(ITEM 1D) 2014-09-15

N EQU 50

- EXPENSES DB 3*N DUP(?) ; at most 50 entries of 3 BYTES

entry:

[0]	[1]	[2]
V D D D D D C C	E E E E E E E E	F F F F F F F F

V = 0 => NON VALID ENTRY, V = 1 => VALID ENTRY

D = DAY OF MONTH (1-31)

C = CURRENCY 00 = USD, 01 = CAD, 10 = CHF, 11 = EUR

E = AMOUNT (integer part)

F = AMOUNT (fractional part)

- EXCHANGE DB 32*3 DUP(?)

entry 0 → AVERAGE exchange rate

entry 1-31 → exchange rate FOR EACH DAY (FORMAT: 0.TTTTTTTT)

entry:

[0]	[1]	[2]
U U U U U U U U	A A A A A A A A	H H H H H H H H
USD -> EUR	CAD -> EUR	CHF -> EUR

(fractional part)

- CONVERTED DB 3*N DUP(?) ← same format of EXPENSES array

V = 1 (valid entry) E.F ← VALUE CONVERTED IN EUR

CC = 11 (converted in eur)

D = original day

CONVERSION:

	AH	AL		
(AX) 16 bits	E E E E E E E E	F F F F F F F F	×	← Original value
	CH	CL		
(CX) 16 bits	0 0 0 0 0 0 0 0	T T T T T T T T	=	← Exchange rate

(DX·AX) 32 bits ← RESULT

DH	DL	AH	AL
☒	E E E E E E E E	F F F F F F F F	☒
	↑ converted: integer part	↑ converted: fractional part (TRUNCATED)	

FRACTIONAL PART ON 8 BITS

0.	x	x	x	x	x	x	x	x
	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}

FOR A CORRECT PRINT OF THE FRACTIONAL PART, YOU MUST DIVIDE IT BY 256 (8 SHR)

```
    XOR SI, SI
convertloop:
    MOV AL, EXPENSES [SI]
    TEST AL, 10000000B              ; check validity of the entry
    JE endOfConversion
    AND AL, 00000011B               ; keep currency
    CMP AL, 00000011B               ; check if EUR
    JE eurNoConversion
    MOV AL, EXPENSES [SI]
    AND AL, 01111100B               ; keep date (index for EXCHANGE array)
    MOV CONVERTED [SI], AL          ; store the date
    OR CONVERTED [SI], 10000011B    ; set V=1 and CC = EUR
    SHR AL, 1
    SHR AL, 1                       ; AL ← 000DDDDD
    MOV CL, 3                       ; because each entry of EXCHANGE is on 3 BYTES
    MUL CX                          ; <AX> ← base index of EXCHANGE array
    XOR DI, DI
    ADD DI, AX
    MOV AL, EXPENSES [SI]
    AND AL, 00000011B               ; select the correct EXCHANGE RATE
    ADD DI, AX
    MOV CL, EXCHANGE [DI]           ; exchange rate
    XOR CH, CH                      ;<CX> ← 00000000TTTTTTTT
    MOV AH, EXPENSES [SI+1]         ; original amount (integer part)
    MOV AL, EXPENSES [SI+2]         ; original value (fractional part)
```

```
        MUL CX
        MOV CONVERTED [SI+1], DL              ; converted (integer part)
        MOV CONVERTED [SI+2], AH              ; converted (fractional part)
        JMP nextConversion
eurNoConversion:
        MOV AL, EXPENSES [SI]
        MOV CONVERTED [SI], AL
        MOV AL, EXPENSES [SI+1]
        MOV CONVERTED [SI+1], AL
        MOV AL, EXPENSES [SI+2]
        MOV CONVERTED [SI+2], AL
nextConversion:
        ; print old value and new value
        ADD SI, 3
        JMP convertloop
endOfConversion:
        .exit
```

Example arrays:

- EXPENSES

```
    DAY 2    CHF
 10001010|00010000|00101110        16,1796875 → 8,08984375

    DAY 3    EUR
 10001111|00001001|00110010        9,1953125 → 9,1953125

    DAY 5    USD
 10010100|11101011|01100001        235,37890625 → 58,8447265625

    DAY 6    CAD
 10011001|10000000|00100000        128,125 → 96,09375
```

- EXCHANGE

```
          USD        CAD        CHF

[0]   00000101|00001010|00000001

[1]   00000000|00000000|00000000

[2]   01110001|01000000|10000000        DAY 2 : CHF
                         0.5

[3]   00001000|00000010|10000000        DAY 3 : EUR (no conversion)

[4]   00000000|00000000|00000000

[5]   01000000|10000000|11100000        DAY 5: USD
       0.25

[6]   10100000|11000000|10000000        DAY 6: CAD
                 0.75
```