**QUESTION 1:**

INCLUDE Irvine32.inc

.data

arr BYTE "hello world",0

arr2 BYTE "result",0

.code

main PROC

mov edi, OFFSET arr2

mov esi, OFFSET arr ; start of string

NextWord:

mov al, [esi]

;----------- end loop -----------

cmp al, 0

je Done

;--------- skip spaces -------

cmp al, ' '

je PrintSpace

;--------- find start of word

mov ebx, esi

FindEnd:

mov al, [esi]

cmp al, ' '

je Reverse

cmp al, 0

je Reverse

inc esi

jmp FindEnd

Reverse:

mov edi, esi ; EDI = end of word (space or null)

dec edi ; step back to last character

PrintBackward:

mov al, [edi]

cmp edi, ebx ; reached start?

je AfterWord

dec edi

jmp PrintBackward

AfterWord:

cmp byte ptr [esi], 0

je Done ; if end of string, exit

inc esi ; skip the space

jmp NextWord

PrintSpace:

inc esi

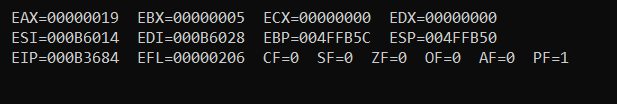
jmp NextWord

Done:

exit

main ENDP

END main



**QUESTION 2:**

INCLUDE Irvine32.inc

.data

arr DWORD 2,2,3,4,5,4,3,2,2,0 ; input array

count1 DWORD 0

count2 DWORD 0

count3 DWORD 0

count4 DWORD 0

count5 DWORD 0

count6 DWORD 0

count7 DWORD 0

count8 DWORD 0

.code

main PROC

mov esi, OFFSET arr ; esi stroe of first element of arr

l1:

mov eax,[esi]

cmp eax,0

je Done

cmp eax,1

je Counter1

cmp eax,2

je Counter2

cmp eax,3

je Counter3

cmp eax,4

je Counter4

cmp eax,5

je Counter5

add esi,4

jmp L1

Counter1:

inc count1

jmp L1

Counter2:

inc count2

jmp L1

Counter3:

inc count3

jmp L1

Counter4:

inc count4

jmp L1

Counter5:

inc count5

jmp L1

Counter6:

inc count6

jmp L1

Counter7:

inc count7

jmp L1

Counter8:

inc count8

jmp L1

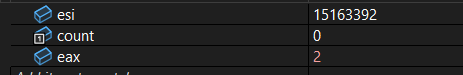
Done:

call DumpRegs

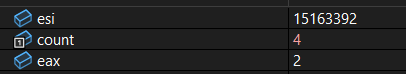
exit

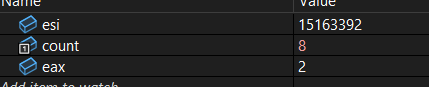
main ENDP

END main

****

Counter: 3,then 4





**QUESTION 3:**

INCLUDE Irvine32.inc

.data

arr DB "hello word",0 ; input array

count BYTE 0

.code

main PROC

mov esi, OFFSET arr ; esi stroe of first element of arr

L1:

mov al, [esi] ; DEREFRERNCE ESI

cmp al,0 ; the last one is 0== end of array

je Done

cmp al,'a'

je Counter

cmp al,'e'

je Counter

cmp al,'i'

je Counter

cmp al,'o'

je Counter

cmp al,'u'

je Counter

cmp al,'a'

je Counter

cmp al,'A'

je Counter

cmp al,'E'

je Counter

cmp al,'O'

je Counter

cmp al,'I'

je Counter

cmp al,'U'

je Counter

add esi,1

jmp L1

Counter:

inc count

jmp L1

Done:

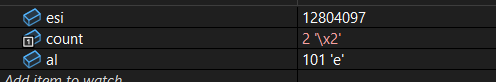
mov al,count

call DumpRegs

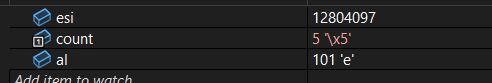
exit

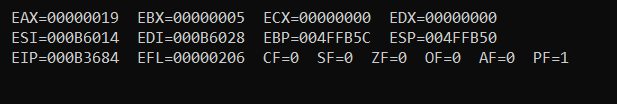
main ENDP

END main



2,3,4,then count=5





**QUESTION 4:**

INCLUDE Irvine32.inc

.data

arr DWORD 1,2,3,4,5 ; input array

res DWORD 5 DUP(?) ; FOR RESULT

.code

main PROC

mov ecx, 5 ; loop

mov esi, OFFSET arr ; esi stroe of first element of arr

mov edi, OFFSET res

L1:

mov eax, [esi] ; DEREFRERNCE ESI

mov ebx, eax

mul ebx ; EAX = EAX \* EBX

mov [edi], eax ; store result

add esi, 4 ; jump to next value

add edi, 4

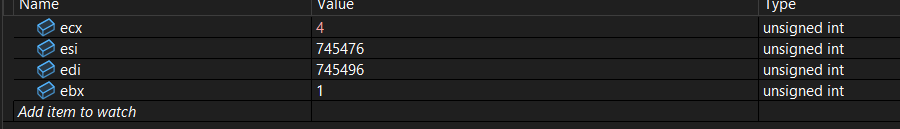
loop L1

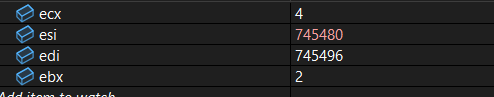
call DumpRegs ; show registers

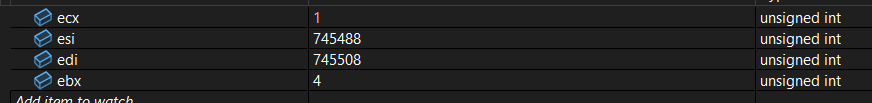
exit

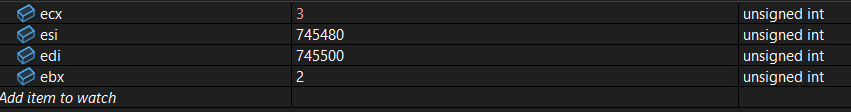
main ENDP

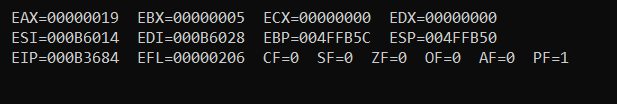
END main











**QUESTION 5:**

INCLUDE Irvine32.inc

.data

arr byte "hello",0

arr1 byte "PALINDROME",0

arr2 byte "NOT PALINDROME",0

.code

mov esi, OFFSET arr

mov edi, OFFSET arr

main PROC

;------------ FIND THE END OF ARRAY -----------

SearchEnd:

mov al,[edi]

cmp al,0

je foundLAST

add edi,1

jmp SearchEnd

foundLAST:

sub edi,1

;------------- COMPARE FROM START AND END ------------

Compare:

cmp esi,edi

jge Palindrome ; esi at start, esi++

; edi at end, edi--

; if they crossed ...win win

;else

mov al,[esi]

mov bl,[edi]

cmp al,bl

jne NotPalindrome

add esi,1

sub edi,1

jmp Compare

Palindrome:

mov edx, OFFSET arr1

jmp Done

NotPalindrome:

mov edx, OFFSET arr2

jmp Done

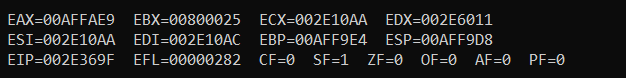
Done:

call DumpRegs

exit

main ENDP

END main



**QUESTION 6:**

INCLUDE Irvine32.inc

.data

arr DWORD 1,2,3,4

.code

main PROC

mov ecx,4

mov esi, OFFSET arr ; esi stroe of first element of arr

mov edi, OFFSET arr

add edi, 12 ; edi +12 = last place

mov eax, [edi] ; eax=4

Rotate:

cmp edi,esi ; untill edi == start

je StoreLast

;------------ get previous element and send 1 step ahead --------

mov ebx, [edi-4] ; preveios element , 3

mov [edi], ebx ; send 3 to last ... [1, 2, 3, 3]

sub edi,4

jmp Rotate

; ------ at last put 4 at the start ----------------

StoreLast:

mov [esi], eax

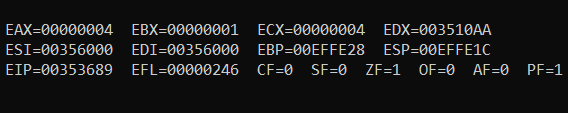
Done:

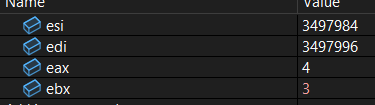
call DumpRegs

exit

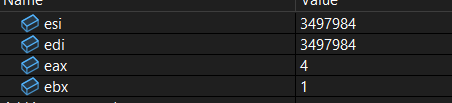
main ENDP

END main



All value changed first:  


At the end [4,1,2,3]



Eax stores 4

**QUESTION 7:**

INCLUDE Irvine32.inc

.data

arr DWORD 1, 13, 72, 42, 9, 41, 5, 24, 46, 8

count DWORD 10

.code

main PROC

mov ecx, count ; outer loop counter

dec ecx ; last element automaticl gets sorted

OuterLoop:

mov esi, OFFSET arr

mov ebx, ecx

InnerLoop:

mov eax, [esi] ; current element

mov edx, [esi + 4] ; nex element

cmp eax, edx ; compare current and next

jle NoSwap ; if already in order, skip swap

;------------------ swap arr[i] and arr[i+1] --------------------

mov [esi], edx

mov [esi + 4], eax

NoSwap:

add esi, 4 ; move to next

dec ebx

jnz InnerLoop

dec ecx

jnz OuterLoop ; repeat

mov esi, OFFSET arr

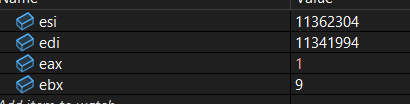
call DumpRegs

exit

main ENDP

END main

All value changed first:



eax = current

edx = next

