**COAL LAB 10**

**23k-0575**

Questions:

1. Write a program which contains a procedure named ThreeProd that displays the

product of three numeric parameters passed through a stack.

2. Write a program which contains a procedure named MinMaxArray that displays the

minimum & maximum values in an array. Pass a size-20 array by reference to this

procedure.

3. Write a program which contains a procedure named LocalSquare . The procedure must

declare a local variable. Initialize this variable by taking an input value from the user and

then display its square. Use ENTER & LEAVE instructions to allocate and de-allocate the

local variable.

4. Write a program to take 4 input numbers from the users. Then make two procedures

CheckPrime and LargestPrime. The program should first check if a given number is a

prime number or not. If all of the input numbers are prime numbers then the program

should call the procedure LargestPrime.

CheckPrime: This procedure tests if a number is prime or not

LargestPrime: This procedure finds and displays the largest of the four prime numbers.

5. Write a program which contains a procedure named BubbleSort that sorts an array

which is passed through a stack using indirect addressing.

Q1

Code:

INCLUDE Irvine32.inc

.data

var1 DWORD 5

var2 DWORD 6

var3 DWORD 7

.code

main PROC

push var1

push var2

push var3

call ThreeProd

call Writeint

exit

main ENDP

ThreeProd PROC

push ebp

mov ebp, esp

mov eax , [ebp+16]

mov ebx , [ebp+12]

mul ebx ; eax = eax(var1)\*ebx(var2)

mov edx , [ebp+8]

mul edx

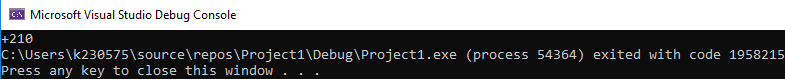
;call Writeint

pop ebp

ret 12

ThreeProd ENDP

END main



Q2

Code:  
INCLUDE Irvine32.inc

.data

arr DWORD 2,5,7,1,10,34,54,43,2,32,21,43,65,76,43,32,23,12,67,64

str1 BYTE "Maximum value of the array: " , 0

str2 BYTE "Minimum value of the array: " , 0

.code

main PROC

push offset arr

call MinMaxArray

exit

main ENDP

MinMaxArray PROC

push ebp

mov ebp, esp

pushad

;to find the max value of the array

mov ecx , LENGTHOF arr

mov esi , [ebp+8]

mov eax , [esi] ; load the first element of the array into eax

L1:

add esi , 4

cmp ecx , 1

je doneMax

mov ebx , [esi]

cmp eax , ebx

jg L2

mov eax , ebx

L2:

loop L1

doneMax:

mov edx , offset str1

call Writestring

call Writeint

call crlf

;to find the min value in the array

mov ecx , LENGTHOF arr

mov esi , [ebp+8]

mov eax , [esi] ; load the first element of the array into eax

L3:

add esi , 4

cmp ecx , 1

je doneMin

mov ebx , [esi]

cmp eax , ebx

jl L4

mov eax , ebx

L4:

loop L3

doneMin:

mov edx , offset str2

call Writestring

call Writeint

call crlf

popad

pop ebp

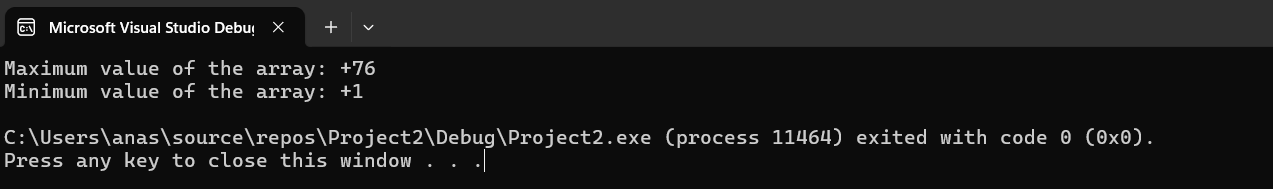
ret

MinMaxArray ENDP

exit

;main ENDP

END main



Q3

Code:

INCLUDE Irvine32.inc

.data

str1 BYTE "Enter the value: " ,0

str2 BYTE "Square of the number: " , 0

.code

main PROC

call LocalSquare

exit

main ENDP

LocalSquare PROC

LOCAL temp : DWORD

enter 0 , 0

mov edx , offset str1

call Writestring

call Readint

mov temp , eax

mul eax

call crlf

mov edx , offset str2

call Writestring

call Writeint

leave

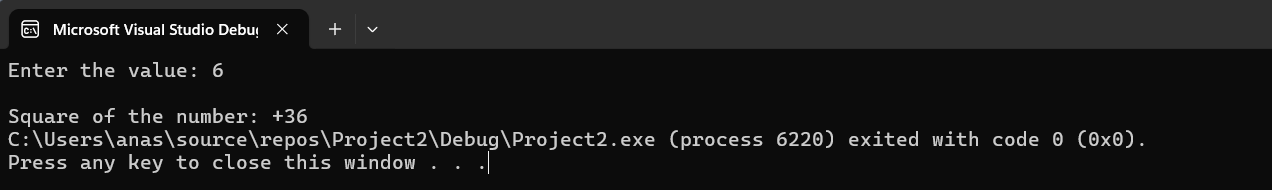
ret

LocalSquare ENDP

;exit

;main ENDP

END main



Q4

Code:

INCLUDE Irvine32.inc

.data

str1 BYTE "Enter the value: " ,0

str2 BYTE " is a prime" , 0

str3 BYTE " is not a prime" , 0

str4 BYTE "Largest prime number: " , 0

var1 DWORD ?

var2 DWORD ?

var3 DWORD ?

var4 DWORD ?

count DWORD ?

notprime\_count DWORD ?

primecount DWORD ?

.code

main PROC

mov count , 0

mov notprime\_count , 0

mov primecount , 0

;taking input from the user

mov edx , offset str1

call Writestring

call Readint

mov var1 , eax

mov edx , offset str1

call Writestring

call Readint

mov var2 , eax

mov edx , offset str1

call Writestring

call Readint

mov var3 , eax

mov edx , offset str1

call Writestring

call Readint

mov var4 , eax

push var1

call CheckPrime

push var2

call CheckPrime

push var3

call CheckPrime

push var4

call CheckPrime

push var1 ; index = 20

push var2 ; index = 16

push var3 ; index = 12

push var4 ; index = 8

call LargestPrime

exit

main ENDP

CheckPrime PROC

push ebp

mov ebp, esp

mov eax , [ebp+8]

cmp eax , 2

jl notprime

mov ecx , 2

L1:

cmp ecx , eax

jge prime ; condition to exit loop and if ecx becomes greater than eax hence it is prime

mov edx , 0

div ecx

cmp edx , 0

je notprime

inc ecx

jmp L1 ; repeat loop

prime:

inc primecount

mov eax , [ebp+8]

call Writeint

mov edx , offset str2

call Writestring

call crlf

jmp done

notprime:

inc notprime\_count

mov eax , [ebp+8]

call Writeint

mov edx , offset str3

call Writestring

call crlf

jmp done

done:

pop ebp

ret

CheckPrime ENDP

;to check the largest primenumbers if all 4 numbers are prime

LargestPrime PROC

push ebp

mov ebp, esp

cmp primecount , 4

jne bye

mov eax , [ebp+8]

mov ebx , [ebp+12]

mov ecx , [ebp+16]

mov edx , [ebp+20]

cmp eax , ebx

jg compare2 ; if eax>ebx

mov eax , ebx

compare2:

cmp eax , ecx ; if eax>ecx

jg compare3

mov eax , ecx

compare3:

cmp eax , edx ; if eax > edx

jg largest

mov eax , edx

largest:

call crlf

mov edx , offset str4

call Writestring

call Writeint

ret

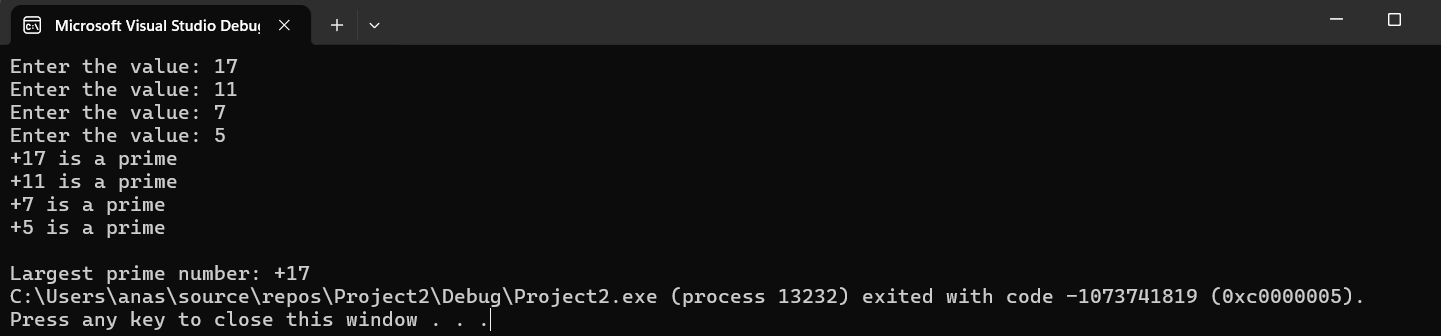
LargestPrime ENDP

bye:

exit

;main ENDP

END main



Q5

Code:

INCLUDE Irvine32.inc

.data

arr DWORD 2, 5, 7, 1, 10, 34, 54, 43, 2, 32, 21, 43, 65, 76, 43, 32, 23, 12, 67, 64

str1 BYTE " , ", 0

.code

main PROC

push offset arr

call BubbleSort

; Printing array

mov ecx, LENGTHOF arr

mov esi, 0

L3:

mov eax, arr[esi\*4]

call Writeint

mov edx, offset str1

call Writestring

inc esi

loop L3

exit

main ENDP

BubbleSort PROC

push ebp

mov ebp, esp

pushad

mov ecx, LENGTHOF arr

dec ecx

;Outer loop

L1:

push ecx

mov esi, 0

mov ebx, ecx

;Inner loop

L2:

mov eax, arr[esi\*4] ; Load the first element of the arr into eax

mov edx, arr[esi\*4+4] ; Load the next element into edx

cmp eax, edx

jle skipSwap

;Swap elements

mov arr[esi\*4], edx ; Store edx into arr[esi]

mov arr[esi\*4+4], eax

skipSwap:

inc esi

dec ebx

jnz L2

pop ecx ; Restore outer loop counter

dec ecx

jnz L1

done:

popad

pop ebp

ret

BubbleSort ENDP

END main

