Συστήματα Μικρουπολογιστών

5η Ομάδα Ασκήσεων

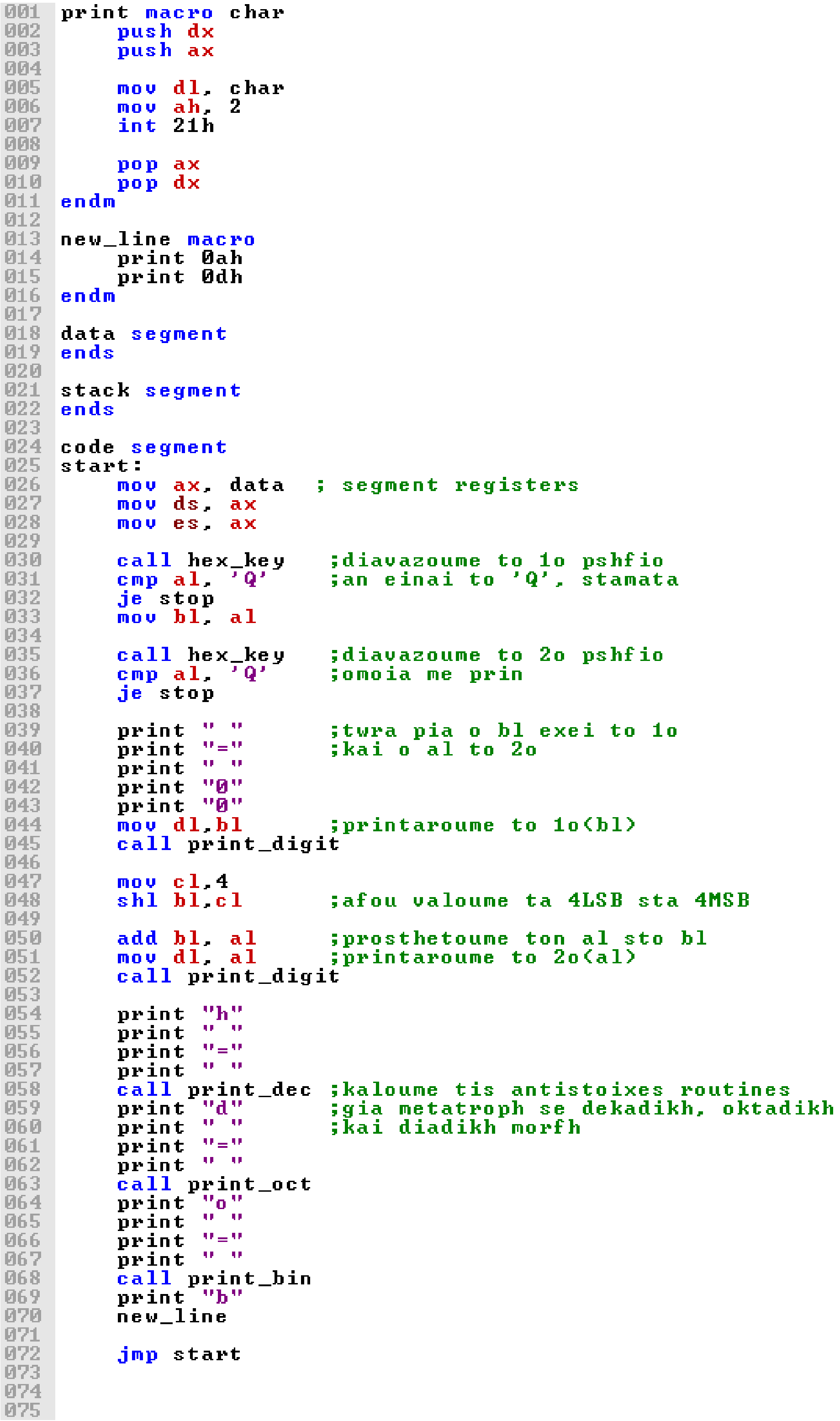
Μετζάκης Ιωάννης

Α.Μ.: 03116202

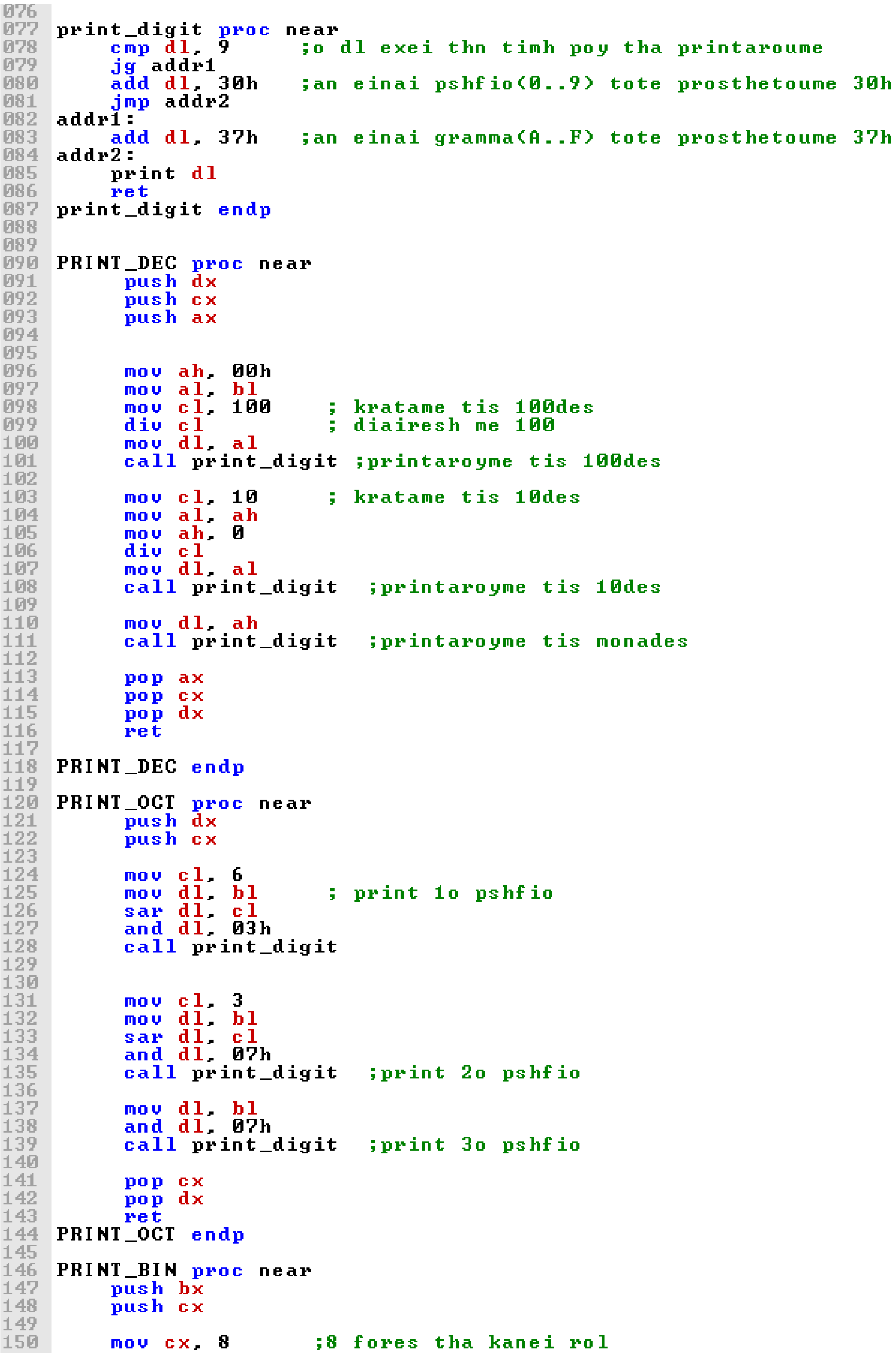
ΣΗΜΜΥ 8ο

1η ΑΣΚΗΣΗ

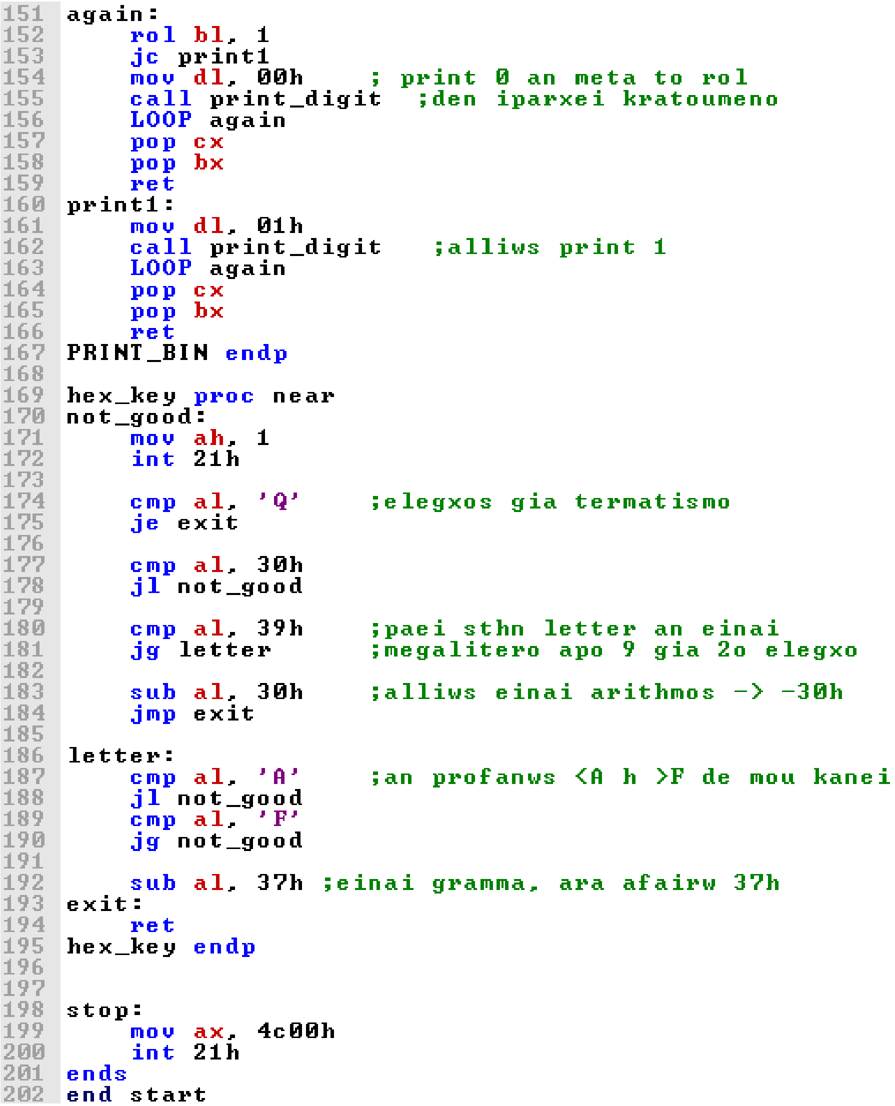
Παρακάτω ο κώδικας σε **assembly** με τα απαραίτητα σχόλια:



```
print macro char
    push dx
    push ax

    mov dl, char
    mov ah, 2
    int 21h

    pop ax
    pop dx
endm

new_line macro
    print 0ah
    print 0dh
endm

data segment
ends

stack segment
ends

code segment
start:
    mov ax, data  ; segment registers
    mov ds, ax
    mov es, ax

    call hex_key   ;diavazoume to 1o pshfio
    cmp al, 'Q'    ;an einai to 'Q', stamata
    je stop
    mov bl, al

    call hex_key   ;diavazoume to 2o pshfio
    cmp al, 'Q'    ;omoia me prin
    je stop

    print " "      ;twra pia o bl exei to 1o
    print "="      ;kai o al to 2o
    print " "
    print "0"
    print "0"
    mov dl,bl      ;printaroume to 1o(bl)
    call print_digit

    mov cl,4
    shl bl,cl      ;afou valoume ta 4LSB sta 4MSB

    add bl, al     ;prosthetoume ton al sto bl
    mov dl, al     ;printaroume to 2o(al)
    call print_digit

    print "h"
    print " "
    print "="
    print " "
    call print_dec ;kaloume tis antistoixes routines
    print "d"      ;gia metatroph se dekadikh, oktadikh
    print " "      ;kai diadikh morfh
    print "="
    print " "
    call print_oct
    print "o"
    print " "
    print "="
    print " "
    call print_bin
    print "b"
    new_line

    jmp start


```

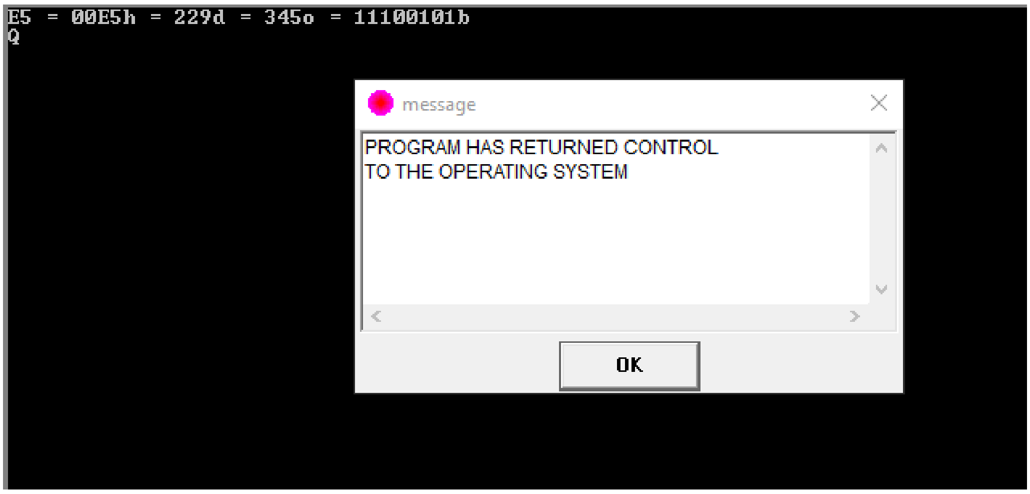


```

print_digit proc near
    cmp dl, 9     ;o dl exei thn timh poy tha printaroume
    jg addr1
    add dl, 30h   ;an einai pshfio(0..9) tote prosthetoume 30h
    jmp addr2
addr1:
    add dl, 37h   ;an einai gramma(A..F) tote prosthetoume 37h
addr2:
    print dl
    ret
print_digit endp


PRINT_DEC proc near
     push dx
     push cx
     push ax


     mov ah, 00h
     mov al, bl
     mov cl, 100    ; kratame tis 100des
     div cl         ; diairesh me 100
     mov dl, al
     call print_digit ;printaroyme tis 100des

     mov cl, 10     ; kratame tis 10des
     mov al, ah
     mov ah, 0
     div cl
     mov dl, al
     call print_digit  ;printaroyme tis 10des

     mov dl, ah
     call print_digit  ;printaroyme tis monades

     pop ax
     pop cx
     pop dx
     ret

PRINT_DEC endp

PRINT_OCT proc near
     push dx
     push cx

     mov cl, 6
     mov dl, bl     ; print 1o pshfio
     sar dl, cl
     and dl, 03h
     call print_digit


     mov cl, 3
     mov dl, bl
     sar dl, cl
     and dl, 07h
     call print_digit  ;print 2o pshfio

     mov dl, bl
     and dl, 07h
     call print_digit  ;print 3o pshfio

     pop cx
     pop dx
     ret
PRINT_OCT endp

PRINT_BIN proc near
    push bx
    push cx

    mov cx, 8      ;8 fores tha kanei rol
```



```
again:
        rol bl, 1
        jc print1
        mov dl, 00h     ; print 0 an meta to rol
        call print_digit   ;den iparxei kratoumeno
        LOOP again
        pop cx
        pop bx
        ret
print1:
        mov dl, 01h
        call print_digit    ;alliws print 1
        LOOP again
        pop cx
        pop bx
        ret
PRINT_BIN endp

hex_key proc near
not_good:
        mov ah, 1
        int 21h

        cmp al, 'Q'     ;elegxos gia termatismo
        je exit

        cmp al, 30h
        jl not_good

        cmp al, 39h     ;paei sthn letter an einai
        jg letter       ;megalitero apo 9 gia 2o elegxo

        sub al, 30h     ;alliws einai arithmos -> -30h
        jmp exit

letter:
        cmp al, 'A'     ;an profanws <A h >F de mou kanei
        jl not_good
        cmp al, 'F'
        jg not_good

        sub al, 37h ;einai gramma, ara afairw 37h
exit:
        ret
hex_key endp


stop:
        mov ax, 4c00h
        int 21h
ends
end start
```

Παρακάτω βρίσκεται ένα στιγμιότυπο από το output της οθόνης για είσοδο *E5*, και μετέπειτα εισαγωγή του χαρακτήρα *'Q'* για τερματισμό:

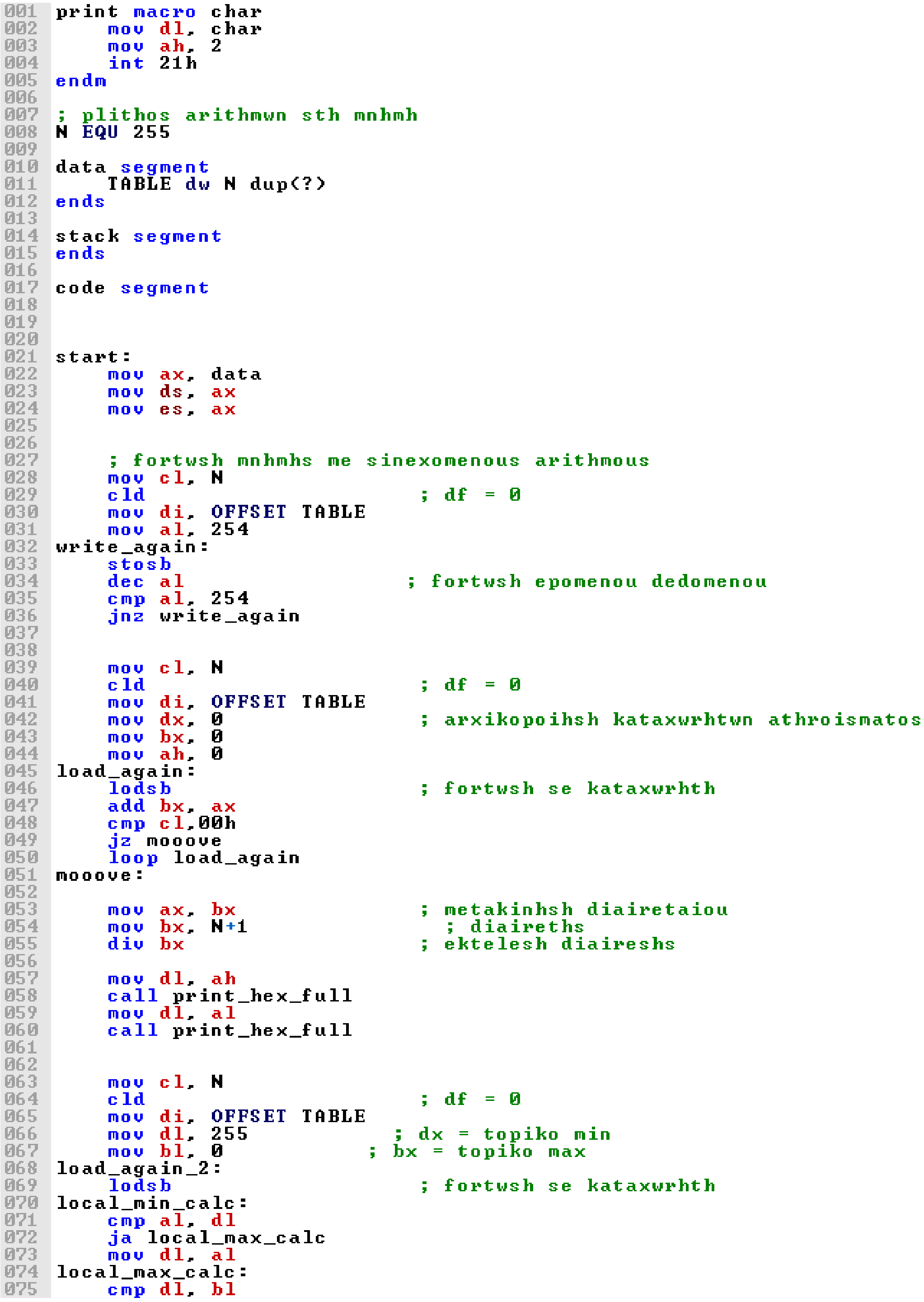


```
E5 = 00E5h = 229d = 345o = 11100101b
Q
```

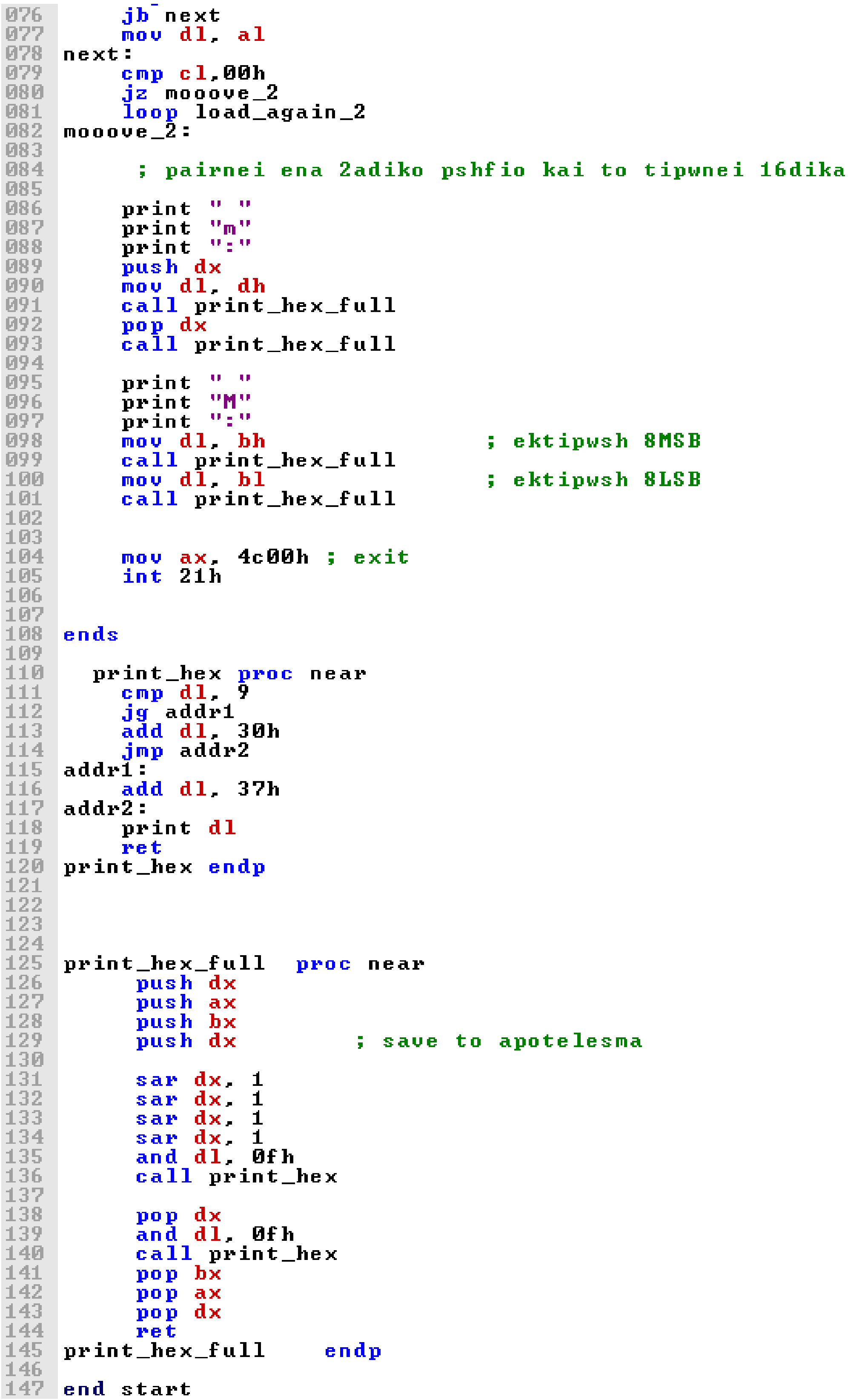
PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM

2η ΑΣΚΗΣΗ

Παρακάτω ο κώδικας σε **assembly** με τα απαραίτητα σχόλια:



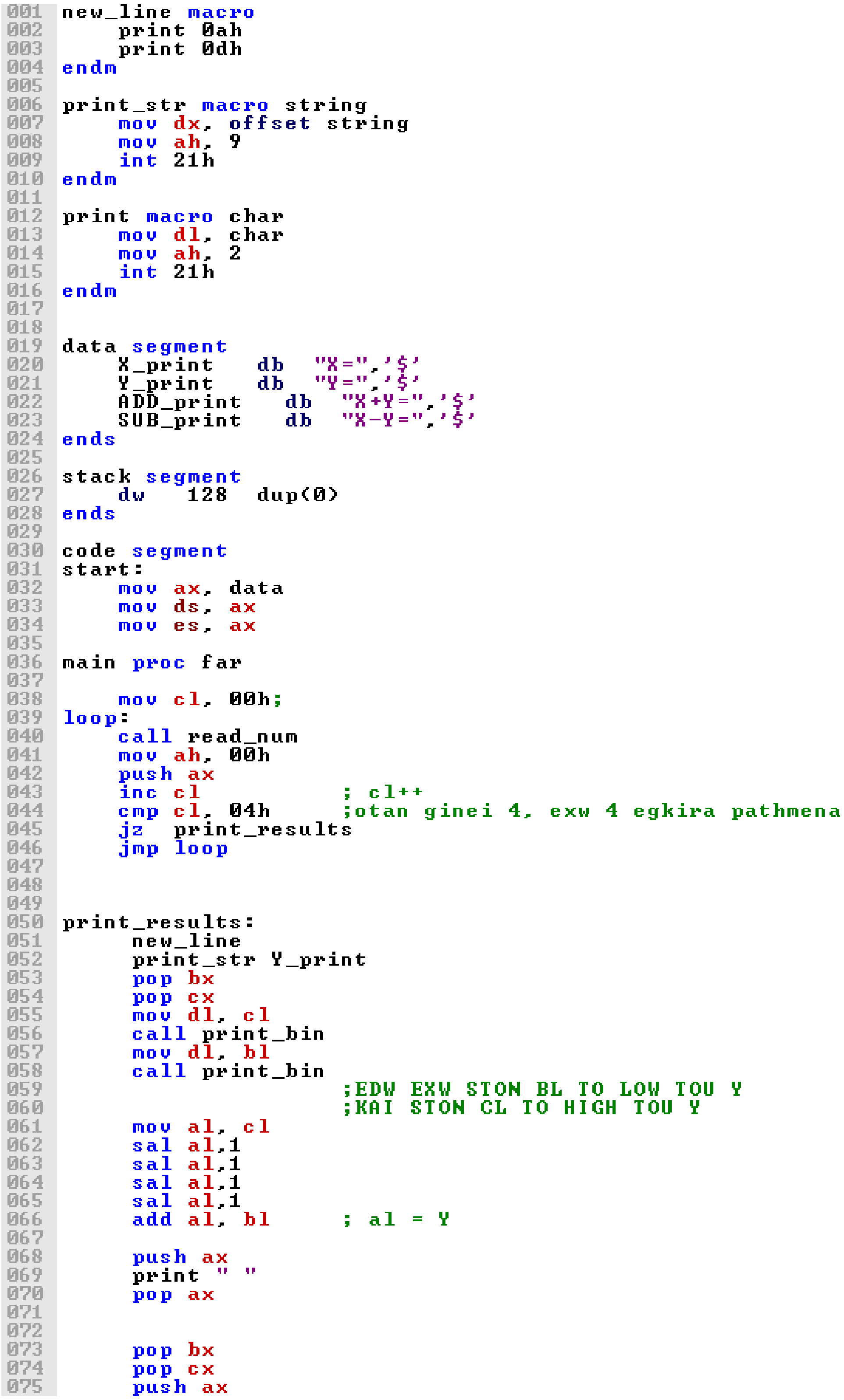
```
print macro char
    mov dl, char
    mov ah, 2
    int 21h
endm

; plithos arithmwn sth mnhmh
N EQU 255

data segment
    TABLE dw N dup(?)
ends

stack segment
ends

code segment



start:
    mov ax, data
    mov ds, ax
    mov es, ax


    ; fortwsh mnhmhs me sinexomenous arithmous
    mov cl, N
    cld                     ; df = 0
    mov di, OFFSET TABLE
    mov al, 254
write_again:
    stosb
    dec al                 ; fortwsh epomenou dedomenou
    cmp al, 254
    jnz write_again


    mov cl, N
    cld                     ; df = 0
    mov di, OFFSET TABLE
    mov dx, 0               ; arxikopoihsh kataxwrhtwn athroismatos
    mov bx, 0
    mov ah, 0
load_again:
    lodsb                   ; fortwsh se kataxwrhth
    add bx, ax
    cmp cl,00h
    jz mooove
    loop load_again
mooove:

    mov ax, bx              ; metakinhsh diairetaiou
    mov bx, N+1               ; diaireths
    div bx                  ; ektelesh diaireshs

    mov dl, ah
    call print_hex_full
    mov dl, al
    call print_hex_full


    mov cl, N
    cld                     ; df = 0
    mov di, OFFSET TABLE
    mov dl, 255           ; dx = topiko min
    mov bl, 0           ; bx = topiko max
load_again_2:
    lodsb                   ; fortwsh se kataxwrhth
local_min_calc:
    cmp al, dl
    ja local_max_calc
    mov dl, al
local_max_calc:
    cmp dl, bl
```



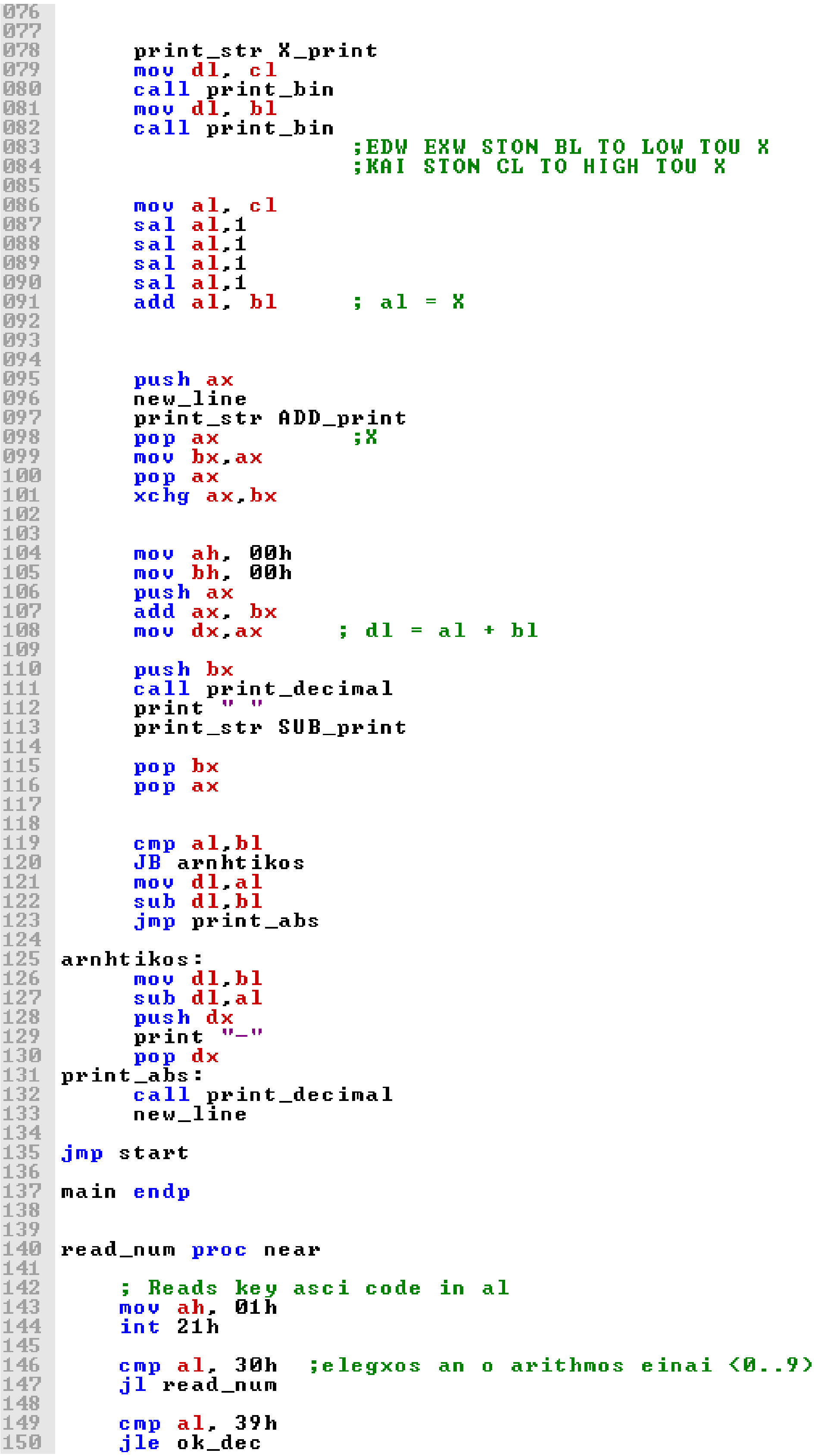
```
    jb next
    mov dl, al
next:
    cmp cl,00h
    jz mooove_2
    loop load_again_2
mooove_2:

     ; pairnei ena 2adiko pshfio kai to tipwnei 16dika

    print " "
    print "m"
    print ":"
    push dx
    mov dl, dh
    call print_hex_full
    pop dx
    call print_hex_full

    print " "
    print "M"
    print ":"
    mov dl, bh               ; ektipwsh 8MSB
    call print_hex_full
    mov dl, bl               ; ektipwsh 8LSB
    call print_hex_full


    mov ax, 4c00h ; exit
    int 21h


ends

  print_hex proc near
    cmp dl, 9
    jg addr1
    add dl, 30h
    jmp addr2
addr1:
    add dl, 37h
addr2:
    print dl
    ret
print_hex endp




print_hex_full  proc near
     push dx
     push ax
     push bx
     push dx        ; save to apotelesma

     sar dx, 1
     sar dx, 1
     sar dx, 1
     sar dx, 1
     and dl, 0fh
     call print_hex

     pop dx
     and dl, 0fh
     call print_hex
     pop bx
     pop ax
     pop dx
     ret
print_hex_full    endp

end start
```

3η ΑΣΚΗΣΗ

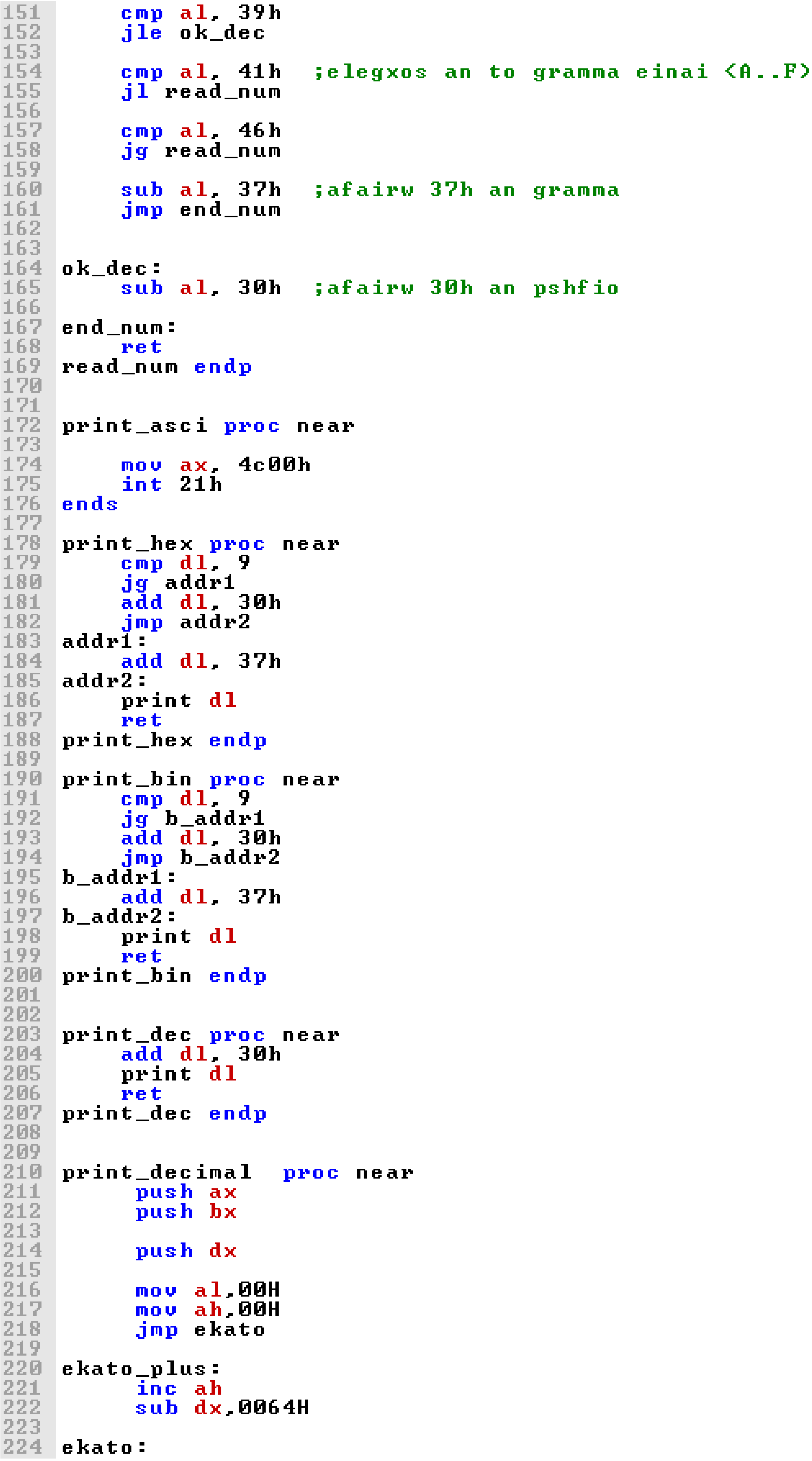
Παρακάτω ο κώδικας σε **assembly** με τα απαραίτητα σχόλια:



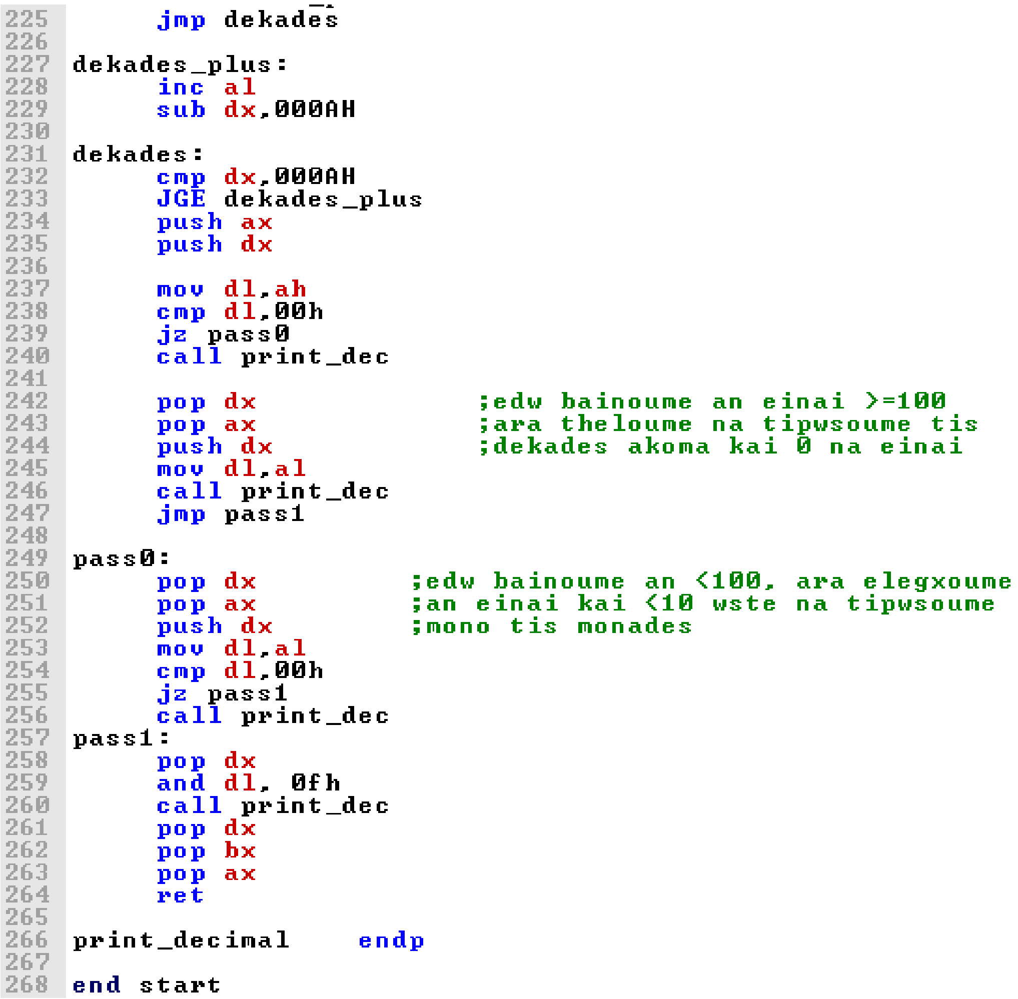
```
new_line macro
    print 0ah
    print 0dh
endm

print_str macro string
    mov dx, offset string
    mov ah, 9
    int 21h
endm

print macro char
    mov dl, char
    mov ah, 2
    int 21h
endm


data segment
    X_print   db   "X=",'$'
    Y_print   db   "Y=",'$'
    ADD_print   db   "X+Y=",'$'
    SUB_print   db   "X-Y=",'$'
ends

stack segment
    dw   128  dup(0)
ends

code segment
start:
    mov ax, data
    mov ds, ax
    mov es, ax

main proc far

    mov cl, 00h;
loop:
    call read_num
    mov ah, 00h
    push ax
    inc cl          ; cl++
    cmp cl, 04h     ;otan ginei 4, exw 4 egkira pathmena
    jz  print_results
    jmp loop



print_results:
     new_line
     print_str Y_print
     pop bx
     pop cx
     mov dl, cl
     call print_bin
     mov dl, bl
     call print_bin
                    ;EDW EXW STON BL TO LOW TOU Y
                    ;KAI STON CL TO HIGH TOU Y
     mov al, cl
     sal al,1
     sal al,1
     sal al,1
     sal al,1
     add al, bl     ; al = Y

     push ax
     print " "
     pop ax


     pop bx
     pop cx
     push ax
```



```


     print_str X_print
     mov dl, cl
     call print_bin
     mov dl, bl
     call print_bin
                    ;EDW EXW STON BL TO LOW TOU X
                    ;KAI STON CL TO HIGH TOU X

     mov al, cl
     sal al,1
     sal al,1
     sal al,1
     sal al,1
     add al, bl     ; al = X



     push ax
     new_line
     print_str ADD_print
     pop ax         ;X
     mov bx,ax
     pop ax
     xchg ax,bx


     mov ah, 00h
     mov bh, 00h
     push ax
     add ax, bx
     mov dx,ax     ; dl = al + bl

     push bx
     call print_decimal
     print " "
     print_str SUB_print

     pop bx
     pop ax


     cmp al,bl
     JB arnhtikos
     mov dl,al
     sub dl,bl
     jmp print_abs

arnhtikos:
     mov dl,bl
     sub dl,al
     push dx
     print "-"
     pop dx
print_abs:
     call print_decimal
     new_line

jmp start

main endp


read_num proc near

    ; Reads key asci code in al
    mov ah, 01h
    int 21h

    cmp al, 30h  ;elegxos an o arithmos einai <0..9>
    jl read_num

    cmp al, 39h
    jle ok_dec
```

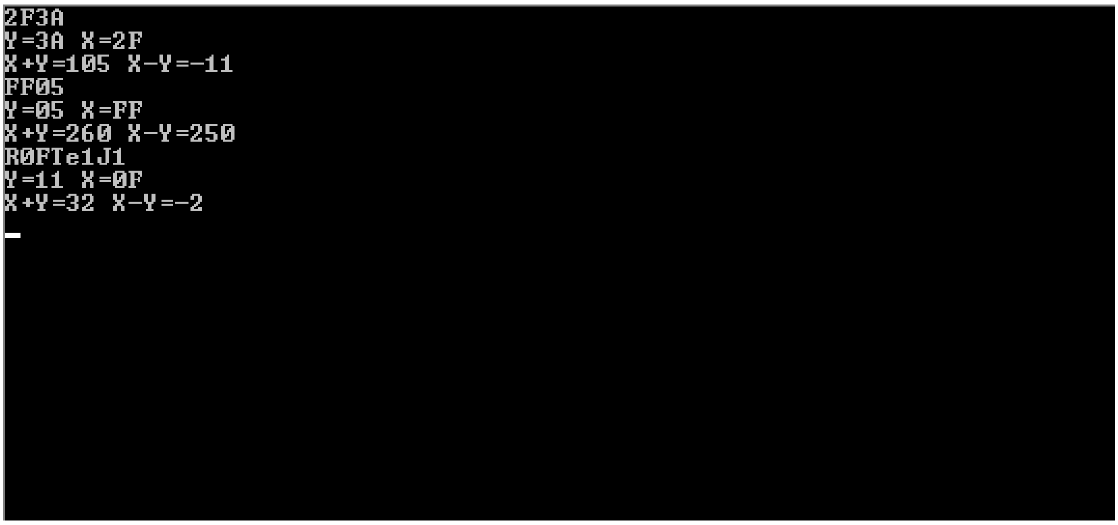


```
    cmp al, 39h
    jle ok_dec

    cmp al, 41h  ;elegxos an to gramma einai <A..F>
    jl read_num

    cmp al, 46h
    jg read_num

    sub al, 37h  ;afairw 37h an gramma
    jmp end_num


ok_dec:
    sub al, 30h  ;afairw 30h an pshfio

end_num:
    ret
read_num endp


print_asci proc near

    mov ax, 4c00h
    int 21h
ends

print_hex proc near
    cmp dl, 9
    jg addr1
    add dl, 30h
    jmp addr2
addr1:
    add dl, 37h
addr2:
    print dl
    ret
print_hex endp

print_bin proc near
    cmp dl, 9
    jg b_addr1
    add dl, 30h
    jmp b_addr2
b_addr1:
    add dl, 37h
b_addr2:
    print dl
    ret
print_bin endp


print_dec proc near
    add dl, 30h
    print dl
    ret
print_dec endp


print_decimal  proc near
     push ax
     push bx

     push dx

     mov al,00H
     mov ah,00H
     jmp ekato

ekato_plus:
     inc ah
     sub dx,0064H

ekato:
```



```
        jmp dekades

dekades_plus:
        inc al
        sub dx,000AH

dekades:
        cmp dx,000AH
        JGE dekades_plus
        push ax
        push dx

        mov dl,ah
        cmp dl,00h
        jz pass0
        call print_dec

        pop dx                  ;edw bainoume an einai >=100
        pop ax                  ;ara theloume na tipwsoume tis
        push dx                 ;dekades akoma kai 0 na einai
        mov dl,al
        call print_dec
        jmp pass1

pass0:
        pop dx              ;edw bainoume an <100, ara elegxoume
        pop ax              ;an einai kai <10 wste na tipwsoume
        push dx             ;mono tis monades
        mov dl,al
        cmp dl,00h
        jz pass1
        call print_dec
pass1:
        pop dx
        and dl, 0fh
        call print_dec
        pop dx
        pop bx
        pop ax
        ret

print_decimal    endp

end start
```

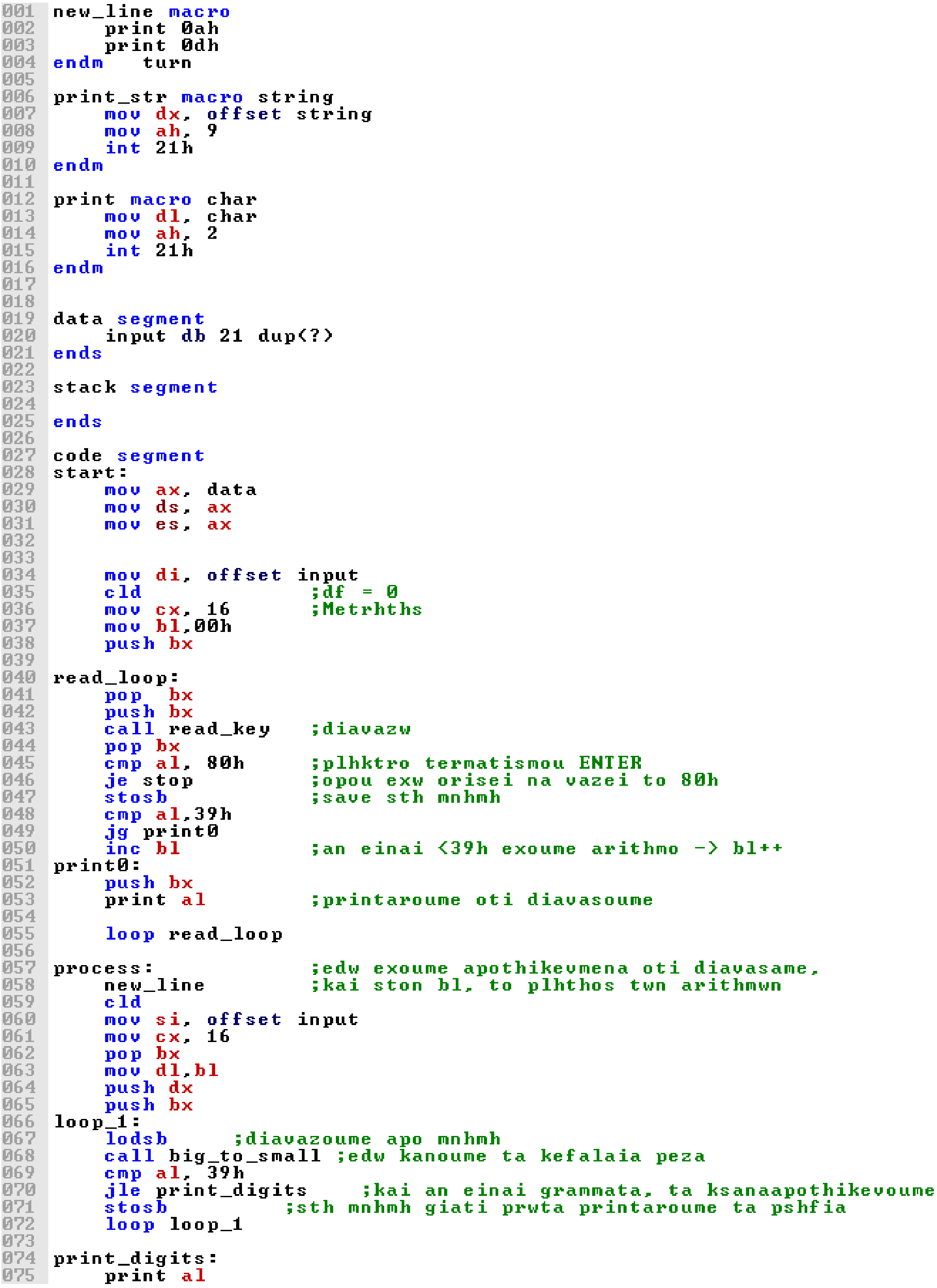
Παρακάτω βρίσκεται ένα στιγμιότυπο από το output της οθόνης για διαδοχικές εισόδους *2F3A* (παράδειγμα εκφώνησης), *FF05* και *R0FTe1J1* (για να φανεί ο έλεγχος των έγκυρων HEX ψηφίων, δηλαδή των 0F11):



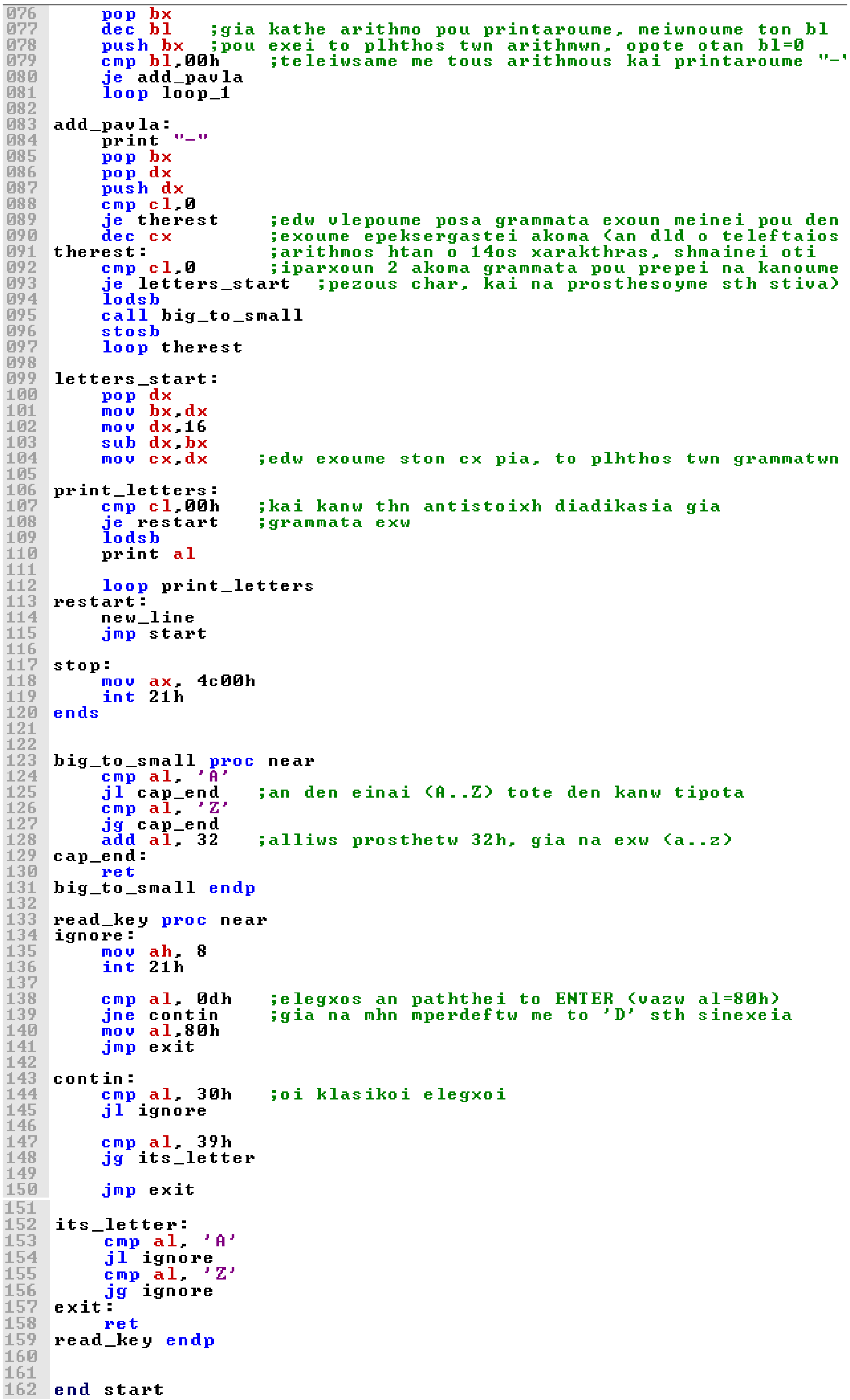
```
2F3A
Y=3A X=2F
X+Y=105 X-Y=-11
FF05
Y=05 X=FF
X+Y=260 X-Y=250
R0FTe1J1
Y=11 X=0F
X+Y=32 X-Y=-2
```

4η ΑΣΚΗΣΗ

Παρακάτω ο κώδικας σε **assembly** με τα απαραίτητα σχόλια:

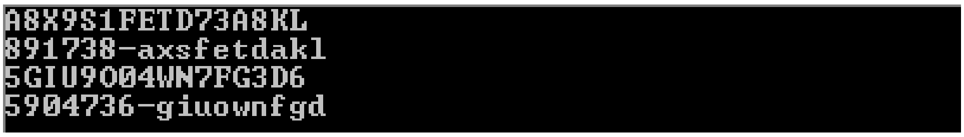


```
new_line macro
    print 0ah
    print 0dh
endm   turn

print_str macro string
    mov dx, offset string
    mov ah, 9
    int 21h
endm

print macro char
    mov dl, char
    mov ah, 2
    int 21h
endm


data segment
    input db 21 dup(?)
ends

stack segment

ends

code segment
start:
    mov ax, data
    mov ds, ax
    mov es, ax


    mov di, offset input
    cld             ;df = 0
    mov cx, 16      ;Metrhths
    mov bl,00h
    push bx

read_loop:
    pop  bx
    push bx
    call read_key   ;diavazw
    pop bx
    cmp al, 80h     ;plhktro termatismou ENTER
    je stop         ;opou exw orisei na vazei to 80h
    stosb           ;save sth mnhmh
    cmp al,39h
    jg print0
    inc bl          ;an einai <39h exoume arithmo -> bl++
print0:
    push bx
    print al        ;printaroume oti diavasoume

    loop read_loop

process:            ;edw exoume apothikevmena oti diavasame,
    new_line        ;kai ston bl, to plhthos twn arithmwn
    cld
    mov si, offset input
    mov cx, 16
    pop bx
    mov dl,bl
    push dx
    push bx
loop_1:
    lodsb     ;diavazoume apo mnhmh
    call big_to_small ;edw kanoume ta kefalaia peza
    cmp al, 39h
    jle print_digits    ;kai an einai grammata, ta ksanaapothikevoume
    stosb         ;sth mnhmh giati prwta printaroume ta pshfia
    loop loop_1

print_digits:
    print al
```



```
    pop bx
    dec bl   ;gia kathe arithmo pou printaroume, meiwnoume ton bl
    push bx  ;pou exei to plhthos twn arithmwn, opote otan bl=0
    cmp bl,00h    ;teleiwsame me tous arithmous kai printaroume "-'
    je add_pavla
    loop loop_1

add_pavla:
    print "-"
    pop bx
    pop dx
    push dx
    cmp cl,0
    je therest    ;edw vlepoume posa grammata exoun meinei pou den
    dec cx        ;exoume epeksergastei akoma (an dld o teleftaios
therest:          ;arithmos htan o 14os xarakthras, shmainei oti
    cmp cl,0      ;iparxoun 2 akoma grammata pou prepei na kanoume
    je letters_start  ;pezous char, kai na prosthesoyme sth stiva)
    lodsb
    call big_to_small
    stosb
    loop therest

letters_start:
    pop dx
    mov bx,dx
    mov dx,16
    sub dx,bx
    mov cx,dx    ;edw exoume ston cx pia, to plhthos twn grammatwn

print_letters:
    cmp cl,00h   ;kai kanw thn antistoixh diadikasia gia
    je restart   ;grammata exw
    lodsb
    print al

    loop print_letters
restart:
    new_line
    jmp start

stop:
    mov ax, 4c00h
    int 21h
ends


big_to_small proc near
    cmp al, 'A'
    jl cap_end   ;an den einai (A..Z) tote den kanw tipota
    cmp al, 'Z'
    jg cap_end
    add al, 32   ;alliws prosthetw 32h, gia na exw (a..z)
cap_end:
    ret
big_to_small endp

read_key proc near
ignore:
    mov ah, 8
    int 21h

    cmp al, 0dh   ;elegxos an paththei to ENTER (vazw al=80h)
    jne contin    ;gia na mhn mperdeftw me to 'D' sth sinexeia
    mov al,80h
    jmp exit

contin:
    cmp al, 30h   ;oi klasikoi elegxoi
    jl ignore

    cmp al, 39h
    jg its_letter

    jmp exit

its_letter:
    cmp al, 'A'
    jl ignore
    cmp al, 'Z'
    jg ignore
exit:
    ret
read_key endp


end start
```

Και ένα στιγμιότυπο από το output της οθόνης για διαδοχικές εισοδους *A8X9S1FETD73A8KL* (παράδειγμα εκφώνησης) και *5GIU9O04WN7FG3D6*:



```
A8X9S1FETD73A8KL
891738-axsfetdakl
5GIU9004WN7FG3D6
5904736-giuownfgd
```