25 June 2013 -- Computer Architectures –- part 1/2

Last & First Name, Matricola .....

A community has a small library with slightly less than 6600 books.

- The books are stored into 4 rooms (R0, R1, R2, R3) each one hosting 128 "storage units", a.k.a. "blocks" numbered from 0 to 127.
- Each book has a unique inventory code given by an upper case letter (from A to Z) followed by a number from 0 to 255.
- Only registered users can borrow books; registration is a uniquely coded by a number from 0 to 2^{18}-1, followed by a lower case letter (a..z).
- As, by rule of the library, all users return the books by the end of each solar year, the database on loans resets at the end of the year, and therefore the date of a loan can be expressed simply by the number of the day of the current year (from 1 to 365).

It is required to write a 8086 assembly program to manage the database of loans and positions of books inside the library shelves. The database is defined as follows:

```
N EQU 6600
LOANS DB 7*N DUP (?)
```

where each entry of LOAN is made of 7 bytes, encoded as follows from most significant to least significant:

- **Status on 2 bits**, as follows: 00=empty entry of the database, 10=book available for loan, 11=book currently on loan;
- **Book inventory code on 13 bits**, as follows: most significant 5 bits for coding the inventory letter A to Z (i.e., 26 letters), followed by the least significant 8 bits of this field for coding the inventory number from 0 to 255;
- **Position of the book on 9 bits**, as follows: most significant 2 bits for coding the room number (i.e., R0 to R3), followed by the least significant 7 bits of this field for coding the block number from 0 to 127;
- **Registration number of the User on 23 bits** (if the corresponding book has been borrowed, otherwise this value is not significant), as follows: number on the 18 most significant bits, followed by the least significant 5 bits of this field for coding the registration letter a to z (i.e., 26 letters);
- **Starting day of loan on 9 bits** (if the corresponding book has been borrowed, otherwise this value is not significant), as follows: number of the day of the year when the book was last borrowed by an User.

It is assumed that:

- The number of books does not change during a solar year
- The entries of the database are not sorted either according to location or book inventory code
- In case of M books, then the related information is stored into the first M entries of the database
- It is known today's date in the form of a number between 1 and 366, and the initial database of today's operations is already filled with the information of "yesterday"

It is NOT permitted to:

- Mirror, or make a copy of the full database, as being a large data structure, it would not fit into a single data segment
- Sort the original database

It is requested to write a 8086 assembly program to manage the library and offering the following features (provided that all necessary information is provided):

- **Item 1 (max 16 points) MANDATORY!**
  - Manage loan and return of books (upon their inventory code) by registered users (upon their registration code, when necessary), including the update of the last loan date and computation of the number of days of loan in case the book is returned. Assume that inventory code and user registration code are valid;

25 June 2013 -- Computer Architectures –- part 1/2

Last & First Name, Matricola .....

- Item 2 (max 4 points for one of the two items, plus max 2 additional points if also the other item is solved)
  - Return the position of a book given its inventory code (independently if it has been borrowed or not), with proper error management if the book inventory code is not in the database;
  - Return the inventory code of a book by entering its location (independently if it has been borrowed or not), with proper error management if the location is not in the database;

- Item 3 (max 4 points for one of the two items, plus max 2 additional points if also the other item is solved)
  - Return the information if a book is available or has been borrowed, by entering its location position, assuming that location position is valid (i.e. present and valid in the database);
  - Return the information if a book is available or has been borrowed, by entering its inventory code, assuming that inventory code is valid (i.e. present and valid in the database);

- Item 4 (max 4 points)
  - Before agreeing a loan, check if the current User has already borrowed 4 other books and, in case she/he has, do not allow the loan and display an alert on the video

- Item 5 (max 2 points for any of the three items)
  - Return the statistics on how many books are available and how many have been borrowed, as it appears from the current picture of the database;
  - Return the statistics on what is the average current loan time (only for books under loan, indeed);
  - Return the inventory code and registration of number of the User, for the current oldest loan;

REQUIREMENTS (SHARP)

- **Prior solving other items, it is MANDATORY to solve Item 1**
- It is not required to provide the optimal (shortest, most efficient, fastest, …) solution, but a working and clear solution.
- It is required to write at class time a short and clear explanation of the algorithm used.
- It is required to write at class time significant comments to the instructions.
- The input-output part is not necessary in the class-developed solution, but its implementation is mandatory to be discussed at oral exam.
- Minimum score to "pass" this part is 15 (to be averaged with second part and to yield a value at least 18)

REQUIREMENTS ON THE I/O PART TO BE DONE AT HOME

- The database has to be defined and initialized inside the code and should contain at least 20 items
- The program should present a menu with the choices corresponding only to the items developed during the written exam
- All inputs and outputs should be in readable ASCII form (no binary is permitted), including proper messages in case errors are found for Item 2.

Please use carbon copy ONLY (NO PICTURES ARE ALLOWED) and retain one copy for home implementation and debug. At the end of the exam please give to professors all the sheets of your solution. Missing or late sheet will not be evaluated. Please provide your classroom submitted solution with several explanatory and significant comments. Please remember that only what has been developed at class time can and will be evaluated at oral time and that it is necessary to write the instructions of the program and not just the description of the algorithm.

When coming to oral discussion, please clearly mark in red on your "classroom" copy, all modifications. Please also provide an error-free and running release of the solution, as well as with its printed list of instructions. Please consider that the above are necessary but not sufficient requirements to success the exam, since the final evaluation will be based on a number of parameters.

FAILURE TO ACCOMPLISH ALL THE ABOVE NECESSARY REQUIREMENTS WILL CAUSE NO-QUESTION-ASKED AND IMMEDIATE REJECTION.