11 February 2014 -- Computer Architectures –- part 1/2

LAST Name, first name, Matricola ……

Let us consider a ski resort area counting 5 lift stations at the following elevations (in meters)
A:1510, B:1760, C:1830, D:2040; E: 2120.

The following lift low-to-high connections are available: A to B; A to C; B to D; C to D; C to E.
Ski tracks are available for any higher to lower station combination.
Each station is equipped with a ski pass tracking and validation system where the code of the ski pass owner as well as her/his transit hour are stored to a record. At the end of the day the full database is processed and all records to a specific ski pass owner are isolated in an array sorted by increasing passage hours.

It is known in advance that any skier does not take the same low-to-high connection more than 15 times, and does not ski the same high-to-low track more than 12 times. It is also assumed that no break (say, for lunch) is taken and that no more than 150 records are stored for each skier. A line with all zeroes identifies the end of valid records in the database. Finally, it is assumed that **no lift or descend takes more than 30 minutes**. For example:

Owner	station	passage hour
23	A	8:00:06
23	B	8:04:02 → B is higher than A (and is connected); lift time is 00:03:56
23	A	8:09:00 → A is lower than B; descend time is 00:04:56
23	C	8:15:14 → C is higher than A (and is connected); lift time is 00:06:14
23	D	8:22:58 → D is higher than C (and is connected); lift time is 00:07:44
23	B	8:35:22 → B is lower than D; descend time is 00:12:24
23	…	

The format of a record is as follows

Owner	station code	hours	minutes	seconds
12 bits	3 bits	5 bits	6 bits	6 bits

All time-sorted records related to the same owner are stored into the array DATABASE DD N DUP(?) . Students have to write an 8086 assembly program performing the following operations:

- Item 0: It is requested to define and fill a database related to all lifts and descends; it is mandatory to properly choose the most appropriate bit lengths for each field.
  With reference to the previous example:

type	start station	arrival station	time	
lift	A	B	03:56	(three minutes, 56 seconds)
descend	B	A	04:56	
lift	A	C	06:14	
lift	C	D	07:44	
descend	D	B	12:24	

  For this Item, Students can avoid (at the cost of receiving less points) implementing the code to compute the difference between two hours; however, prior implementing other Items, Students must implement the time difference code.

- Item 1: It is requested to compute and store to a suitable database, the statistic values of the number of passages of all possible up-lifts and down-descends.

- Item 2: It is requested to compute and store to a suitable database, the statistic values of slowest **or** fastest times of all possible up-lifts and down-descends (please choose either the slowest or fastest option; doing both will not increase the points accounted)

11 February 2014 -- Computer Architectures –- part 1/2

LAST Name, first name, Matricola **……**

- Item 3: It is requested to compute and store to a suitable database, the statistic values of the average time of passages of all possible up-lifts and down-descends.

Prior solving other items it is mandatory to solve Item 0; prior solving other items it is mandatory to implement the time difference procedure.

The points related to writing correct 8086 assembly code for each completed item (as uncompleted items will not be evaluated)

- Item 0 (**MANDATORY**): 23 points **(if the code to implement the difference of times is not implemented, the max points is 19). The time difference procedure is mandatory prior solving the other Items**
- Item 1: 4
- Item 2: 5
- Item 3: 6

Please consider that a maximum of 33 points can be accounted here; larger values will be "cut" to 33.

REQUIREMENTS (SHARP)

- **Prior solving other items it is mandatory to solve Item 0. prior solving other items it is mandatory to implement the time difference procedure.**
- 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
- 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).

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.