```
; Computer Architectures (02LSEOV)
; EXAM 2 SEPTEMBER 2013
;ERIC SERRA s209007
;Electronics Engineering - Embedded Systems @ POLITO
NUM EQU 6
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

    .MODEL small
    .STACK
    .DATA
  ;EG = MON=1 DAY=27 EMPL=12 CHARGE = 512
  ;0001 11011 01100 1000000000
  ;00011101 1011001 000000000
  ;     29      178         0
CANTEEN_CHARGES_DATABASE DB N_BYTES_OF_RECORD DUP (0)
CHARGE DW ?     ;TEMP DATA
DAY DB ?        ;TEMP DATA
EMPL DB ?       ;TEMP DATA
MON DB ?        ;TEMP DATA
SEARCH_MON DB ?     ;REQUESTED DATA
SEARCH_DAY DB ?     ;REQUESTED DATA
SEARCH_EMP DB ?     ;REQUESTED DATA
TOT DB 3 DUP (0)        ;RESULTS 23 BITS NEEDED, SINCE I HAVE TO USE MULTIPLE OF 8, 24BITS
IS SMALLEST POSSIBLE.
EMPTOT DB 3 DUP (0)     ;RESULTS
MONTOT DB 3 DUP (0)     ;RESULTS
EMPMON DW ?             ;RESULTS
DAYMON DW ?             ;RESULTS

    .CODE
    .STARTUP

;INITIAL ACQUISITION MANAGES THE CALL OF ITERATIONS

;====================================== ITEM1
=====================================================
ITEM1:
    MOV CX,N_RECORDS
    XOR SI,SI
    XOR BX,BX           ;USED REGS
    XOR DX,DX           ;TO CALCULATE TOTAL
    TOTALOOP: LEA AX,CANTEEN_CHARGES_DATABASE[SI]
        PUSH BX
        PUSH AX
        CALL read_rec       ;FUNCTION THAT RECEIVES ADDR OF A RECORD, GIVES BACK THE VALUES
        CONTAINEDIN IT , IN GLOBAL VARIABLES.
        POP AX
        POP BX
        CMP MON,0       ;IF MON=0 RECORD NON VALID->DATABASE ENDED
        JE END1
        ADD BX,CHARGE       ;MY WAY TO IMPLEMENT ADDITION 24BIT + 10BIT -> 24BIT:
```

```
                            ;AT MOST , WHEN ADDING 1000, ONE OVERFLOW OCCURS AND "CF" IS SET
        ADC DL,0            ;I USE ADC TO ADD THE (EVENTUALLY SET) CF
        ADD SI,3
        LOOP TOTALOOP
END1:MOV TOT,BL
    MOV [TOT+1],BH ;AT THE AND SAVES RESULT ON A VARIABLE
    MOV [TOT+2],DL


    JMP THE_END

;====================================== ITEM2
=========================================================
ITEM2: ;SEARCH_EMP COMES AS INPUT

    MOV CX,N_RECORDS
    XOR SI,SI
    XOR BX,BX
    XOR DX,DX
    EMPLOOP: LEA AX,CANTEEN_CHARGES_DATABASE[SI]
        PUSH BX
        PUSH AX
        CALL read_rec
        POP AX
        POP BX
        CMP MON,0       ;IF MON=0 RECORD NON VALID->DATABASE ENDED
        JE END2
        MOV AL,SEARCH_EMP
        CMP EMPL,AL             ;IF THE EMPLOYEE IS THE RIGHT ONE, ADD THE AMOUNT TO HIS
        TOTAL CHARGE
        JNE NEXT_EMP
        ADD BX,CHARGE
        ADC DL,0
        NEXT_EMP:
        ADD SI,3
        LOOP EMPLOOP
END2:MOV EMPTOT,BL
    MOV EMPTOT+1,BH
    MOV [EMPTOT+2],DL

    JMP THE_END

;====================================== ITEM3
=========================================================
ITEM3: ;SEARCH_MON COMES AS INPUT

LEA DX,SEARCH1      ;RECEIVE ANSWER
    MOV AH,0AH
    INT 21H
    MOV AL,SEARCH1[2]       ;CONV. TO NUMBER
    SUB AL,'0'
    MOV CL,10
    MUL CL
    MOV CL,SEARCH1[3]
    SUB CL,'0'
    ADD AL,CL
    MOV SEARCH_MON,AL
```

```
    MOV CX,N_RECORDS
    XOR SI,SI
    XOR BX,BX
    XOR DX,DX
    MONLOOP: LEA AX,CANTEEN_CHARGES_DATABASE[SI]
        PUSH BX
        PUSH AX
        CALL read_rec
        POP AX
        POP BX
        CMP MON,0       ;IF MON=0 RECORD NON VALID->DATABASE ENDED
        JE END3
        MOV AL,SEARCH_MON
        CMP MON,AL      ;IF THE MONTH IS THE RIGHT ONE, ADD THE AMOUNT TO HIS TOTAL CHARGE
        JNE NEXT_MON
        ADD BX,CHARGE
        ADC DL,0
        NEXT_MON:
        ADD SI,3
        LOOP MONLOOP
END3:MOV MONTOT,BL
    MOV MONTOT+1,BH
    MOV [MONTOT+2],DL

    JMP THE_END
;==================================== ITEM4
================================================
ITEM4:;SEARCH_EMP AND SEARCH_MON COME AS INPUTS

    MOV CX,N_RECORDS
    XOR SI,SI
    XOR DX,DX
    EMPMONLOOP: LEA AX,CANTEEN_CHARGES_DATABASE[SI]
        PUSH AX
        CALL read_rec
        POP AX
        CMP MON,0       ;IF MON=0 RECORD NON VALID->DATABASE ENDED
        JE END4
        MOV AL,SEARCH_EMP
        MOV AH,SEARCH_MON
        MOV BL,EMPL             ;GOOD WAY TO TEST 2 THINGS TOGETHER , WITH ONE COMPARE
        MOV BH,MON
        CMP AX,BX           ;IF BOTH MONTH AND EMPLOYEE ARE CORRECT, ADD THE CURRENT CHARGE
        TO THE TOTAL
        JNE NEXT_REC
        ADD DX,CHARGE
        NEXT_REC:
        ADD SI,3
        LOOP EMPMONLOOP
END4:MOV EMPMON,DX

JMP THE_END
;========================================= ITEM5
=================================================
ITEM5: ;SEARCH_DAY AND SEARCH_MON COME AS INPUTS
```

```
    MOV CX,N_RECORDS
    XOR SI,SI
    XOR DX,DX
    DAYMONLOOP: LEA AX,CANTEEN_CHARGES_DATABASE[SI]
        PUSH AX
        CALL read_rec
        POP AX
        CMP MON,0        ;IF MON=0 RECORD NON VALID->DATABASE ENDED
        JE END5
        CMP EMPL,0
        JE END5
        MOV AL,SEARCH_DAY
        MOV AH,SEARCH_MON
        MOV BL,DAY
        MOV BH,MON
        CMP AX,BX            ;IF BOTH MONTH AND DAY ARE CORRECT, ADD THE CURRENT CHARGE TO
        THE TOTAL
        JNE NEXT_DAYMON
        ADD DX,CHARGE
        NEXT_DAYMON:
        ADD SI,3
        LOOP DAYMONLOOP
END5:MOV DAYMON,DX


JMP THE_END
;=================================================================================================

THE_END:
    .EXIT

read_rec PROC


        PUSH BP
        MOV BP,SP
        MOV BX,[BP+4] ;ADDRESS OF PUSHED RECORD'S 1ST BYTE
        PUSH CX

        ;SINCE DATA IS SCRAMBLED UPON 3 BYTES I MADE THIS PROCEDURE TO MAKE SOME ORDER
        ;TO MAKE THINGS EASIER : I PUT EVERY PIECE OF THE RECORD IN A DIFFERENT GLOBAL
        VARIABLE
        ; MON   DAY   EMPL  CHARGE
        ;|----|-----|-----|----------|
        ;|---- ----|--------|--------|
        ; 3RD BYTE    2ND      1ST
        ; [BX+2]     [BX+1]    [BX]

        ;EXTRACT CHARGES
        MOV AX, [BX]
        AND AX,03FFH ;PUTS TO ZERO USELESS (NON REFERRED TO CHARGES) BITS  --> EXTRACTS THE
        VALUE FOR CHARGES OF THIS RECORD
        MOV CHARGE,AX
        ;EXTRACT EMPLOYEE
        MOV AL,[BX+1]  ;(RECORD'S 2ND BYTE)
        MOV CL,2
        SHR AX,CL
```

```
        AND AL,00011111B
        MOV EMPL,AL
        ;EXTRACT DAY
        MOV AX,[BX+1] ; AH= 3RD BYTE , AL= 2ND BYTE
        MOV CL,7
        SHR AX,CL
        AND AL,00011111B
        MOV DAY,AL
        ;EXTRACT MONTH
        MOV AL,[BX+2]
        MOV CL,4
        SHR AX,CL
        AND AL,00001111B
        MOV MON,AL

        POP CX
        POP BP
        RET
read_rec ENDP


    END
```