**LAB 6**

Questions:

Task: 1 Write a program that uses a loop to calculate the first ten numbers of Fibonacci sequence.

Task: 3 write a program to take input data for 5 employees and store it in appropriate variables. The

program should ask for Employee ID, Name, Year of Birth & Annual Salary from the user. All variables

should be stored in an array whose index represent employee number. The program should then calculate

the annual salary for all employees by adding all the elements in AnnualSalary array.

Task: 4 Initialize an array named Source and use a loop with indexed addressing to copy a string

represented as an array of bytes with a null terminator value in an array named as target.

Task: 5 Use a loop with direct or indirect addressing to reverse the elements of an integer array in place.

Do not copy elements to any other array. Use SIZEOF, TYPE and LENGTHOF operators to make program

flexible.

Task: 6 initialize a double word array consisting of elements 8, 5,1,2,6. Sort the given array in ascending

order using bubble sort.

Q1

Code:

INCLUDE Irvine32.inc

.data

prev DWORD 0 ;

current DWORD 1 ;

.code

main PROC

mov eax , 0

mov ecx , 10 ; to run loop 10 times

L1:

mov eax , prev

call Writeint

mov eax , prev

add eax , current ; prev + current

mov ebx , current ;

mov prev , ebx ; update prev to current

mov current , eax ; update current to eax

call crlf

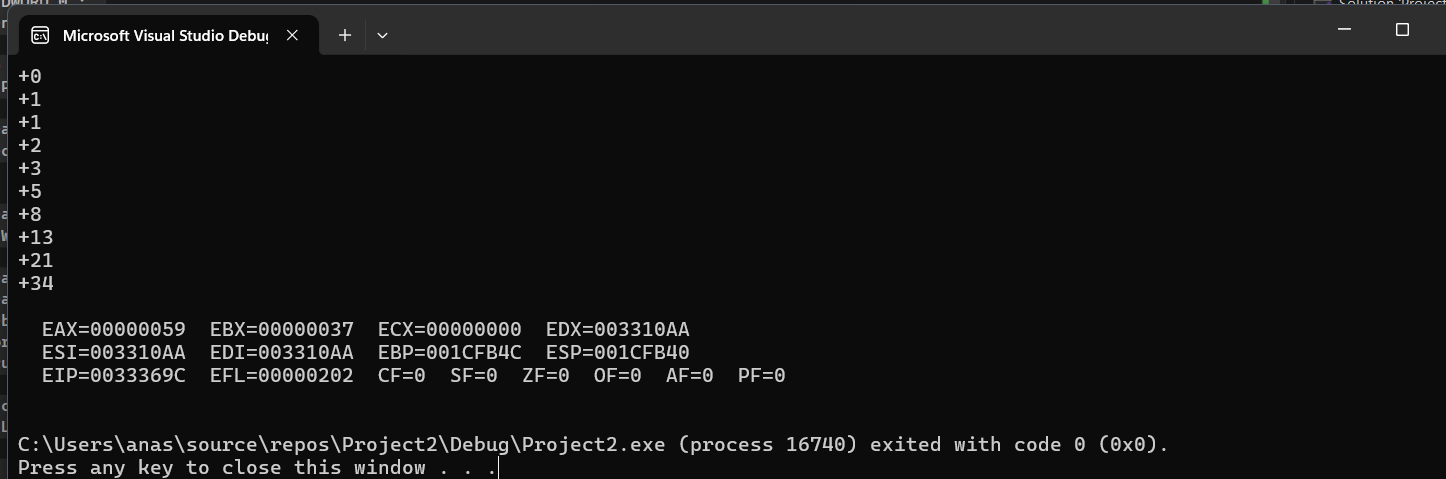
loop L1

call DumpRegs

exit

main ENDP

END main



Q2

Pattern :

1

11

111

1111

Code:

INCLUDE Irvine32.inc

.data

count DWORD ? ; to save the outer loop count

var DWORD 1

.code

main PROC

mov eax , 0

mov ecx , 4 ; save the outer loop count (4 rows)

mov ebx , ecx ; save the outer loop count

L1:

mov count , ecx ; save the current outer loop count

mov ecx , ebx ; restore the outer loop count

sub ecx , count ; set inner loop count

inc ecx

L2:

mov eax , var ; load the character

call Writeint

loop L2 ; repeat the inner loop

call crlf

mov ecx , count

loop L1

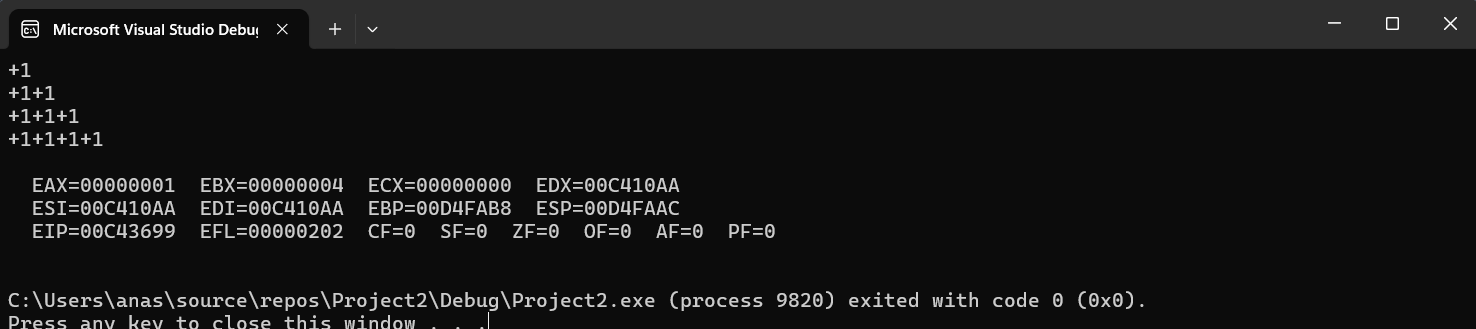
call DumpRegs

exit

exit

main ENDP

END main



Pattern :

1111

111

11

1

Code:

INCLUDE Irvine32.inc

.data

count DWORD ? ; to save the outer loop count

var DWORD 1

.code

main PROC

mov eax , 0

mov ecx , 4 ; save the outer loop count (4 rows)

mov ebx , ecx ; save the outer loop count

L1:

mov count , ecx ; save the current outer loop count

mov ecx , count ; restore the inner loop count

L2:

mov eax , var ; load the character

call Writeint

loop L2 ; repeat the inner loop

call crlf

mov ecx , count

loop L1 ; by calling this it is automatically decrementing ecx by 1

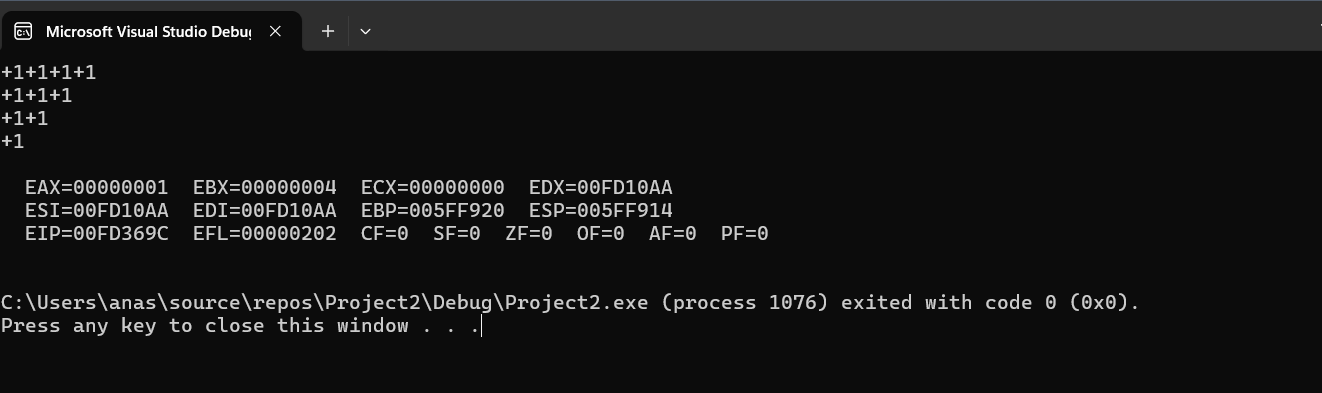
call DumpRegs

exit

exit

main ENDP

END main



Pattern:

4321

432

43

4

Code:

INCLUDE Irvine32.inc

.data

count DWORD ? ; to save the outer loop count

var1 DWORD 4 , 3 , 2, 1

.code

main PROC

mov eax , 0

mov esi , 0

mov ecx , 4 ; save the outer loop count (4 rows)

mov ebx , ecx ; save the outer loop count

L1:

mov esi , 0 ; reset index to 0 for each row

mov count , ecx ; save the current outer loop count

mov ecx , count ; set ecx to print number of items

L2:

mov eax , var1[esi \*TYPE var1] ; load the character

call Writeint

inc esi

loop L2 ; repeat the inner loop

call crlf

mov ecx , count

loop L1

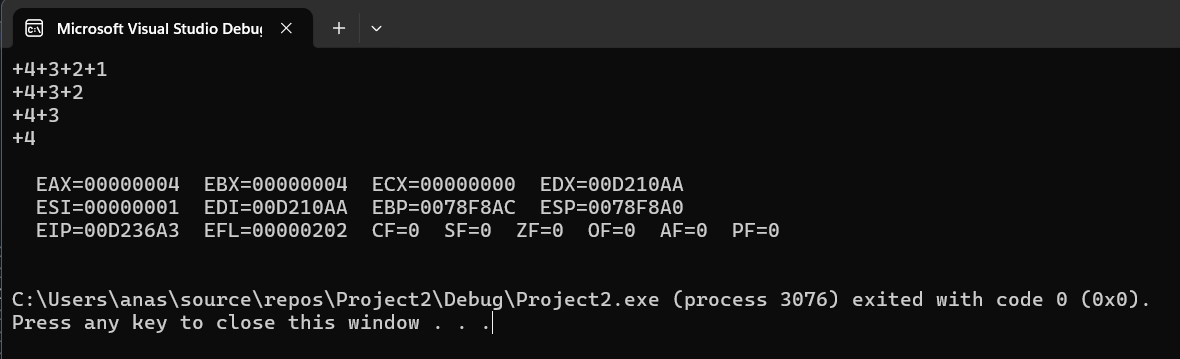
call DumpRegs

exit

exit

main ENDP

END main



Pattern:

1234

123

12

1

Code:  
INCLUDE Irvine32.inc

.data

count DWORD ? ; to save the outer loop count

var1 DWORD 1 , 2 , 3, 4

.code

main PROC

mov eax , 0

mov esi , 0

mov ecx , 4 ; save the outer loop count (4 rows)

mov ebx , ecx ; save the outer loop count

L1:

mov esi , 0 ; reset index to 0 for each row

mov count , ecx ; save the current outer loop count

mov ecx , count ; set ecx to print number of items

L2:

mov eax , var1[esi \*TYPE var1] ; load the character

call Writeint

inc esi

loop L2 ; repeat the inner loop

call crlf

mov ecx , count

loop L1

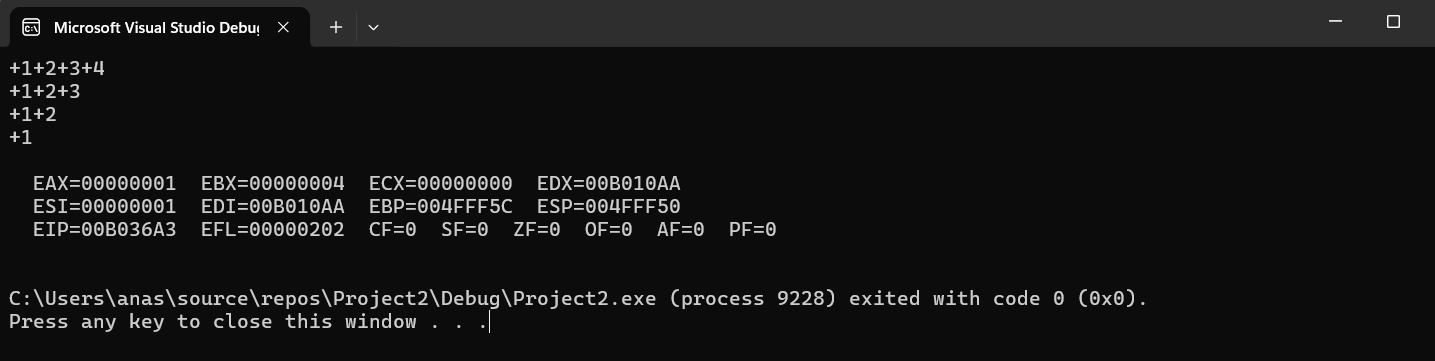
call DumpRegs

exit

exit

main ENDP

END main



Q3

Code:

INCLUDE Irvine32.inc

.data

nameArray BYTE 5 DUP(20 DUP(?))

ID DWORD 5 DUP(?)

Birth DWORD 5 DUP(?)

AnnualSalary DWORD 5 DUP(?)

promptName BYTE "Enter name:",0

promptID BYTE "Enter ID: " , 0

promptBirth BYTE "Enter the year of Birth: " , 0

promptAS BYTE "Enter annual salary: " , 0

newline BYTE 0Dh, 0Ah,0

.code

main PROC

mov ebx,5 ; set up a counter to read info

mov ecx,5 ; set up a loop counter

mov esi, offset nameArray

;entering data in array

label1:

mov edx, offset promptName ; Load the address of the prompt message

call Writestring

mov edx,esi

mov ecx,20

call Readstring

add esi,20

dec ebx

mov ecx,ebx

loop label1

;Entering IDs

mov ebx,5

mov ecx,5

mov esi, offset ID

label2:

mov edx, offset promptID ; Load the address of the prompt message

call Writestring

call Readint

mov [esi] , eax

add esi , 4 ; since ID is of DWORD

dec ebx

mov ecx , ebx

loop label2

;Entering birth years

mov ebx , 5

mov ecx , 5

mov esi , offset Birth

label3:

mov edx , offset promptBirth

call Writestring

call Readint

mov [esi] , eax ; integer tou eax mai hi ata hai

add esi , 4 ;

dec ebx

mov ecx , ebx

loop label3

;Entering annual salaries

mov ebx , 5

mov ecx , 5

mov esi , offset AnnualSalary

label4:

mov edx , offset promptAS

call Writestring

call Readint

mov [esi] , eax

add esi , 4

dec ebx

mov ecx , ebx

loop label4

mov ecx , 5

mov esi , 0

mov eax , 0

;to calculate total annual salaries

label5:

add eax , AnnualSalary[esi \* TYPE AnnualSalary]

inc esi

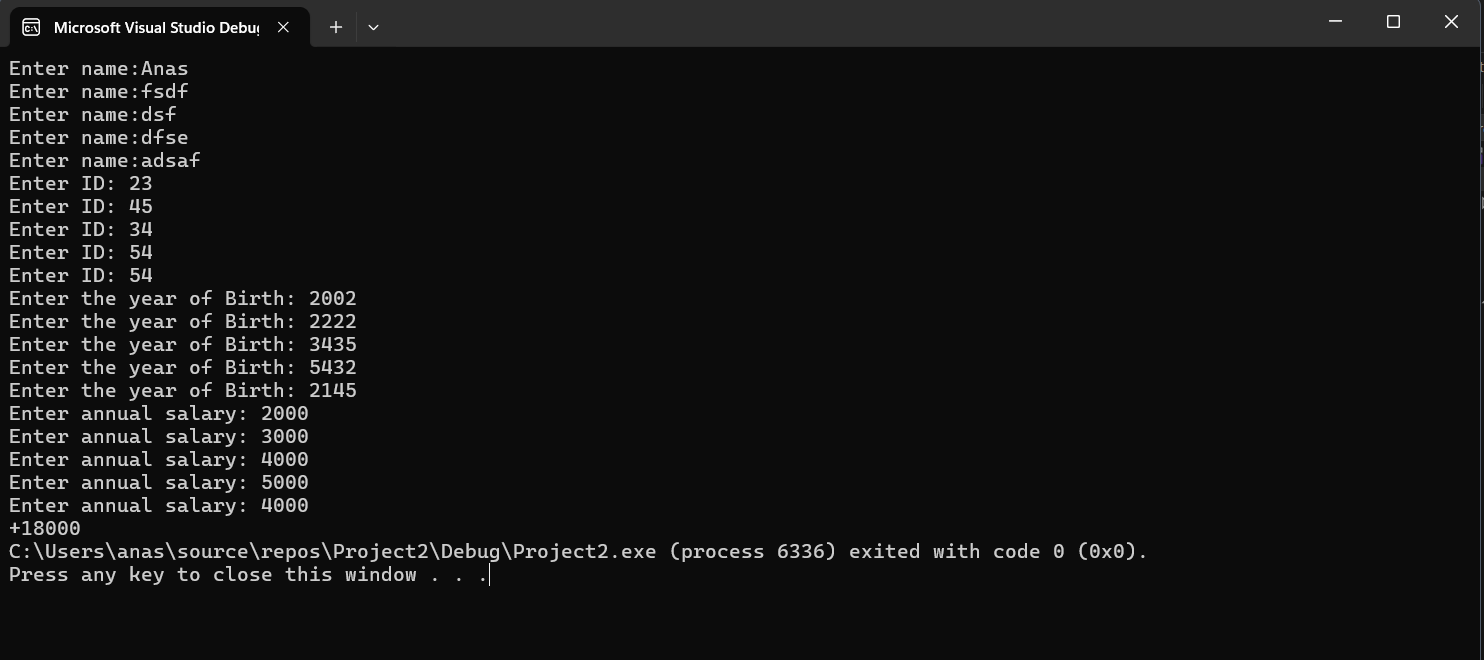
loop label5

call Writeint ; printing salaries

exit

main ENDP

END main



Q4

Code:

INCLUDE Irvine32.inc

.data

Source BYTE "Hello Assembly" , 0 ; 0 here represents null terminator

target BYTE 20 DUP(?)

.code

main PROC

mov edx , 0

mov esi , 0

mov edi , 0

mov ecx , 20

label1:

mov al , Source[esi \* TYPE Source]

mov target[edi] , al

cmp al , 0 ; check if al has encountered null terminator

je done ; if yes then end the loop (means move to done label and end this label)

inc esi

inc edi

jmp label1 ; jump to label 1(repeat)

done:

mov edx, offset target ; Load target address

call Writestring ; Print the copied string

exit

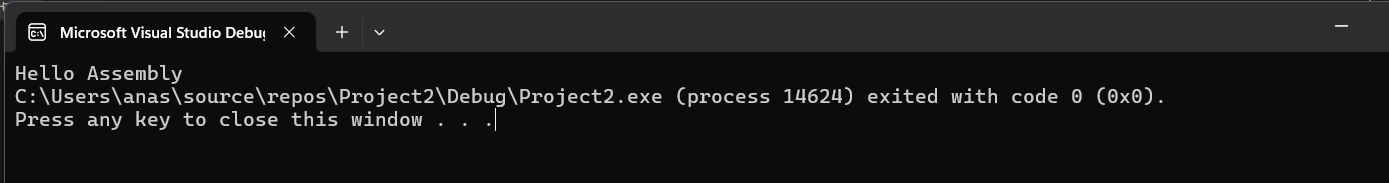
mov edx , offset target

call Writestring

exit

main ENDP

END main



Q5

Code:

INCLUDE Irvine32.inc

.data

arrayint DWORD 1 , 2 , 3 , 4 , 5

.code

main PROC

mov eax , 0

mov ebx , 0

mov esi , 0

mov edi , LENGTHOF arrayint

sub edi , 1

mov ecx , LENGTHOF arrayint

shr ecx , 1 ; divide ecx by 2

label1:

;swapping elements

mov eax , arrayint[esi \* TYPE arrayint]

mov ebx , arrayint[edi \* TYPE arrayint]

mov arrayint[esi \* TYPE arrayint] , ebx

mov arrayint[edi \* TYPE arrayint] , eax

dec edi

inc esi

loop label1

mov eax , 0

mov esi , 0

mov ecx , LENGTHOF arrayint

label2:

mov eax , arrayint[esi \* TYPE arrayint]

call Writeint

inc esi

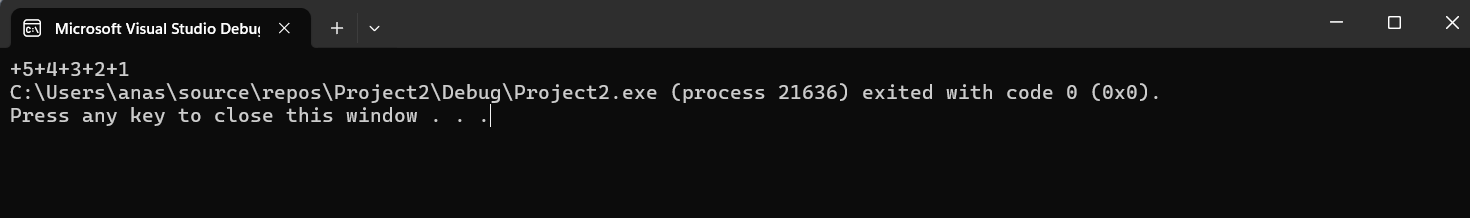
loop label2

exit

exit

main ENDP

END main



Q6

Code:

INCLUDE Irvine32.inc

.data

array DWORD 8, 5, 1, 2, 6

.code

main PROC

;clearing registers

mov eax , 0

mov ebx , 0

mov ecx , 0

mov edx , 0

mov esi , 0

mov edi , LENGTHOF array

;(outer loop)

label2:

mov esi , 0

;(inner loop)

label1:

mov eax , array[esi \* TYPE array]

inc esi

mov ebx , array[esi \* TYPE array] ; move to the adjacent element

dec esi

cmp eax , ebx

JLE no\_swap ; if current <=swap then no need to swap

;if not , then need to perform swapping

mov array[esi \* TYPE array] , ebx

inc esi

mov array[esi \* TYPE array] , eax

dec esi

no\_swap:

inc esi

cmp esi , edi

JL label1

dec edi

cmp edi , 1 ; compare if edi has reached the first element

JG label2 ; if greater then jump to label2 (outer loop) else move to printing the sorted array

;Printing the sorted array

mov esi, 0

print\_loop:

mov eax, array[esi \* TYPE array]

call Writeint

call crlf

inc esi

cmp esi, LENGTHOF array ; Check if all elements are printed

jl print\_loop ; Continue printing if not

exit

main ENDP

END main

