# PROGRAMMING TECHNIQUES, A.A. 2022/2023

# Laboratory 4

Exercise 3 is a home assignment that can be optionally submitted for evaluation, to obtain the exam's bonus. Deadline is 14/5/2023, 11:59 pm, and it will be the same for laboratories 4, 5 and 6. The home assignments of Lab4+Lab5+Lab6 need to be submitted in one shot by the given deadline, following the instructions that are provided in the Portale della Didattica (see the document: Instructions for submission of assignments.pdf)

#### Objectives

• Solve iterative numerical problems, using arrays (mono-dimensional arrays and matrixes) (C3b-Problem solving with arrays: part I)

## Technical content

- I/O basics
- Functions
- Conditional and iterative problems
- Operations with arrays (of int and float)

#### Exercise 1.

Category: problems with numerical sequences

## Numerical sequences in arrays

v is a mono-dimensional array of n integers (with  $n \le 30$ ). Write a C program that, after acquiring the values of the array from keyboard, calls a function with the following prototype:

```
subSequences(int v[], int n);
```

The function should print on the screen all the sub-vectors of maximum size formed by contiguous elements, containing non-zero values.

## **Example**

If v is  $[1\ 3\ 4\ 0\ 1\ 0\ 9\ 4\ 2\ 0]$ , the two sub-vectors of maximum dimension (3) with non-zero contiguous elements are  $[1\ 3\ 4]$  and  $[9\ 4\ 2]$ .

#### Tips:

*Note that the problem can be addressed by solving two sub-problems:* 

- identify sub-vectors of non-zero elements
  - it is sufficient to "recognize" the beginning and the end of these sub-vectors
- select those with maximum length:
  - a simple method consists in "trying" all the possible lengths in descending order, stopping at the first length for which there exists a sub-vector
  - a more efficient method is to first determine the maximum length, and then search for the corresponding sub-vectors
  - alternatively, one could also try to do a single iteration on the main vector to recognize the subvectors, determine the maximum length and "remember" (using another vector) the beginnings of the sub-vectors: but the complexity/efficiency would not change (you would still need a further

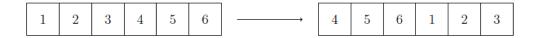
iteration to print the sub-vectors) and (probably) the program would be more complicated than in the previous version.

#### Exercise 2.

Category: problems with ordered numerical sequences

#### **Rotation of arrays**

Write a function C capable of rotating the content of an array of n integers to the right or to the left by a number of positions p. The array is to be understood as a *circular* array, in the sense that the element to the right of the index cell N-1 is the one with index 0 and the element to the left of the index 0 is the index N-1. The following figure illustrates a right rotation by 3 positions:



The function should have the following prototype:

The main should:

- 1. acquire n from keyboard ( $n \le maxN$  with #define maxN 30)
- 2. acquire the values of the array v from keyboard
- 3. perform repeated rotations of the input array, where at each iteration the user decides the value of p (p < n, p=0) to terminate the program) as well as the direction of the rotation (dir =-1 for right, dir = 1 for left) and the resulting rotated array is printed on the screen.

## Exercise 3. (THIS EXERCISE IS TO BE SUBMITTED FOR THE EXAM BONUS)

Category: problems numerical matrixes

#### **Iterations on matrixes**

A text file contains an array of integers with the following format:

- the first line of the file specifies the dimensions of the matrix (number of rows nr and number of columns nc), separated by spaces. Assume that both values are  $\leq 20$ .
- each of the subsequent lines contains the no values corresponding to a row of the matrix, with one or more spaces as separators.

Write a C program that:

- reads this matrix from the input file (the name of the file, maximum 20 characters, is read from the keyboard)
- repeatedly asks the user for a dim value between 1 and the minimum between nr and nc, and prints all the square sub-matrixes of size dim that are contained in the input matrix
- prints the square sub-matrix, among those previously identified, that has the maximum sum of elements
- terminates the iterations if the user enters a value that is inconsistent with the size of the matrix

# Example

If the content of the input file is the following:

```
1 2 3 4
5 6 7 8
9 0 1 1
and dim=2, the program should print on the screen:
The square sub-matrixes of dimension 2 are:
1 2
5 6
2 3
6 7
3 4
7 8
5 6
9 0
6 7
0 1
7 8
1 1
The submatrix with maximum sum of elements (22) is:
3 4
7 8
```