Programming 2024/25

Law, Law-PL, Law-CE

Laboratory class 1

Unidimensional tables / problem solving strategies

Bibliography: K. N. King. *C programming: A Modern Approach* (2 nd edition). W. W. Norton: Chapters 1 to 9.

Class Support Code: https://gist.github.co
m/franciscobpereira

Compulsory Exercises

1. Write a function in C that finds the largest element stored in an integers table. The function rec
eives as arguments the name and size of the table. Returns as a result the highest value existing in the table.

2. Write a function in C that finds the largest element stored in an integers table. The function rec
eives as arguments the name and size of the table. Returns as a result the position of the highest v
alue in the table. If the highest value arises several times in the table, the function returns the first position in which it occurs.

3. Write a function in C that tells how many times the highest element in an integers table arises. The function receives as arguments the name and size of the table. Returns as a result the number of times the largest element appears in the table.

4. Write a function in C that finds out which element arises most often in an integers table. The fu
nction receives as arguments the name and size of the table. It returns as a result the most commo
n element of the table (i.e., which arises more often). If there are several more common elements, the function should return the largest of them.

5. Write a function in C that reverse the order by which the elements are in an integers table. The f
unction receives as arguments the name and size of the table.

6. Write a function in C that moves all elements of an integer table a position to the right. In this c
ase, the last element should become the first. The function receives as arguments the name and si
ze of the table.

Programming 2024/25

Law, Law-PL, Law-CE

Complementary exercises

7. Write a function in C that receives an integers vector and write all the only values on the console. The written values must be in the same order that appear in the vector, in a single line and separated by a blank space. The prototype of the function is as follows:

```
void unicos(int v[ ], int tam);
```

Example: Given the table {2, 30, 4, 7, 10, 3, 12, 15, 2, 10} with 10 elements
the function must present in the console: 30 4 7 3 12 15

8. Write a function in C that receives a table of integers more than or equal to 3 and write on the console the combination of all sets of 3 elements of the table whose sum equals a certain value. This value is also passed by argument. The prototype of the function is as follows:

```
void calculaSoma(int tab[], int dim, int valor);
```

Example: Given the table {1, -2, 3, 4, -5, 6} with 6 elements and the value 8
THE function must present in the console: 1 3 4 -2 4 6

Autonomous work

9. Write a function in C that make sure a certain value is in an integers table. The table is increasingly ordered and contains no repeated elements. The function receives as arguments the name and size of the table and the value to search. Returns 1 if the value is in the table, or 0, otherwise. The prototype of the function is as follows:

```
int research (int tab [], int dim, int value);
```

This is a classic research problem in an orderly table. There are several possible strategies to solve it and, in the context of this form, it is intended to implement two:

I. Linear Research: It is the simplest strategy for finding the element. The function begins the research at the beginning of the vector and sequentially travels the elements until the value is found (or to find that the value is not in the table - do not forget that the vector is ordered in an increasing way).

II. Binary Research: It is an algorithm based on the successive division of the research space. It is simple to understand and implement and solve the problem much more efficiently than linear research. Look for an explanation of this strategy and implement the corresponding function.