**Lecture 5 - Exercises**

**Ex\_5-1** - Determine if a number is negative, positive, or zero.

**Write a program that determines whether an integer is a negative number, a positive number,**

**or zero, and display the output shown below.**

* First, prompt the user to enter an integer.
* Compare the number to zero.
* If the number is less than zero, jump to a label and say it’s a negative number.
* If the number is equal to zero, jump to a label and say it’s equal to zero.
* Otherwise, say it’s a positive number.

Comment! ---- **OUTPUT** ------------------------------------

Enter a number: -3

The number is negative. ; If 0, then 🡪 The number is zero.

; If >= 0, then 🡪 The number is positive.

Press any key to continue . . . !

**TITLE Ex\_5-1** ---- **Determine if a number is neg., pos., or zero** -------

INCLUDE Irvine32.inc

List\_SIZE = 32

.data

getNumber BYTE "Enter a number: ",0

sayNegative BYTE "The number is negative.",0dh,0ah,0

sayPositive BYTE "The number is positive.",0dh,0ah,0

sayEven BYTE "The number is zero.",0dh,0ah,0

.code

main PROC

mov eax, 0

mov edx, OFFSET getNumber

call WriteString

call ReadInt

cmp eax, 0

jl NegNum

cmp eax, 0

je ZeroNum

mov edx, OFFSET sayPositive

call WriteString

jmp Done

NegNum:

mov edx, OFFSET sayNegative

call WriteString

jmp Done

ZeroNum:

mov edx, OFFSET sayEven

call WriteString

jmp Done

Done:

call Crlf

exit

main ENDP

END main

**Ex\_5-2** - Write a while loop using the .WHILE and .ENDW directives

**Write a program that outputs “Do your homework” five times.**

* Use the .WHILE and .ENDW directives.

Comment! ---- OUTPUT ----

Do your homework!

Do your homework!

Do your homework!

Do your homework!

Do your homework!

Press any key to continue . . .

**TITLE Ex\_5-2** ---- **Determine if a number is neg., pos., or zero** -------

INCLUDE Irvine32.inc

.data

count DWORD 0

msg BYTE "Hello",0

.code

main PROC

.WHILE count < 5

mov edx, OFFSET msg

call WriteString

call Crlf

inc count

.ENDW

main ENDP

exit

END main

comment! ---- OUTPUT ----

-1693693069

-230634897

-313304682

-57215027

-814224860

+1446550572

-1051256079

-2131210227

+1132678262

-98012153

-714025234

+1150390630

-149067653

-555958330

+699356226

+1209677756

-340501879

+42826349

-1121734177

-1938573773

The number of negative numbers is: 14

Press any key to continue . . !

**Ex\_5-4**

1.) Write a program that does the following:

* Fill a 32-bit array with 20 random integers.
* With each iteration, create a random number,

display it on the screen, and put it in an array.

(2) Use a second loop to access the array elements

and count the number of negative values.

* Include a variable to keep count (initialize it to 0).
* Use the following code to compare a number to zero.
* If the number is greater or equal to zero, jump

around the code to increment the count.

* If it’s not, then increment the count.

The ESI holds the OFFSET of an array of random numbers.

L2:

cmp [esi],0 ; compare value in the array to zero

jge L3 ; negative value?

inc count ; yes: add to count

L3:

add esi,4

loop L2

(3) After the loop finishes, display the count.

**TITLE Ex\_5-4** ---------------------------------------------------

INCLUDE Irvine32.inc

.data

intArray SDWORD 20 DUP(?)

count DWORD 0

displayNums BYTE "Here are 20 random numbers:",0dh,0ah,0dh,0ah,0

negNumCount BYTE "The number of negative numbers is: ",0

.code

main PROC

**call Randomize**

; Fill the array with random values

mov esi,OFFSET intArray ; point to the array

mov ecx,LENGTHOF intArray ; loop counter

mov edx, OFFSET displayNums

call WriteString

L1: **call Random32**  ; EAX = random value

call WriteInt

call Crlf

mov [esi],eax

add esi,TYPE intArray

loop L1

; **Search for negative values**

mov esi,OFFSET intArray ; point to the array

mov ecx,LENGTHOF intArray ; loop counter

**L2:**

mov eax, [esi]

cmp eax ,0 ; compare value to zero

; negative value?

**jge L3** ; **jump** if greater than or equal to 0

inc count ; if negative, add to count

**L3**:

add esi,4

**loop L2**

call Crlf

mov edx, OFFSET negNumCount

call WriteString

mov eax,count

call WriteDec

call Crlf

call Crlf

exit

main ENDP

END main

; ==================================================================