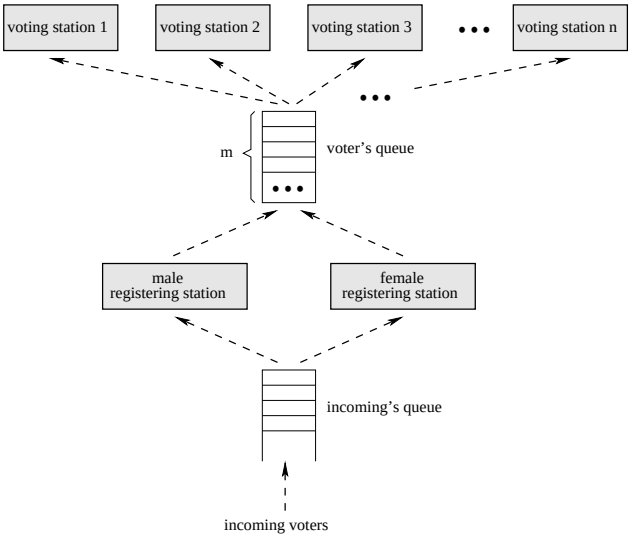
System and Device Programming

Examination Test – Programming Part
03 July 2014

Examination time: 1h 45min. Evaluation: 18 marks.
Textbooks and/or course material allowed.
The final mark is the sum of the 1st and the 2nd parts; it cannot be refused (no retry for marks ≥ 18*).*

The candidate is requested to write a concurrent program, in the Visual Studio environment, to simulate a polling station during a vote section. The polling station is organized as represented in the following picture.



Incoming people (both males and females) wait for their turn to have access to the register station in a unique queue. Males and females have separate register stations. After registering, each person enter a new queue (of size **m**), before proceeding to one of the **n** voting station for the final vote. The value of `m` and `n` depend on the room size where the polling station is located.

The program receives four parameters on the command line: An input file name, two integer values `n` (the number of voting stations) and `m` (the size of the internal queue), and an output file name.

The input files specifies incoming voters. Each line of the file reports the voter identifier (its "codice fiscale"), surname, name, sex, arrival time (in the format `hh:mm`), time necessary to register (in minutes), and time requested to vote (in minutes). The following is a correct example of such a file:

```
aaabbbccdee000f White John   M   08:30   3   4
aaabbbccdee001f Red   Mary   F   08:30   2   3
aaabbbccdee002f Blue  Gary   M   08:30   2   6
...
```

All records in the file have fixed size length, and are ordered by arrival time.

The program has to create a file in which each voter identifier is followed by the voting station number in which he voted and by the ending time of its voting procedure. The following is a correct example of such a file:

```
aaabbbccdee000f Voting_Station_1 08.37
aaabbbccdee001f Voting_Station_2 08.35
aaabbbccdee002f Voting_Station_3 08.41
...
```

(notice that the third voter has to wait 3 minutes before registering).

Notice that in all cases there is a unique male and a unique female register station. The voter's queue has to be implemented using a proper data structure and no file. Register stations have to wait if the voter's queue is full. Voting stations have to wait if the voter's queue is empty. The candidate **must** design a proper termination for the program (and all its threads) when the number of voters is terminated.