Lab. 6 Array Search and Sorting; Input/Output to Text Files

Do the exercises below in the Octave IDE. Make sure the files and the programs are in the same working directory.

0. Adapt Bubble Sort

Adapt the implementation of Bubble Sort presented in the slides of class 6, by adding to the results a) the number of bubbles that were considered, and b) the number of bubbles that were swapped. Use the signature

```
function [S,bb,sw] = bubble_sort(V)
```

1. Optimise Bubble Sort

In the implementation of Bubble Sort presented in the slides of class 6, the procedure executes exactly **n-1** sweeps (outer loop – **for k = n:-1:2**), each over with decreasingly ranges of the vector (inner loop – **for i = 1:k-1**). However, if the vector is already sorted, or if a prefix of the vector is already sorted, sweeping the bubble does not change the vector any longer, and only wastes time.

Adapt the function presented in the slides of class 6, so that this inefficiency is eliminated.

2. Input a Structure Array

Read to a structure arrays, **grades**, the data stored in file "grades.txt" available in the web site.

3. Sort a Structure Array (numeric field)

Define a function with the signature below, that returns the structure array given as input, G, sorted by the grades of the students, in the order specified in the second argument (order can be either 0/1 to denote decreasing/increasing order). Use Bubble Sort as the sorting algorithm.

```
function S = sort_grades(G,order)
```

4. Searching Elements in an Array Structure

Define the two functions below that return p, the index of an element with a given grade from an array structure A, like that of the previous questions (either G or S). If no such element exists, return 0. If more than one exists, return the index of one of them. In addition, return also k, the number of elements analysed before finishing execution. The two functions differ on the algorithm used: search_lin performs a linear search, whereas search_bip performs a bipartite search.

```
function [p,k] = search_lin(A,grade)
function [p,k] = search_bip(A,grade)
```

5. Compare two strings

Define the function below that compares two strings str1 and str2

```
function c = comp_strings(str1, str2)
```

and return one of the values 0, 1 or 2 depending when the two strings are equal (0), str1 is alphabetically before str2 (1), and str2 is alphabetically before str1 (2).

6. Sort a Structure Array (text field)

Define a function with the signature below, that returns the structure array given as input, G, sorted by the names of the students, in the order specified in the second argument (order can be either 0/1 to denote decreasing/increasing order). Use Insert Sort as the sorting algorithm.

```
function S = sort_names(G,order)
```