ΣΥΣΤΗΜΑΤΑ ΜΙΚΡΟΥΠΟΛΟΓΙΣΤΩΝ
ΣΕΙΡΑ ΑΣΚΗΣΕΩΝ 5

Ζαρίφης Νικόλαος ΑΜ 03112178
Ιωάννης Ορούτζογλου ΑΜ 03112124

Άσκηση 1: Στις ασκήσεις χρησιμοποιώ πότε καιφαλέα πότε μικρά γιατί δεν ειναι case-sensitive. Επίσεις τα δυο ερώτηματα τις ασκήσεις 1 , τα έχω κάνει σε ένα πρόγραμμα.

```
DATA_SEG SEGMENT ; for store min max
FIS DW 0;
SEC DW 11111111b ;max number
DW 11111111b
DATA_SEG ENDS
main:
ASSUME DS:DATA_SEG
mov si,0 ; lets say that the data mem of the number is in zero.
mov bx,0
mov cx,5
mov dl,0
loopa:
mov ax,[si]    ; si is used for the data in mem
push ax
push bx
mov bx,OFFSET FIS     ; take prev max
cmp ax,bx
jg change        ; check if it is greater to change val
min:
mov bx,OFFSET SEC
cmp ax,bx
jg rest
MOV OFFSET SEC,ax    ;update min
rest:
pop bx
pop ax
test ax,1  ; test if it is even
jz  even
jmp notev:
even:
add bx,ax
inc dl
notev:
inc si
inc si ; 2 times because we have 16 bit vals

loop loopa
mov ax,bx
mov bx,dx
xor dx,dx ; zero dx
div bx   ;akereo meros
```

```
; there we can put exit, or something else, i will leave it empty

change:
MOV OFFSET FIS,ax
jmp min
```

Άσκηση 2

Εδώ μέρικα είναι απο το βίβλιο κι άλλα απο την άσκηση 3 πιο κάτω.

```
PRINT_HEX PROC NEAR
   PUSHF
   PUSH bx
   CMP BL,9
   JG ADDK
   ADD BL,30H ; 0 Char
   JMP ADDK2
ADDK:
ADD BL,37H      ; CHARACTER A= 10
ADDK2:
   PUSH dx
   PUSH ax
   MOV DL,BL
   MOV AH,2
   INT 21H
   POP ax
   POP dx
   POP bx
   POPF
   RET
PRINT_HEX ENDP
MACRO PRINT CHAR          ; book code
   PUSH dx
   PUSH ax
   MOV DL,CHAR
   MOV AH,2
   INT 21H
   POP ax
   POP dx
ENDM
READ MACRO
   mov ah,8
   int 21h
ENDM
READ_2D PROC NEAR
   PUSH ax
   push dx
   push bx
```

```
    call READ_DEC ; read until a digits
    mov dx,0
    mov ah,0
    mov bl,10
    mul bl
    mov cl,al
    INC di     ; next place to store
    call READ_DEC
    mov ah,0
    add cl,al ;
    pop bx
    pop dx
    pop ax
READ_2D ENDP
PRINT_STR MACRO STRING
    push dx
    push ax
    mov dx, OFFSET STRING
    mov ah,9
    int 21h
    pop ax
    pop dx
ENDM
STORE MACRO
    MOV di,al
    ENDM
READ_DEC PROC NEAR
    pushf
    notnumber:
    READ
    cmp al,30h   ; check if it is a digit
    jl notnumber
    cmp al,39h
    jg notnumber
    mov ah,0
    mov di,ax      ;store in data as ascii code
    sub al,30h
    popf

READ_DEC ENDP
DATA_SEG SEGMENT
    FIRST DB  "x="
    XVAL1 DW '0'
    DW '0'
    DB " y="
    YVAL1 DW '0'
    DW '0'
    DB 0AH,0DH,'$'  ; NEW LINE
    PRINTSUB DB "x+y=$"
    PRINTMINUS DB "x-y=-$"
    PRINTMINUS2 DB "x-y=-$"
    NEWL DB 0AH,0DH,'$'  ; NEW LINE

DATA_SEG ENDS
```

```
CODE_SEG SEGMENT
   ASSUME CS:CODE_SEG,DS:DATA_SEG
   MAIN PROC FAR
      mov di, OFFSET XVAL1
     call READ_2D
      mov bl,cl
      mov di, OFFSET YVAL1
     call READ_2D
      PRINT_STR FIRST ; print the input
      push bx ;store value
      add bl, cl ; now we have   the sum
      PRINT_STR PRINTSUB
      call PRINT_HEX
      pop bx
      cmp bl,cl
      jg printno
      PRINT_STR PRINTMINUS
      sub cl,bl
      mov bl,cl

      jmp end1
      printno:
      PRINT_STR PRINTMINUS2
       sub bl,cl
       end1:
        call PRINT_HEX
        PRINT_STR NEWL
  CODE_SEG ENDS
   END MAIN
```

Άσκηση 3:

```
PRINT_HEX PROC NEAR
   PUSHF
   PUSH bx
   CMP BL,9
   JG ADDK
   ADD BL,30H ; 0 Char
   JMP ADDK2
ADDK:
ADD BL,37H      ; CHARACTER A= 10
ADDK2:
   PUSH dx
   PUSH ax
   MOV DL,BL
   MOV AH,2
   INT 21H
   POP ax
   POP dx
   POP bx
   POPF
   RET
PRINT_HEX ENDP
```

```
PRINT_OCT PROC NEAR
   PUSHF
   PUSH BX
   PUSH AX
   push dx
   mov al,bl
   mov ah,3
   RCL BL,2
   and bl,ah    ; take first 2 bits (actually 2 MSBs and mask)

   mov dl,bl ; time to print
   add dl,30H ;char 0
   mov ah,2
   int 21H
   push cx
   mov cx,2
   loopa1:
   mov bl,al ; time for the rest of the number  3 bits per octal number
   rcl bl,5
   mov ah,3
   and bl,ah
   mov dl,bl
   add dl,30H
   mov ah,2
   int 21H
   loop loopa1
   POP CX
   POP DX
   POP AX
   POP BX
   POPF
   RET
PRINT_OCT ENDP

PRINT_BIN PROC NEAR
   PUSHF
   PUSH BX
   PUSH AX
   push dx
   push cx
   mov cx,8
   loopa:
    rcl bl ,1 ; one bit per time
   mov ah,1
   and bl,ah
   mov dl,bl
   add dl,30H ; go to number 0 + offset of number (o or 1)
   mov ah,2H
   int 21H
   loop loopa
   POP CX
   POP DX
   POP AX
   POP BX
   POPF
   RET
PRINT_BIN ENDP
MACRO PRINT CHAR
   PUSH DX
   PUSH AX
   MOV DL,CHAR
```

```
    MOV AH,2
    INT 21H
    POP AX
    POP DX
ENDM
READ MACRO
    mov ah,8
    int 21h
ENDM
READ_DEC PROC NEAR
    pushf
    notnumber:
    READ
    cmp al,30h   ; check if it is a digit
    jl notnumber
    cmp al,39h
    jg notnumber
    sub al,30h
    popf

READ_DEC ENDP
START:
    call READ_DEC ; read until a digits
    mov dx,0
    mov ah,0
    mov bl,10
    mul bl
    mov cl,al
    call READ_DEC
    mov ah,0
    add cl,al ; now i have the number
    mov bl,al
    call PRINT_HEX
    PRINT '='
    call PRINT_OCT
    PRINT '='
    call PRINT_BIN
    PRINT 0AH
    PRINT 0DH ; new line
```