

Fourth lesson: Recursivity. The merge-sorting algorithm. Complexity of sorting algorithms.

Recursivity is a programming technique such that a function calls itself.

For example, the following function prints the first 'aux' natural numbers.

```
def stampa(i):  
    if i == 0:  
        return  
    stampa(i-1)  
    print i
```

(please note the difference in the function output when inverting the order of the two last lines of the function).

EX0. Write recursive functions performing the following operations: computes the factorial of a given number; prints all the Fibonacci numbers lower than its argument; checks whether the input number is prime.

As a further application of recursivity we propose the recursive solution of the merge-sort algorithm for sorting a list.

It is based on the following strategy, that we illustrate with the list [8,5,7,3,2,1,0]:

A) split a list in elements and merge them in two-lengthed lists so that they are in order: [5,8] , [3,7] , [1,2] , [0]

B) merge the lists in couples forming a larger list from each couple of lists. The merging is done beginning from the leftmost element of each list, so that the merged list is in order (one cancels the lower element of one of the lists, and continues comparing the leftmost elements): [3,5,7,8] , [0,1,2]

C) repeat recursively such an ordered merging of lists: [0,1,2,3,7,8]

The merge-sort algorithm complexity in time is of order $n \log(n)$ in the worst, average and best cases (n being the list length).

EX1. Write a Python module containing a (top-bottom) implementation of the merge-sort algorithm. The module may consist in two functions, one which merges two list in order: the second splits the original list in two halves, calling itself with each one of the halves as argument, then merging the resulting list. Notice that the recursive self-call should be done before merging them (top-bottom implementation).

In the python notebook 'timeTestSortFunctions.ipynb' we provide a framework to estimate the computational complexity of several sorting algorithms.

