**COAL LAB 11 (LAB TASKS)**

**TASK # 01:**

**CODE:**

TITLE Task1(test.asm)

include irvine32.inc

.data

var1 dword 5

var2 dword 10

var3 dword 7

Message1 byte "The multiplication of 3 numbers (5,10,7) : ",0

.code

main proc

push var1

push var2

push var3

mov edx, 0

call ThreeProd

add esp, 12

call dumpregs

exit

main endp

ThreeProd proc

push ebp

mov ebp, esp

mov eax, [ebp+8]

mul dword ptr [ebp+12]

mul dword ptr [ebp+16]

mov edx, offset Message1

call WriteString

call writeDec

call crlf

call crlf

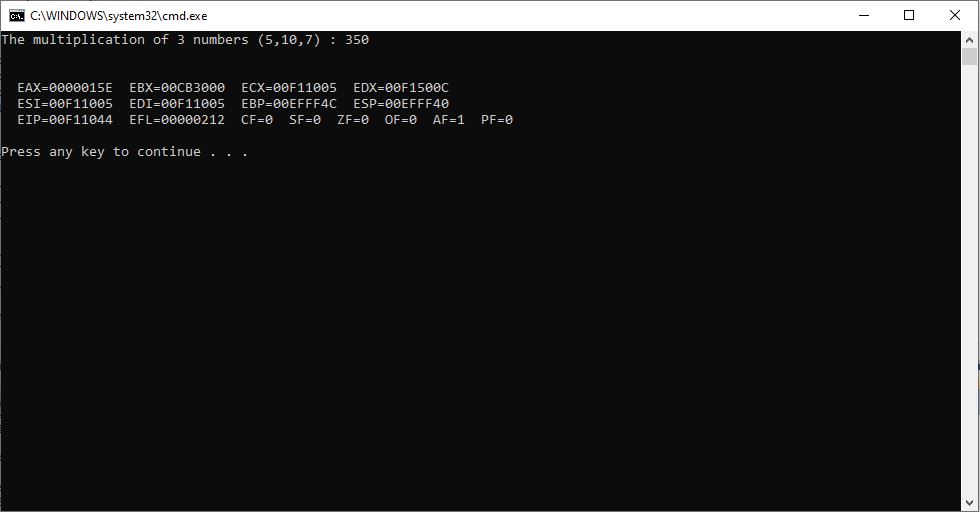
pop ebp

ret

ThreeProd endp

end main

**OUTPUT:**

****

**TASK # 02:**

**CODE:**

TITLE Task2(test.asm)

include irvine32.inc

.data

array byte 20 dup(?)

MaxNum BYTE ?

MinNum BYTE ?

DisplayNumbers byte "The 20 random numbers in array are : ", 0

MaxMessage byte "The maximum number : ", 0

MinMessage byte "The minimum number : ", 0

space byte " ",0

.code

main proc

mov esi, offset array

mov ecx, lengthof array

;randomly assigning values to array of size 20 instead of hard-coded values

call randomize

l1:

mov eax, 100

call randomrange

mov [esi], al

inc esi

loop l1

mov esi, offset array

mov ecx, lengthof array

mov edx, offset DisplayNumbers

call writeString

call crlf

mov edx, offset space

;displaying values of array stored randomly

l2:

mov al, [esi]

call writeDec

call WriteString

inc esi

loop l2

call crlf

call crlf

; passing reference/address of an array through pushing esi to stack (another way of addr/ptr)

MOV ESI, OFFSET ARRAY

PUSH ESI

CALL MinMaxArray

;Displaying maximum\_number

movzx eax, MaxNum

mov edx, offset MaxMessage

call writeString

call writeDec

call crlf

;Displaying minimum\_number

movzx eax, MinNum

mov edx, offset MinMessage

call writeString

call writeDec

call crlf

call crlf

call dumpregs

exit

main endp

; procedure

MinMaxArray proc

push ebp

mov ebp, esp

mov esi, [ebp+8]

MOV AL, [ESI]

MOV MaxNum, AL

MOV MinNum, AL

inc esi

MOV ECX, LENGTHOF array

CheckMaxMin:

mov AL, [ESI]

CMP AL, MaxNum

JNG CheckLess

MOV MaxNum, AL

CheckLess:

CMP AL, MinNum

JGE outside

MOV MinNum, AL

outside:

INC ESI

loop CheckMaxMin

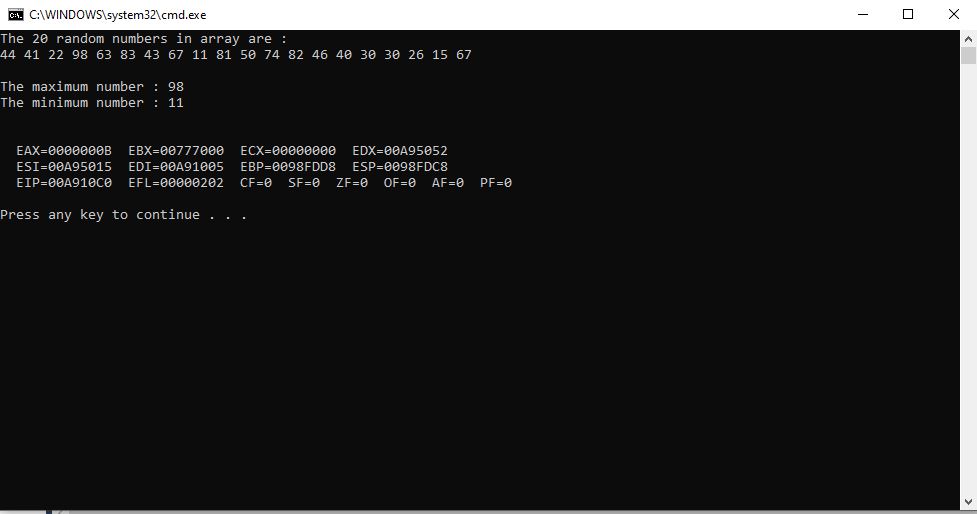
pop ebp

ret

MinMaxArray endp

end main

**OUTPUT:**

****

**TASK # 03:**

**CODE:**

TITLE Task3(test.asm)

include irvine32.inc

.data

MessagePrompt byte "Enter any number : ",0

MessageFinal byte "Square of your entered number : ",0

.code

LocalSquare proto

main proc

invoke LocalSquare

call dumpregs

exit

main endp

; procedure

LocalSquare PROC

local temp: dword

ENTER 0,1

mov edx, offset MessagePrompt

call WriteString

call ReadInt

mov temp, eax

mov EAX, temp

MUL EAX

mov edx, offset MessageFinal

call WriteString

call WriteDec

call crlf

call crlf

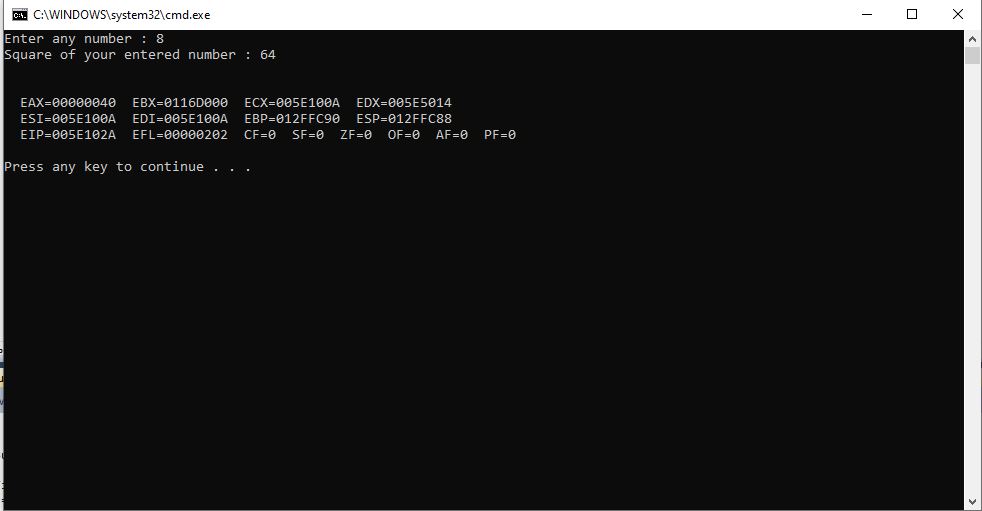
LEAVE

ret 4

LocalSquare ENDP

end main

**OUTPUT:**



**TASK # 04:**

**CODE:**

TITLE Task4(test.asm)

include irvine32.inc

.data

Num dword ?

MessagePrompt byte "Enter any number to calculate factorial : ",0

MessageFinal byte "Factorial of your entered number : ",0

.code

main proc

mov edx, offset MessagePrompt

call WriteString

call ReadInt

mov Num, eax

mov eax, 1

push Num

call Factorial

mov edx, offset MessageFinal

call WriteString

call WriteDec

call crlf

call crlf

call dumpregs

exit

main endp

Factorial proc

ENTER 0,1

mov ebx, [ebp+8]

mul ebx

dec ebx

cmp ebx, 0

JE outside

push ebx

call Factorial

outside:

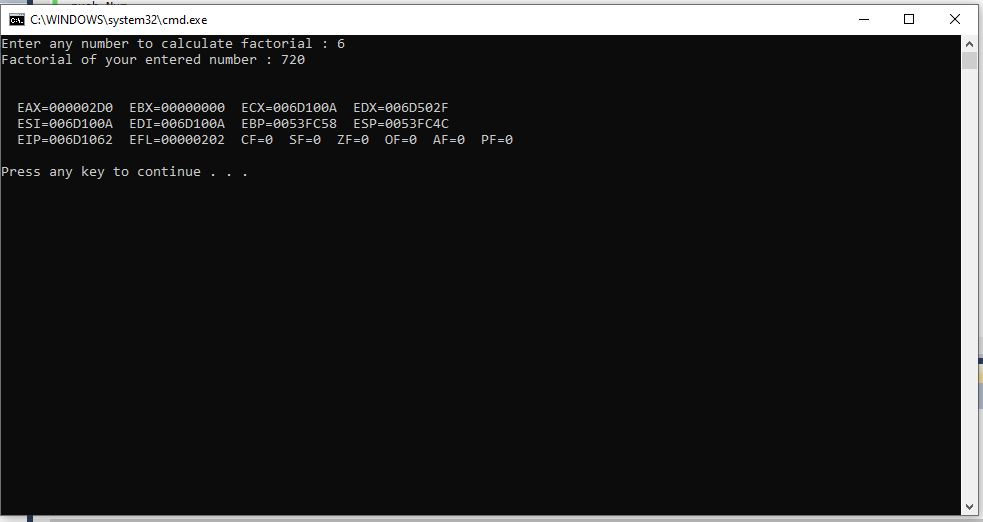
LEAVE

ret 4

Factorial endp

end main

**OUTPUT:**



**TASK # 05:**

**CODE:**

TITLE Task5(test.asm)

include irvine32.inc

.data

Num dword ?

StartTime dword ?

EndTime1 dword ?

TotalTime1 dword ?

StartTime2 dword ?

EndTime2 dword ?

TotalTime2 dword ?

MessagePrompt byte "Enter any number to calculate factorial : ",0

MessageFinal byte "Factorial of your entered number : ",0

Time byte "The time taken by factorial by iteration (in milli seconds) : ",0

Time2 byte "The time taken by factorial by recursion (in milli seconds) : ",0

Greater1 byte "Iterative method took more time than recursive.",0

Greater2 byte "Recrusive method took more time than iteraive.",0

.code

main proc

mov edx, offset MessagePrompt

call WriteString

call ReadInt

mov Num, eax

mov eax, 1

push Num

call GetMSeconds

mov StartTime, eax

call Factorial

mov edx, offset Time

call WriteString

call GetMSeconds

mov Endtime1, eax

sub eax, StartTime

mov TotalTime1, eax

call WriteDec

call crlf

call GetMSeconds

mov StartTime2, eax

mov eax, 1

push Num

call FactorialRecursive

mov edx, offset Time2

call WriteString

call GetMSeconds

mov EndTime2, eax

sub eax, StartTime

mov TotalTime2, eax

call WriteDec

call crlf

call crlf

mov eax, TotalTime1

cmp eax, TotalTime2

JNG two

mov edx, offset Greater1

call WriteString

call crlf

two:

mov edx, offset Greater2

call WriteString

call crlf

call dumpregs

exit

main endp

Factorial proc

ENTER 0,1

mov ebx, [ebp+8]

mov eax, 1

mov ecx, ebx

l1:

mul ecx

loop l1

mov edx, offset MessageFinal

call WriteString

call WriteDec

call crlf

call crlf

leave

ret 4

Factorial endp

FactorialRecursive proc

ENTER 0,1

mov ebx, [ebp+8]

mul ebx

dec ebx

cmp ebx, 0

JE outside

push ebx

call FactorialRecursive

outside:

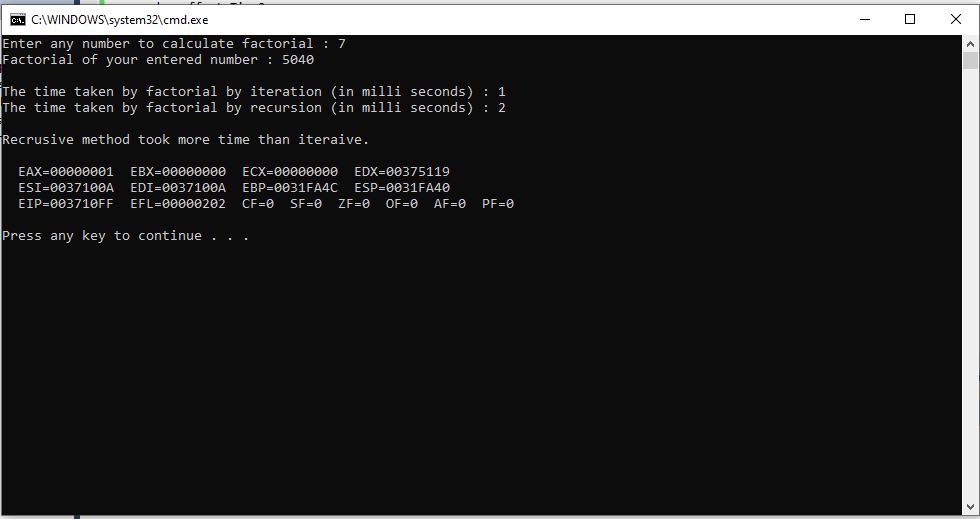
LEAVE

ret 4

FactorialRecursive endp

end main

**OUTPUT:**

****

**TASK # 06:**

**CODE:**

TITLE Task6(test.asm)

include irvine32.inc

.data

input dword 4 dup(?)

counter dword 0

greatest dword ?

message1 byte "Enter a number: ",0

NotPrime byte "Numbers are not prime.",0

Prime byte "Numbers are prime.",0

greatestNumber Byte "The greatest number among these 4 is : ",0

.code

main proc

mov esi, offset input

mov ecx, 4

l1:

mov eax, 0

mov edx, offset message1

call WriteString

call ReadInt

mov [esi], eax

add esi, type input

loop l1

call crlf

mov esi, offset input

push esi

call CheckPrime

call dumpregs

exit

main endp

CheckPrime proc

push ebp

mov ebp, esp

mov esi, [ebp+8]

mov ebx, 2

mov ecx, 4

checkAll:

mov counter, ecx

mov eax, [esi]

mov edx, 0

div bx

movzx ecx, ax

mov eax, 1

cmp [esi], eax

JE OUTSIDE

check:

mov eax, [esi]

mov edx, 0

cmp eax, ebx

JE nextiteration

div bx

cmp dx, 0

JE OUTSIDE

inc edx

loop check

nextiteration:

add esi, type input

mov ecx, counter

loop checkAll

JMP Next

OUTSIDE:

mov edx, offset NotPrime

call WriteString

call crlf

jmp quit

Next:

mov edx, offset prime

call writeString

call crlf

mov esi, offset input

push esi

call LargestPrime

quit:

pop ebp

RET 4

CheckPrime endp

LargestPrime proc

push ebp

mov ebp, esp

mov esi, [ebp+8]

; check if 1st is largest

mov eax, [esi]

cmp eax, [esi+4]

JNG check2

cmp eax, [esi+8]

JNG check2

cmp eax, [esi+12]

JNG check2

mov greatest, eax

mov edx, offset greatestNumber

call WriteString

call WriteDec

call crlf

JMP Outside

; check if 2nd is largest

check2:

mov eax, [esi+4]

cmp eax, [esi+8]

JNG check3

cmp eax, [esi+12]

JNG check3

mov greatest, eax

mov edx, offset greatestNumber

call WriteString

call WriteDec

call crlf

JMP Outside

; check if 3rd is largest

check3:

mov eax, [esi+8]

cmp eax, [esi+12]

JNG check4

mov greatest, eax

mov edx, offset greatestNumber

call WriteString

call WriteDec

call crlf

JMP Outside

; check if 4th is largest

check4:

mov eax, [esi+12]

mov greatest, eax

mov edx, offset greatestNumber

call WriteString

call WriteDec

call crlf

JMP Outside

Outside:

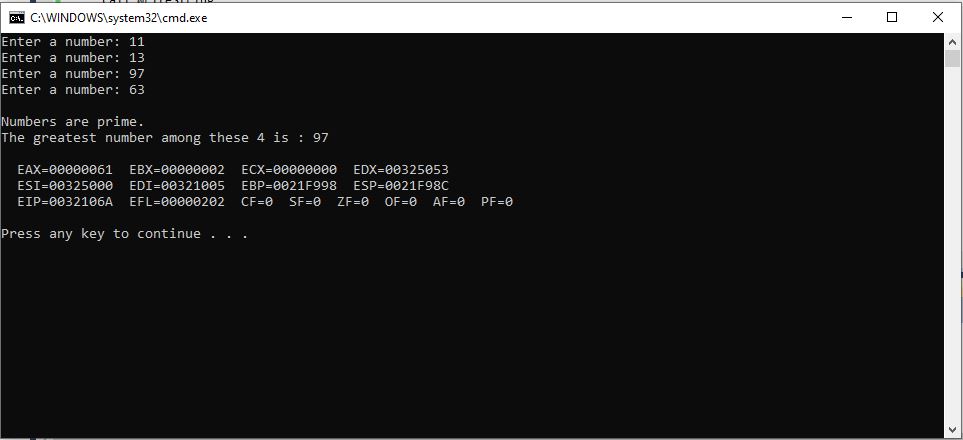
pop ebp

ret 4

LargestPrime endp

end main

**OUTPUT:**

****