

1. Make a recursive method that receives a string and returns another string:
 - a) With its characters with the same order of the original string.
 - b) With its characters with reverse order of the original string.

2. Implement a recursive algorithm:
 - a) To compute the product of two positive integers, m and n , using only the arithmetic operations: addition and subtraction.
 - b) To find the maximum element in an array of n elements.
 - c) To calculate the greatest common divisor of two positive integers using the algorithm of successive divisions. For example, $m.d.c(48,30)=6$.
 - d) To convert a string of digits into the integer it represents. For example, "13531" represents the integer 13531.
 - e) To see if a number is palindrome that is, the number is the same when written forwards or backwards (examples: 99, 101, 111, 121, 1221, 21112, 10001, ...).
 - f) To compute the sum of all the elements in an $n \times n$ (two-dimensional) array of integers.

3. Develop a recursive method to demonstrate the backtracking through the search of a path in a labyrinth. Consider that only horizontal and vertical movements are allowed (diagonal are prohibited) and movements obey the following order: north↑, east→, south↓ and west←. Represent the labyrinth by a matrix of zeros and ones, in which the walls represent zeros and ones halls. For example, for the following labyrinth 7x13:

0	1	1	1	0	1	1	0	0	0	1	1	1	1
1	1	0	1	1	1	0	1	1	1	1	1	0	1
2	1	0	0	0	1	0	1	0	1	0	1	0	1
3	1	0	0	0	1	1	1	0	1	0	1	1	1
4	1	1	1	1	1	0	0	0	0	1	0	0	0
5	0	0	0	0	1	0	0	0	0	0	0	0	0
6	0	0	0	0	1	1	1	1	1	1	1	1	1
	0	1	2	3	4	5	6	7	8	9	10	11	12

A recursive method that seeks a path between the source (0, 0) and the destination (6, 12) should mark the labyrinth path cells with **9** and cells accessed but which have not led to the solution with **2**. So the return positions (backtracking) can be viewed with **2** like in the following matrix.

