**COAL LAB 06 (LAB TASKS)**

**TASK # 01:**

**CODE:**

TITLE Q1 (test.asm)

INCLUDE Irvine32.inc

.data

var1 byte ?

var2 byte ?

align 2

var3 word ?

var4 word ?

align 4

var5 dword ?

var6 dword ?

align 2

.code

main PROC

mov esi, offset var1

mov esi, offset var2

mov esi, offset var3

mov esi, offset var4

mov esi, offset var5

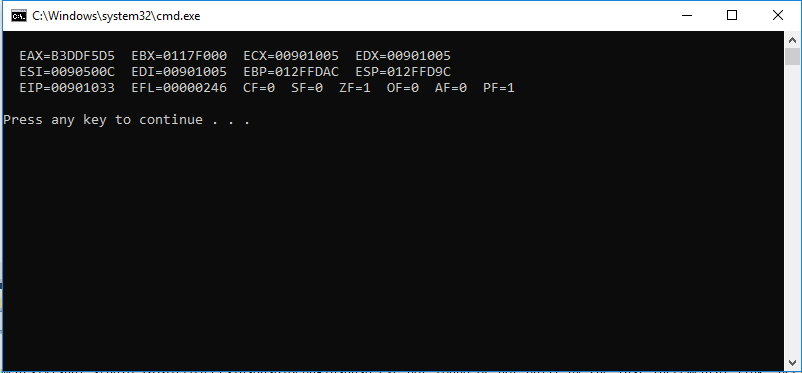
mov esi, offset var6

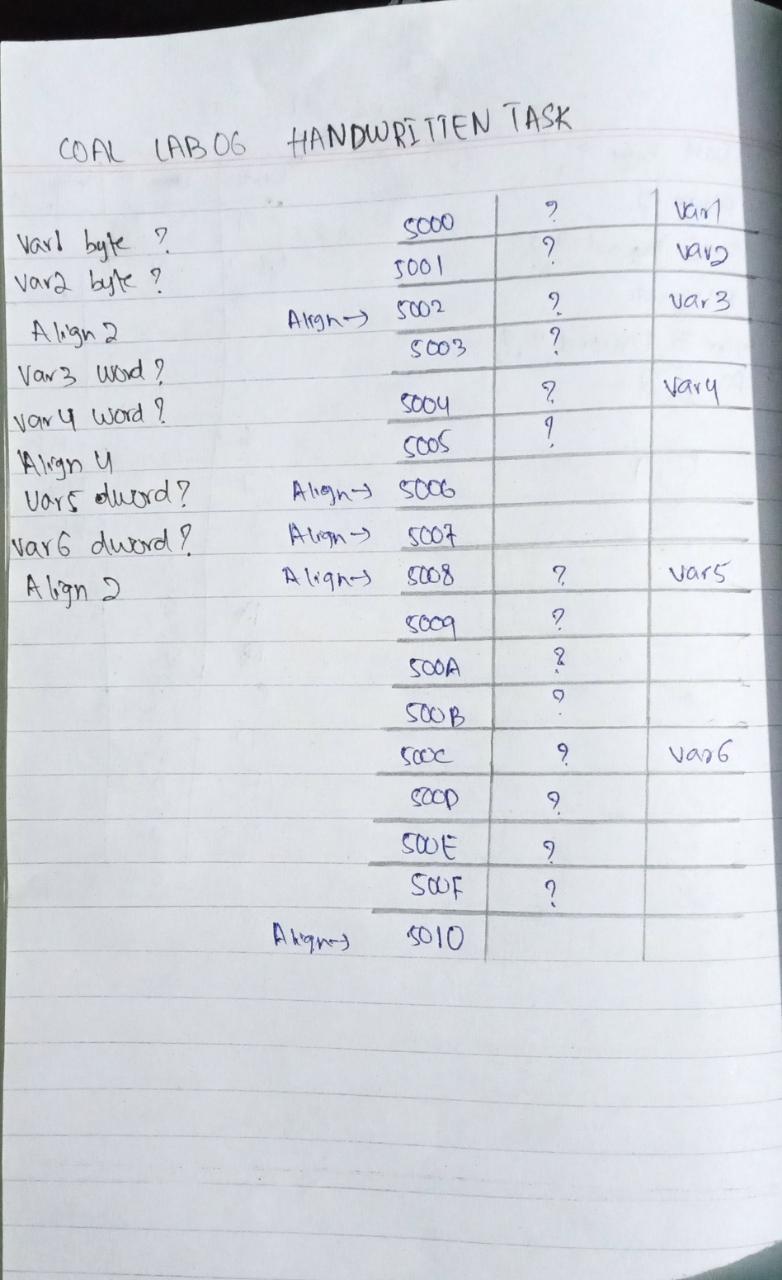
call DumpRegs

exit

main ENDP

END main

**OUTPUT:**

**HANDWRITTEN:**

**TASK # 02:**

**CODE:**

**Original Code:**

.data

intArray WORD 0100h, 0200h, 0300h, 0400h, 0500h, 0600h

.code

main proc

mov esi,offset intArray

mov ax,[esi]

top:

inc esi

inc esi

mov ax,[esi]

loop top

**Error:**

Loop does not have a terminating condition, thus in order to run loop a limited number of times, we have to assign a counting value in register ecx, and to display 600 on screen, we will basically run it for length-1 times because 1st execution is natural execution rather than looped execution.

**After Debugging Code:**

TITLE Q2 (test.asm)

INCLUDE Irvine32.inc

.data

intArray WORD 0100h, 0200h, 0300h, 0400h, 0500h, 0600h

.code

main proc

mov esi,offset intArray

mov ax,[esi]

mov ecx, lengthof intArray-1

top:

inc esi

inc esi

mov ax,[esi]

loop top

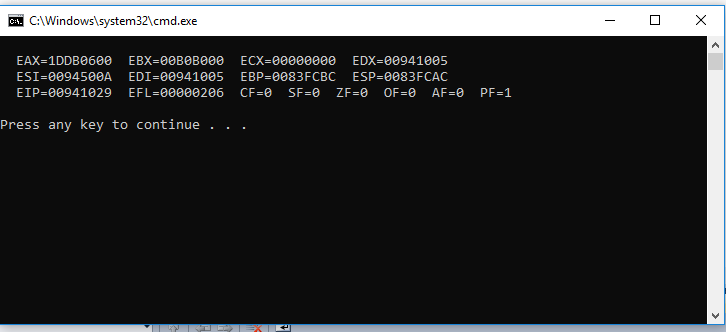
call DumpRegs

exit

main ENDP

END main

**OUTPUT:**



**TASK # 03:**

**CODE:**

TITLE Q3 (test.asm)

INCLUDE Irvine32.inc

.data

arrayB BYTE 20h,40h,60h,80h

arrayW WORD 200h, 300h, 400h

arrayD DWORD 500h, 900h, 1100h

.code

main proc

mov esi, 3

mov cl, arrayB

add cl, arrayB[esi\* TYPE arrayB] ; storing answer for byte in cl

mov esi, 2

mov bx, arrayW

add bx, arrayW[esi\* TYPE arrayW] ; storing answer for byte in bx

mov esi, 2

mov eax, arrayD

add eax, arrayD[esi \* TYPE arrayD] ; storing answer for byte in eax

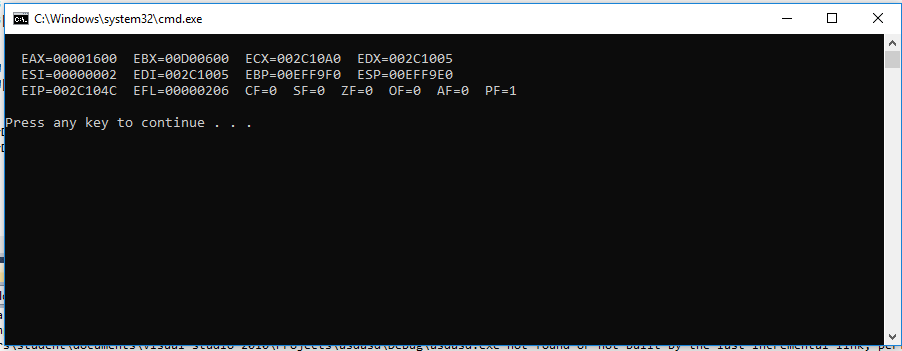
call DumpRegs

exit

main ENDP

END main

**OUTPUT:**



**TASK # 04:**

**CODE:**

TITLE Q4 (test.asm)

INCLUDE Irvine32.inc

.data

arr Word 100h, 200h, 400h, 600h

ptr1 dword offset arr

ptr2 dword offset [arr+2]

ptr3 dword offset [arr+4]

ptr4 dword offset [arr+6]

.code

main PROC

mov esi, ptr1

mov ax, [esi]

mov esi, ptr2

mov ax, [esi]

mov esi, ptr3

mov ax, [esi]

mov esi, ptr4

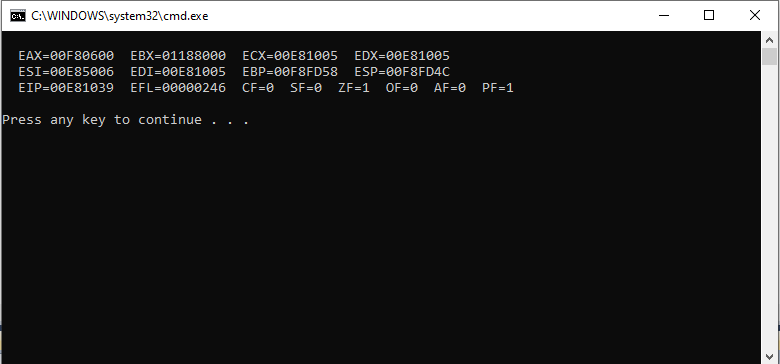
mov ax, [esi]

call DumpRegs

exit

main ENDP

END main

**OUTPUT:**

**TASK # 05:**

**CODE:**

TITLE Q5 (test.asm)

INCLUDE Irvine32.inc

.data

count word 0

.code

main proc

mov ecx, 100

L1: inc count

mov ax, count ; The answer will be stored in ax (A separate register)

Loop L1

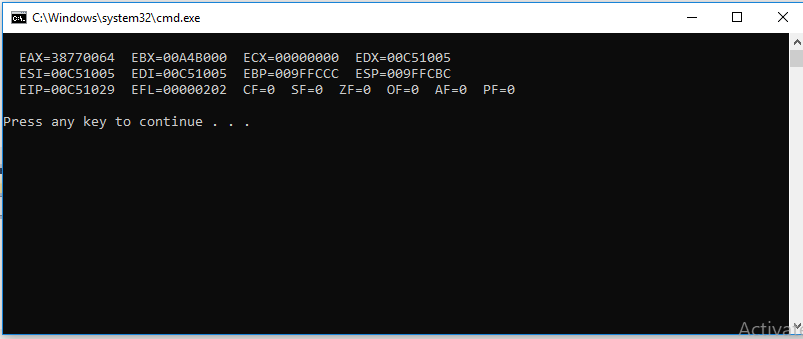
call DumpRegs

exit

main ENDP

END main

**OUTPUT:**



**TASK # 06:**

**CODE:**

TITLE Q5 (test.asm)

INCLUDE Irvine32.inc

.data

Num word 10

T1 word 0

T2 word 1

nextTerm word ?

.code

main proc

mov ax, T1 ; at n=1 it will just display no need to check for any sum

mov ax, T2 ; at n=2 it will just display no need to check for any sum

mov ecx, 8

L1: mov bx, T1

add bx, T2

mov nextTerm, bx

mov dx, T2

mov T1, dx

mov dx, nextTerm

mov T2, dx

mov eax, 0

mov ax, nextTerm

loop L1

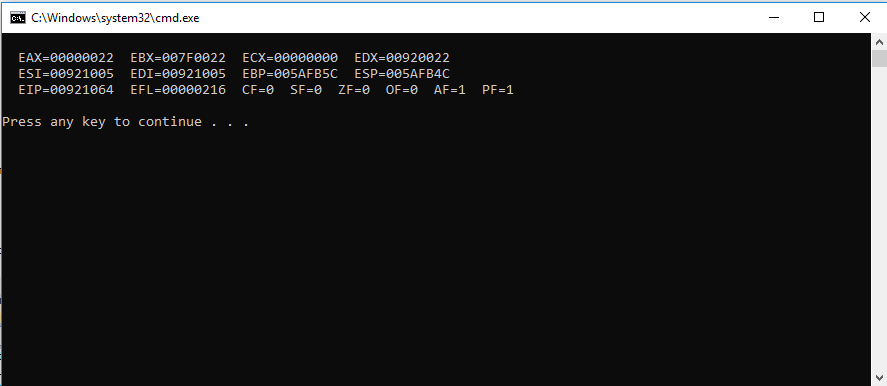
call DumpRegs

exit

main ENDP

END main

**OUTPUT:**



**TASK # 07:**

**CODE:**

TITLE Q7 (test.asm)

INCLUDE Irvine32.inc

.data

Alphabet byte 'A'

.code

main PROC

mov ecx, 26

L1: mov al, Alphabet

inc Alphabet

Loop L1

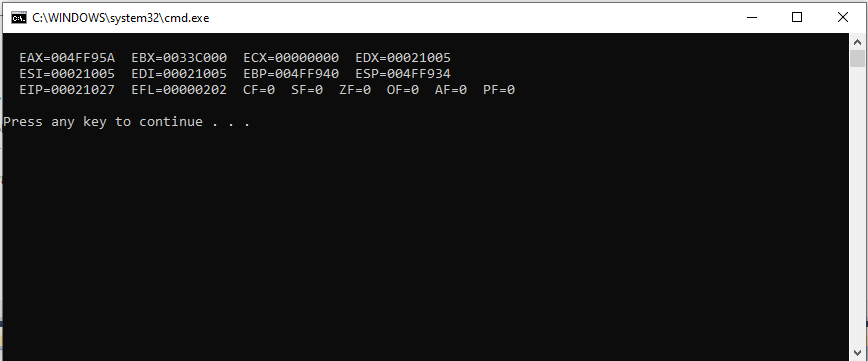
call DumpRegs

exit

main ENDP

END main

**OUTPUT:**



**TASK # 08:**

**CODE:**

TITLE Q8 (test.asm)

INCLUDE Irvine32.inc

.data

count byte 1

arrayCount byte 15 dup(?)

.code

main PROC

mov ecx, 15

mov esi, 0

L1: mov al, count ; displaying decimal value

mov arrayCount[esi], al ; storing it in a different array

inc count

inc esi

loop L1

mov ecx, 15

mov esi, 0

L2: mov bl, arrayCount[esi] ; displaying the array in which value is stored

inc esi

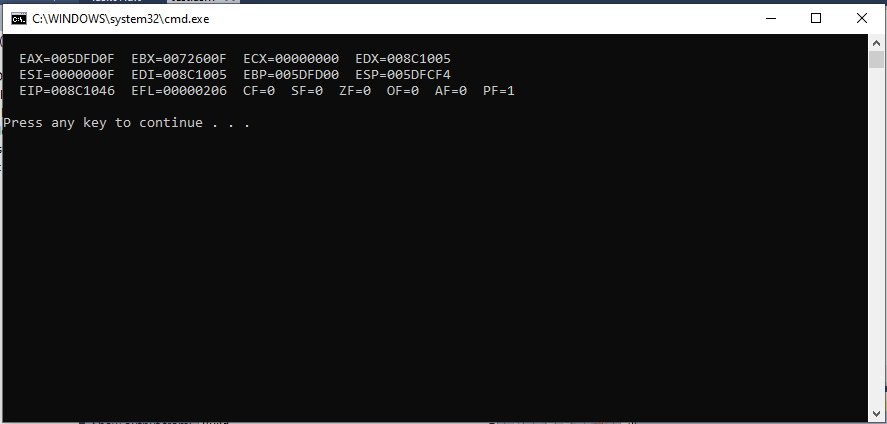
loop L2

call DumpRegs

exit

main ENDP

END main

**OUTPUT:**