Fundamentals of Programming, ETSII

First Final Exam. 5 February, 2016

SURNAME, Name | eMail | CmptrID | Conv#

At the end, hand this sheet in filled with your name, and email, machine number, convocatoria number, please

Use **prEx1**, **prEx2**, **prEx3**, **prEx4** for the exercises 1, 2, 3 y 4 respectively
At the end of the exam you will only need to upload these 4 files.cpp to the corresponding task on the Campus Virtual

If on *CodeBlocks*, don't forget to activate the Preference (at the bottom of Window menu):

`[v] Have g++ follow the C++11 ISO language standard [-std=c++11]`

On Windows work in the directory `C:\FP\EXAMENSEPT`, don't forget to set that as Workspace

On Mac OSX Terminal go to Desktop. Remember:

```
cd Desktop
subl problem1.cpp
(..edit/save...)
c++ prEx1.cpp
./a.out
```

Do not forget to write your Surname, Name; Degree; Group; Machine Number at the top of every .cpp file, each thing in a different line
Do not use any external disks, documentation, nor speak with anyone except the lecturers

1 1pts Write a **function** that guess if a square matrix received as parameter is an Identity Matrix. The function will return true or false. An square matrix is an Identity Matrix when all of its diagonal elements are 1 and the rest are all 0. Use the file `prEx1.cpp` provided filling with the missing code there. Do not change any other part of that file. The program must work in an efficient way. The execution should print:

```
First example of matrix:
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
IS an Indentity Matrix

Second example of matrix:
1 0 5 0
0 1 0 0
0 0 1 0
0 0 0 1
is NOT an Indentity Matrix
```

2 2pts A Set is a collection of non-repeated numbers. You already have defined the type `TSet` admitting up to (const) `MAX` integers; besides this you have in the file `prEx2.cpp` 2 procedures to read and write a `TSet`. To complete the `prEx2.cpp` program you have to build the next two procedures. Add to them whatever other subprograms you consider appropriate.

intersectionOf(): receives two parameters `TSet` and returns a third `TSet` with the intersection of both: elements that belong to both at the same time.

unionOf(): receives two parameters `TSet` and returns a third `TSet` with the union of both: elements that belong to any of them. In this case we will want a third output parameter (`ok`) in which to indicate if there was overflow, more elements than the capacity of a `TSet`

First example of execution:

```
First TSet:
Number of items to read? (<= 10): 5
Input 5 different natural numbers: 4 2 6 15 9

Second TSet:
Number of items to read? (<= 10): 6
Input 6 different natural numbers: 6 3 4 26 1 2

Intersection:
4 2 6

Union:
4 2 6 15 9 3 26 1
```

Second example of execution:

```
First TSet:
Number of items to read? (<= 10): 5
Input 5 diferent natural numbers: 4 2 6 15 9

Second TSet:
Number of items to read? (<= 10): 8
Input 8 diferent natural numbers: 1 10 2 7 3 8 11 17

Intersection:
2

Overflow while building union
```

3 3.5pts Build a subprogram `sliding()` that receives two strings as parameters. The subprogram will return (as a parameter) what is the shortest **displ**acement of the second string (the second one) over the first one in order to have the greatest number of coincidences between both strings. At the same time the subprogram will return (as a second parameter by reference) the number of **coinc**idences reached. For example for the strings `acbaabch` and `aabghc` would return 3 as displacement and 4 as the number of coincidences. Let us see it:

```
acbaabch  // displ: 0; coinc: 2
aabghc
acbaabch  // displ: 1; coinc: 1
 aabghc
acbaabch  // displ: 2; coinc: 1
  aabghc
acbaabch  // displ: 3; coinc: 4
   aabghc
...
acbaabch  // displ: 7; coinc: 0
       aabghc
```

Build a `main()` program for this `prEx4.cpp` that reads 2 strings, call the procedure `sliding()` and prints on the screen the output values

4 3.5pts Design a program that asks the user for a series of words until the word `"end"` is entered. Saving the words but *not adding repeated words*. Call then a function **encodeIt()** for each word, after the call print each returned encoded word. To encode the string add the syllable `pa`, `pe`, `pi`, `po`, `pu` just after `a`, `e`, `i`, `o`, `u`; respectively. There are a limited capacity for words in our system, let us call it, `NMAXWORDS = 10`. For example:

this day can be a nice day end → thipis dapay capan bepe apa nipicepe

better to ask and appear foolish than to not ask and continue in foolishness end → bepetteper topo apask apand apappepeapar fopoopolipish thapan nopot copontipinupuepe ipin