National University of Computer and Emerging Sciences



Lab 11

COAL

|  |  |
| --- | --- |
| Name | Muhammad Zain |
| Roll No. | 19F-0228 |
| LAB INSTRUCTOR | Mr. Abuzar Ghafari |
| Semester | Fall 2020 |

Task 1

# Source Code

*; Author: Muhammad Zain*

; Program Name: Myltiplication

INCLUDE Irvine32.inc

.data

m1 word 2 dup (3 dup (0))

m2 word 3 dup (2 dup (0))

count byte 0

index byte 0

dummy1 byte "Enter Matrix 1", 0;

dummy2 byte "Enter Matrix 2", 0;

dummy3 byte "Entered Matrix 1",0

dummy4 byte "Entered Matrix 2",0

dummy5 byte "After Multiplication",0

.code

main proc

mov esi, offset m1

mov eax,0

mov ecx, 6

mov edx, offset dummy1

call writestring

call crlf

l1:

call readint

mov word ptr[esi], ax

add esi, 2

loop l1

call crlf

mov esi, offset m2

mov eax,0

mov ecx, 6

mov edx, offset dummy2

call writestring

call crlf

l2:

call readint

mov word ptr[esi], ax

add esi, 2

loop l2

mov ecx, 6

mov eax, 0

mov esi, offset m1

mov edx, offset dummy3

call crlf

call writestring

call crlf

l3:

inc count

mov ax, [esi]

call writedec

mov al, ' '

call writechar

add esi, 2

cmp count, 3

je equal

jmp jjj

equal:

mov count, 0

call crlf

jjj:

loop l3

mov ecx, 6

mov eax, 0

mov esi, offset m2

mov edx, offset dummy4

call crlf

call writestring

call crlf

l4:

inc count

mov ax, [esi]

call writedec

mov al, ' '

call writechar

add esi, 2

cmp count, 3

je equal1

jmp jjj1

equal1:

mov count, 0

call crlf

jjj1:

loop l4

mov esi, offset m1

mov edi, offset m2

mov eax,0

mov ecx, 6

l5:

mov bx, [esi]

imul bx, [edi]

mov [esi], bx

add esi, 2

add edi, 2

loop l5

mov ecx, 6

mov eax, 0

mov esi, offset m1

mov edx, offset dummy5

call crlf

call writestring

call crlf

l6:

inc count

mov ax, [esi]

call writedec

mov al, ' '

call writechar

add esi, 2

cmp count, 3

je equal2

jmp jjj2

equal2:

mov count, 0

call crlf

jjj2:

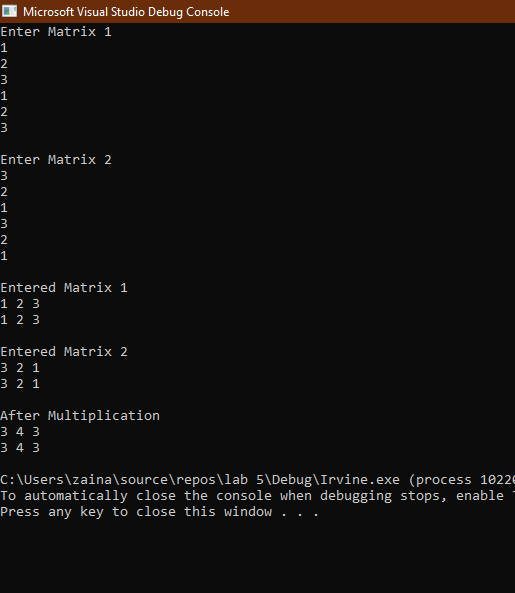
loop l6

exit

main endp

end main

# Snip



Task 2

# Source Code:

INCLUDE Irvine32.inc

.data

outputmsg1 byte "Before Elements in Array",0

outputmsg2 byte "After sorting Elements in Array",0

dummy DWORD 5 dup(?)

outerCounter WORD 0

innerCounter Word 0

searchVal DWORD 0; search value

first DWORD 0; first position

last DWORD 0 ; last position

mid DWORD 0 ; midpoint

.code

main PROC

call input

mov edx,offset outputmsg1

call output

call crlf

call BubbleSort

mov edx,offset outputmsg2

call output

exit

main ENDP

input proc

mov ecx,lengthof dummy

mov esi,0

l2:

mov eax,22 ;get random 0 to 4

call RandomRange ;

call Randomize

mov ebx,offset dummy; we can write it outside loop as well no effect here

mov [ebx+esi],eax

CALL writedec

call crlf

add esi, 4;we can use 2 here as well

loop l2

ret

input endp

output proc

call writestring

call crlf

mov ecx,lengthof dummy

mov esi,0

mov ebx,offset dummy

l1:

mov ax,[ebx+esi]

call writedec

call crlf

add esi, 4

loop l1

call crlf

ret

output endp

;..........................

;..........................

BubbleSort PROC USES eax ecx esi,

pArray:PTR DWORD, ; pointer to array

Count:DWORD ; array size

; Sort an array of 32-bit signed integers in ascending

; order, using the bubble sort algorithm.

; Receives: pointer to array, array size

; Returns: nothing

;-----------------------------------------------------------

mov ecx,5

dec ecx ; decrement count by 1

L1: push ecx ; save outer loop count

mov ebx,offset dummy ; point to first value

mov esi ,0

L2:

mov eax,[ebx+esi] ; get array value

mov esi,4

cmp [ebx+esi],eax ; compare a pair of values

sub esi,4

jg L3 ; if [ESI] <= [ESI+4], no exchange

mov esi,4

xchg eax,[ebx+esi] ; exchange the pair

mov [ebx+esi],eax

L3: add esi,4 ; move both pointers forward

loop L2 ; inner loop

pop ecx ; retrieve outer loop count

loop L1 ; else repeat outer loop

L4: ret

BubbleSort ENDP

BinarySearch PROC

; pointer to array

; array siz

; Searches an array of signed integers for a single value.

; Receives: Pointer to array, array size, search value.

; Returns: If a match is found, EAX = the array position of the

; matching element; otherwise, EAX = -1.

;-------------------------------------------------------------

call readdec

mov eax,searchVal

mov first,0 ; first = 0

mov eax,5 ; last = (count - 1)

dec eax

mov last,eax

mov edi,searchVal ; EDI = searchVal

mov ebx,dummy ; EBX points to the array

L1: ; while first <= last

mov eax,first

cmp eax,last

jg L5 ; exit search

; mid = (last + first) / 2

mov eax,last

add eax,first

shr eax,1

mov mid,eax

; EDX = values[mid]

mov esi,mid

shl esi,4 ; scale mid value by 4

mov edx,[ebx+esi] ; EDX = values[mid]

; if ( EDX < searchval(EDI) )

cmp edx,edi

jge L2

; first = mid + 1

mov eax,mid

inc eax

mov first,eax

jmp L4

; else if( EDX > searchVal(EDI) )

L2: cmp edx,edi ; optional

jle L3

; last = mid - 1

mov eax,mid

dec eax

mov last,eax

jmp L4

; else return mid

L3: mov eax,mid ; value found

jmp L9 ; return (mid)

L4: jmp L1 ; continue the loop

L5: mov eax,-1 ; search failed

L9: ret

BinarySearch ENDP

END main

# Snip:

Text

Description automatically generated