \geq 0$, another way of defining their product $a \cdot b$ is the following:

   - $a \cdot b = 0$, if $b = 0$,
   - $a \cdot b = (a \cdot 2) \cdot (b/2)$, if $b > 0$ y b is even, y
   - $a \cdot b = (a \cdot 2) \cdot (b/2) + a$, if $b > 0$ y b is odd.

   This sort of multiplication is known as *The Russian Peasant Multiplication*. It was used aforetime by Russian traders who didn't know the multiplication table. For performing the product of two positive integer numbers, they only used three arithmetic operations: sums, multiplications by 2, and divisions by 2.

   Whereas it is only possible to use these three arithmetic operations, write a Java method, static and recursive, to perform the requested operation, following the above definition.

7. Given two integer numbers $a \neq 0$ and $b \geq 0$, it can be recursively defined a^b as follows:

   - $a^b = 1$, if $b = 0$,
   - $a^b = a$, if $b = 1$,
   - $a^b = (a^{b/2}) \cdot (a^{b/2})$, if $b > 1$ and b is even, and
   - $a^b = (a^{b/2}) \cdot (a^{b/2}) \cdot a$, if $b > 1$ y b is odd.

   Whereas it is only possible to use divisions by 2 and multiplications, write a Java method, static and recursive, for performing the requested operation and following the above definition.

8. Write a Java method, static and recursive, that computes the sum of the digits of a natural number $n \geq 0$ given as parameter.

9. Write a Java method, static and recursive, that returns the number of the digits of a natural number $n \geq 0$ given as parameter.

10. Write a Java method, static and recursive, that shows on screen in reverse order the digits of a natural number $n \geq 0$ given as parameter.

11. Write a Java method, static and recursive, that returns the binary value (represented as an integer number) of a natural number $n \geq 0$ given as parameter. As an example, if $n = 5$ the method returns `101`, but if $n = 31$ the method returns `11111`.

12. Which is the output on screen of the following program?

```
class Rare {
    /** n>=0 */
    public static void writeRare( int n )
    {
        if ( n > 0 ) {
            System.out.print(n);
            writeRare(n-1);
            System.out.print(n);
        } else
            System.out.print(0);
    }

    public static void main( String[] args )
    {
        writeRare(5);
    }
}
```

13. Write a Java method, static and recursive, that given two `String` objects referenced by $s1$ and $s2$, and without using the methods of the `String` class for solving it, determines:

    a) if $s2$ is prefix of $s1$.

    b) if $s2$ is suffix of $s1$.

    c) if $s2$ is contained within $s1$.

14. Write a Java method, static an recursive, that given a `String` object s and its length l, shows on screen the chars of s in reverse order.

15. Write a Java method, static and recursive, for obtaining the same output in the previous problem without using as argument the length of s, i.e, using just one parameter, s. Hint: you can use the method `substring()` defined in the `String` class.

16. Write a Java method, static and recursive, for checking if a given `String` is palindrome. Hint: you can use the method `substring()` defined in the `String` class for reducing the size of the input parameter at every recursive call.

17. It can be observed in the Pascal triangle of the following figure that each item is the sum of the items atop it, except the first and last numbers of each row, whose value is always one.

```
                  1
               1     1
            1     2     1
         1     3     3     1
      1     4     6     4     1
    1    5    10    10     5     1
 1    6    15    20    15     6     1
```

Write a Java method, static and recursive, that returns the value of the i-th element of the r-th row in a Pascal triangle. The profile of the method should be `public static int PascalTriangle( int r, int i )`.

Additionally, write a Java class with a `main()` method, that ask the user for the number of rows, and using the recursive method shows on screen all the elements of the corresponding Pascal triangle. It is not necessary to show the values in the form of a triangle.

18. Which is the output on screen of the following program?

```
class Sum
{
    public static int sumV( int[] v, int i, int x )
    {
        int sum = 0;
        if ( i < 0 ) return sum;
        if ( v[i] == x ) return sum;

        for( int j=0; j <= i; j++ ) sum = sum + v[j];

        System.out.println( "Partial sum: " + sum );
        return sum + sumV( v, i-1, x );
    }

    public static void main( String[] args )
    {
        int[] v = {1, 2, 3, 4, 5};
        System.out.println("Total: " + sumV( v, 4, 2 ) );
    }
}
```

19. Given an array of `String` objects `v`, write a Java method, static and recursive, that

    a) determines if is palindrome, i.e., the first and last words are the same, the second and the second last are the same, and so on. The method will return `true` if the array is palindrome, `false` otherwise.

    b) given a word `w`, determines if `w` is in the array between the positions `start` and `end`, where $0 \leq$ `start` $\leq$ `end` $<$ `v.length`. The method will return the first position where `w` is in the array from `start` if it exists, and -1 otherwise.

20. Which is the output on screen of the following program?

```
class Trace
{
    public static void main( String[] args )
    {
        int v[] = {0, 11, 2, 13, 4, 5, 6, 17, 8};
        f( v, 0 );
    }
```

```
        public static void f( int[] v, int i )
        {
            if ( i >= v.length )
                System.out.println( "-------------" );
            else if ( i == v[i] ) {
                System.out.printf( "v[%d]==%d\n", i, v[i] );
                f( v, i+1 );
            } else {
                f( v, i+1 );
                System.out.printf( "v[%d]!=%d\n", i, i );
            }
        }
    }
```

21. Implement the recursive version of the Binary Search algorithm that was proposed in the chapter dedicated to arrays.

22. Given an array of integer numbers `v`, write a Java method, static and recursive, that:
    a) Obtains the sum of all the items in the array.
    b) Obtains the position of the maximum (minimum) in the array.
    c) Given an integer number `x` counts how many times `x` appears within the array.
    d) Checks if the array is sorted in ascending order.
    e) Determines the position of the first (last) non null element in the array.
    f) Determines how many consecutive zeros are stored at the end of the array.
    g) Given two indexes `left` and `right` for the array, where $0 \leq$ `left` $\leq$ `right` $<$ `v.length`, reverses all the items of the array between such indexes, i.e. when the method ends `v[left]` contains the value that was stored at `v[right]` and *vice versa*, `v[left+1]` contains the value that was stored at `v[right-1]` and *vice versa*, and so on.
    h) Given two indexes `left` and `right` for the array, where $0 \leq$ `left` $\leq$ `right` $<$ `v.length`, duplicates the value of the items in the array between such indexes.
    i) Given an integer $b > 0$, determines if b is equal to the sum of all the elements of `v`.
    j) Given an integer number x counts how many elements of `v` are lower than x.
    k) Determines the amount of odd values stored in even positions of the array.
    l) Determines the position of the first subsequence in the array of at least three consecutive numbers that are stored in consecutive locations in the array. If it doesn't exist such a subsequence the method should return -1.