15 September 2014 -- Computer Architectures –- part 1/2

Matr, Last Name, First Name **……**

Back from vacation one of the biggest "surprises" arises from the credit card bill. Let us assume that an array EXPENSES (with at most of 50 records) has been provided with the single expenses of our credit card during the month of august, according to the following format: v ddddd cc eeeeeeee ffffffff
v = (1 bit) validity; 0 = not a valid line; 1 = valid line
d= (5 bits) day of month (1-31)
c = (2 bits) currency; 00 = USD, 01 = CAD, 10 = CHF, 11 = euro
e = (8 bits) integer part of amount credited (integer unsigned number)
f = (8 bits) fractional part of amount credited (unsigned number)

in other words, each expense is expressed in the currency cc by the positive value eeeeeeee.ffffffff
N EQU 50 EXPENSES DB 3*N DUP (?)

Another array of 31+1 records is provided for the 3 foreign currencies with the exchange rate in the form 0.tttttttt where only the 8 t-fractional bits are stored, where record 0 (the first one) provides the august average exchange rate, while records 1-31 provide the daily rate for that day (e.g. record 9, i.e. the 10th one, has the rate for august 9)
uuuuuuuu aaaaaaaa hhhhhhhh
where 0.uuuuuuuu is the exchange rate USD→EUR, 0.aaaaaaaa for CAD→EUR and 0.hhhhhhhh for CHF→EUR
EXCHANGE DB 32*3 DUP (?)

It is requested to write a 8086 assembly program to compute in the format eeeeeeee.ffffffff with an acceptable precision, the line by line amount of each expense converted in EUR and to store them in the array CONVERTED of N records in the same format as EXPENSES, i.e. CONVERTED DB 3*N DUP (?), where each valid record has the bits v = 1 (i.e. valid), ddddd as the same date when the expense occurred, cc = 11 saying that the amount has been converted in Eur, and eeeeeeee.ffffffff as the resulting amount, i.e.: v ddddd cc eeeeeeee ffffffff

- Item 1: choose one and only one among the following problems:
  - A. It is requested to compute the currency conversions and store them in the array CONVERTED only for the records of EXPENSES occurred in USD, by using the USD average exchange rate stored in the first line of EXCHANGE.
  - B. It is requested to compute the currency conversions and store them in the array CONVERTED only for the records of EXPENSES occurred in USD, by using the USD daily exchange rate as stored in the right day line of EXCHANGE.
  - C. It is requested to compute the currency conversions and store them in the array CONVERTED for the records of EXPENSES occurred in USD, CAD, CHF, and EUR as well (no conversion is required in this case) by using the average exchange rate stored in the first line of EXCHANGE.
  - D. It is requested to compute the currency conversions and store them in the array CONVERTED for the records of EXPENSES occurred in USD, CAD, CHF, and EUR as well (no conversion is required in this case), by using the corresponding currency daily exchange rate as stored in the right day line of EXCHANGE.
- Item 2: to compute the grand total of expenses which have been stored in the array CONVERTED and store them in the variable GRAND_TOTAL DB 3 DUP (?) in the form of 16 integer and 8 fractional bits.
- Item 3: by receiving in input a date, to compute the day expenses (in Eur) occurred in that day (and stored in one or more lines of CONVERTED) and to store them in the variable DAY_GRAND_TOTAL DB 3 DUP (?) in the form of 16 integer and 8 fractional bits.

For sake of simplicity, the array EXPENSES is sorted by date, from oldest to most recent and the non valid lines are all at the end of the array.

Example:
Record of EXPENSES: 1 01001 00 10011100 11000000 → valid record, expense on 9 of august in USD of 156.75
Assuming to have in the record number 9 (i.e. the tenth) of EXCHANGE the following 3 bytes (from most significant to least significant) 10111111 10101111 11010011

15 September 2014 -- Computer Architectures –- part 1/2

Matr, Last Name, First Name **……**

As we have to convert from USD to EUR, the leftmost (i.e. most significant) has to be taken, and the exchange rate USD→EUR is therefore 0.10111111 (i.e. 0.74609375). By doing the computation 156.75 * 0.74609375 we get 116.9501953125 EUR, which, bounded to 8 integer and 8 fractional, becomes 01110100.11110011 (=116.94921875).

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

- Item 1A: 19 points; Item 1B: 24 points; Item 1C: 24 points; Item 1D: 30 points
- Item 2: 2.5 points
- Item 3: 3.5 points

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

HINTS

- Please observe that as EXPENSES is sorted, also CONVERTED could be easily filled in a sorted way as well.
- Please observe that expenses in Eur do not need conversion and therefore should be copied in the corresponding record of CONVERTED, not forgetting to keep the same date when they occurred.
- Please observe that each conversion rate is smaller than 1, i.e. the values of each expense when converted in Euro continue fitting the (initial) 8 integer (plus the fractional) bits also of a record of CONVERTED
- The multiplication of each expense in the foreign currency to get the value in Euro is between a value on 16 bits (8 integer and 8 fractional) for the expense by a value on 8 (fractional) bits of the exchange rate; in order to use the available instruction MUL of the 8086, students are requested to think at how to "extend the currency exchange rate" to become a 16 bits value, i.e. basically choosing for the representation of the exchange rate between the representations 00000000.tttttttt and 0.tttttttt00000000
- Please observe that the final result of a 16*16 bits multiplication is a value on 32 bits and only the 8 integer plus the 8 most significant fractional have to be stored in the line of the array CONVERTED. This implies that, based on the choice operated at the previous point of this list, the student has to properly choose which bits of the 32 of the final result have to be taken and stored in CONVERTED.

REQUIREMENTS (SHARP)

- **Please choose and solve only one "Item 1" (either A, B, C or D). Then, the remaining Items 2 and 3 will operate on the array CONVERTED that has been obtained**
- It is not required to provide the optimal (shortest, most efficient, fastest) solution, but a working and clear one.
- 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.