2 September 2013 -- Computer Architectures –- part 1/2

Last & First Name, Matricola ....

A company is offering lunch canteen services to its 30 employees, from Monday to Friday, i.e. with a minimum of 10 working days and a maximum of 23 working days per month. Each employee has a card with a unique identification number (from 1 to 30), which is always charged once every working day with an amount from 0.00 to 10.00 dollars. Each working day, 30 transitions (in the following refereed to as "records") are added to a canteen "charges" database, i.e. one for each employee on that day.

Each record is on 24 bits with the following format, from most-to-least significant:

most significant   least significant
<number of month><number of day of month><employee's code><amount charged>
            4   5   5   10

<number of month> ranges from 1 to 12; if <number of month> = 0 then the whole record is not valid
<number of day of month> ranges from 1 to 31; it cannot be equal to 0
<employee's code> ranges from 1 to 30; it cannot be equal to 0
<amount charged> is expressed in dollar cents and ranges from 0 to 1000 (cents)

From the data above, it is clear that the maximum number of records in a month is 23*30 = 690.

Each single record is stored in a canteen "charges" database which, or our purposes, corresponds to an array, according to the following definitions:

N_EMPLOYEES **EQU** 30
N_MAX_ WORKING_DAYS_PER_MONTH **EQU** 23
N_MAX_RECORDS_PER_MONTH **EQU** N_EMPLOYEES * N_MAX_ WORKING_DAYS_PER_MONTH
N_MONTHS **EQU** 12
N_RECORDS **EQU** N_MAX_RECORDS_PER_MONTH * N_MONTHS
N_BYTES_PER_RECORD **EQU** 3
N_BYTES_OF_RECORD **EQU** N_BYTES_PER_RECORD * N_RECORDS

CANTEEN_CHARGES_DATABASE **DB** N_BYTES_OF_RECORD **DUP** (?)

The single records inside the database (i.e. array "CANTEEN_CHARGES_DATABASE") are well sorted, i.e.:

- Sorted by increasing number of month, then
- Sub-sorted by increasing number of day, then
- Sub-sub-sorted by increasing number of employee's code;
- "non-valid-records" are only at the end of the array, i.e. after the end of the sorted array there are only "non-valid-records".

It is requested to write an 8086-assembly program to compute the statistics and the cost to be charged to each employee, according to the Items below.

Items with corresponding points for each completed item (as uncompleted items could be not evaluated)

- Item 1. (up to 8 points) = computation of the grand total charged to all employees during the whole year. As this number can be up to 1000*23*12*30 = 8,280,000 cents, a variable at least on 23 bits is necessary to correctly represent this result. Students have to properly make the right choice and usage of this variable.
- Item 2. (up to 12 points) = by receiving in input the code of an employee, computation of the total to be charged to that employee during the whole year. As this number can be up to 1000*23*12 = 276,000 cents, a variable at least on 19 bits is necessary to correctly represent this result. Students have to properly make the right choice and usage of this variable.
- Item 3. (up to 5 points) = by receiving in input the number of a month, computation of the total to be globally charged to employees during that month. As this number can be up to 1000*23*30 = 690,000 cents, a variable at least on 20 bits is necessary to correctly represent this result. Students have to properly make the right choice and usage of this variable.

Last & First Name, Matricola ....

- Item 4. (up to 4 points) = by receiving in input the code of the employee and number of a month, computation of the total to be globally charged to that employee for that month. As this number can be up to 1000*23= 23,000 cents, a variable at least on 15 bits is necessary to correctly represent this result. Students have to properly make the right choice and usage of this variable.
- Item 5. (up to 3 points) = by receiving in input the number of a day and of a month, computation of the total to be globally charged to all employees on that day of that month. As this number can be up to 1000*30= 30,000 cents, a variable at least on 16 bits is necessary to correctly represent this result. Students have to properly make the right choice and usage of this variable.

WARNING!
Please operate the right choices of variables to store the results, i.e. not smaller and not larger than what it is necessary, clearly fitting with the 8086 ways of declaring and using variables. Also please remember that 8086 operates "little-endian"!

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 90 items accomplishing with the definitions of exam's text above
- 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.