5. Using Signed Numbers and Look-up Tables

Macro Library for BIOS and DOS Services (save as exp5.inc)

```
; ASCII code for carriage return
CR equ 0Dh
; ASCII code for line feed
LF equ 0Ah

al2asc macro buffer
; al to ascii-decimal conversion
 xor ah,ah
 mov cx,100*256+10
 div ch
 mov buffer,al
 or buffer,30h
 mov al,ah
 xor ah,ah
 div cl
mov buffer+1,al
 or buffer+1,30h
 mov buffer+2,ah
 or buffer+2,30h
 mov buffer+3,'$'
al2asc endm

asc2al macro buf
;converts ascii str to number in al
 local hexnumber,numer1,numer2,negative,completed
 mov bl,byte ptr buf+1 ; size of the string
 mov bh,0
 mov al,[bx + offset buf+1]
 or al,20h ; lowercased
 cmp al,'h'
 je hexnumber
;number is decimal
 and al,0Fh
 mov cl,al
 dec bx
 je completed
 mov al,[bx + offset buf+1]
 cmp al,'-'
 je negative
 and al,0Fh
 mov ch,10
```

```
 mul ch
 add cl,al
 dec bx
 je completed
 mov al,[bx + offset buf+1]
 cmp al,'-'
 je negative
 and al,0Fh
 mov ch,100
 mul ch
 add cl,al
 dec bx
 je completed
 mov al,[bx + offset buf+1]
 cmp al,'-'
 je negative
 jmp completed
hexnumber:
 dec bx
 je completed
 mov al,[bx + offset buf+1]
 cmp al,'9'
 jna numer1
 add al,9 ; letter correction
numer1:
 and al,0Fh
 mov cl,al
 dec bx
 je completed
 mov al,[bx + offset buf+1]
 cmp al,'-'
 je negative
 cmp al,'9'
 jna numer2
 add al,9 ; letter correction
numer2:
 and al,0Fh
 mov ch,16
 mul ch
 add cl,al
 dec bx
 je completed
 mov al,[bx + offset buf+1]
 cmp al,'-'
 je negative
 jmp completed
```

```
negative:
 neg cl
completed:
 mov al,cl
asc2al endm

dispclr macro
 mov ax,0600h
 mov bh,0F0h
 mov cx,0000
 mov dx,184Fh
 int 10h
dispclr endm

dispstr macro string
 mov ah, 09h
 mov dx, offset string
 int 21h
dispstr endm

imultx macro prod,op1,op2
 mov ax,op1
 cwd
 mov cx,op2
 imul cx
 mov prod,ax
imultx endm

idivx macro quot,num,denom
; remainder returns in dx
 mov ax,num
 cwd
 mov cx,denom
 idiv cx
 mov quot,ax m
idivx endm

getstr macro buffer
 mov ah, 0Ah
 mov dx, offset buffer
 int 21h
getstr endm

keybch macro
 mov ah, 01h
 int 16h
```

```
keybch endm

setcurs macro row, col
 mov ah,02
 mov bh,00
 mov dl,col
 mov DH,row
 int 10H
setcurs endm

exitdos macro
 mov ah,4ch
 int 21h
exitdos endm
```

Last name counter (save as Exp5p1.asm)

```
 include exp5.inc
 .model small
 .stack 100h
 .data
rowno equ 08
colno equ 05
Message1 db 'What is your last name? ','$'
Buffer1 db 24,?,24 DUP (0)
Message2 db CR, LF,'Letter-count of your last nam count of your last name is: '
Message3 db ' $'

 .code
 mov ax,@data
 mov ds,ax
 dispclr

 setcurs rowno,colno
 dispstr Message1
 getstr buffer1
 ; Mem[buffer1+1] contains the stringlength
 mov al,Buffer1+1
 al2asc Message3
 dispstr Message2
waitkey:
 keybch
 jz waitkey
 exitdos
 end
```

Average by Signed Arithmetic Operations (save as Exp5p2.asm)

```
include exp5.inc
 .model small
 .stack 100h
 .data
snum dw 4
sdata db -3, -12, 5, 2 12, 5, 2 12, 5, 2
aver dw ?
remn dw ?
MessageA db "Average is $"
MessageR db "Remainder is $"
NextLine db 13,10,"$"
dstr db 10 dup(20h),'$'
 .code
 mov ax,@data
 mov ds,ax
 mov cx, snum
 mov bx, offset sdata
 mov dx,0
addloop:
 mov ax,[bx]
 cbw
 add dx,ax
 inc bx
 loop addloop
 mov ax,dx
 cwd
 mov cx,snum
 idiv cx
 mov aver,ax
 mov remn,dx
 mov ax,aver
 cmp ax,0
 jge positive
 mov dstr,'-'
 neg ax
positive:
 al2asc dstr+1
 dispstr NextLine
 dispstr MessageA
 dispstr dstr
 mov ax,remn
 al2asc dstr
 dispstr NextLine
 dispstr MessageR
 dispstr dstr
```

```
waitch:
 keybch
 jz waitch
 exitdos
 end
```

Look-Up Table for the Square Root of an Integer. (save as Exp5p3.asm)

```
include exp5.inc
 .model small .
 .data .data
Msg1 db 'I"ll find the square root using '
 db 'a look-up table.',13,10
 db ' Give me a number in the range [0, 255]: $'
Msg2 db 13,10,' Square-root is $'
lutcnt dw 15
lutin db 0, 1, 4, 9, 16, 25, 36, 49, 64
          db 81, 100, 121, 144,169,196, 225
lutout db 0, 1, 2, 3, 4, 5, 6, 7, 8,
          db 9, 10, 11, 12, 13, 14, 15
buf  db 10h,?,10h dup(' ');
output db 5 dup(' '), '$'
 .code
 mov ax, @data
 mov ds,ax
 dispstr Msg1
 getstr buf
 asc2al buf
 ; find index
 mov cx,lutcnt
lutlp:
 mov bx,cx
 cmp al,[bx + offset lutin]
 jae lutexit
 loop lutlp
lutexit:
 ; read output
 mov al,[bx + offset lutout]
 al2asc output
 dispstr Msg2
 dispstr output
waitch:
 keybch
 jz waitch
 exitdos
 end
```

Simple Look-Up Table for Fibonacci Numbers. (save as Exp5p4.asm)

```
include exp5.inc
 .model small
 .data
 luacnt dw 12
lua db 1,1,2,3,5,8,13,21,34,55,89,144,233
fibnr db ' $'
buf db 20,?, 20 dup(' ')
msga db 'I have a look-up table to get'
db ' the n-th Fibbonachi number.$'
msgb db cr,lf,'Give me a number in the range [0,12] : $'
msgc db cr,lf,'Your Fibonachi number is : $'
 .code
 mov ax,@data
 mov ds,ax
 dispstr msga
again:
 dispstr msgb
 getstr buf
 mov al,byte ptr buf+1
 cmp al,0
 jz emptystr
 asc2al buf
 xor ah,ah ; zero extend to ax
 mov bx,ax
 mov al, [bx + offset lua]
 al2asc fibnr
 dispstr msgc
 dispstr fibnr
 jmp again
emptystr:
 exitdos
 end
```