**LAB 11 TASKS**

**Q1:**

INCLUDE irvine32.inc

.data

var1 DWORD 2

var2 DWORD 2

var3 DWORD 2

.code

main PROC

push var3

push var2

push var1

call ThreeProd

call writedec

exit

ThreeProd PROC

push ebp

mov ebp, esp

mov eax, [ebp + 16]

mov edx, [ebp + 12]

mul edx

mov edx,[ebp + 8]

mul edx

pop ebp

ret

ThreeProd ENDP

main ENDP

END main

**Graphical user interface, text, application, chat or text message

Description automatically generated**

**Q2:**

**a)**

INCLUDE Irvine32.inc

.data

msg BYTE "Enter 2 values : ",0

msg1 BYTE "ESP values : ",0

.code

main PROC

call Takeinput

exit

Takeinput PROC

mov edx, offset msg

call writestring

call readint

mov ebx, eax

call readint

mov ecx, eax

mov edx, offset msg1

call writestring

mov edx, ecx

mov eax, esp

call writedec

call crlf

call gcd

ret

Takeinput ENDP

gcd PROC

top:

cmp ebx, edx

je next

jg here

sub edx, ebx

jmp top

here:

sub ebx, edx

jmp top

next:

mov edx, offset msg1

call writestring

mov eax, esp

call writedec

call crlf

call display

ret

gcd ENDP

display PROC

mov edx, offset msg1

call writestring

mov eax, esp

call writedec

call crlf

mov eax, ebx

call writedec

ret

display ENDP

exit

main ENDP

end main

**Text

Description automatically generated**

**b)**

INCLUDE Irvine32.inc

.data

arr DWORD 2,4,6,8,10,12,14,16,20,18,22,24,26,28,30,32,34,36,38,40

mi BYTE "Min value in the array = ",0

ma BYTE "Max value in the array = ",0

MIN DWORD ?

MAX DWORD 1000

.code

main PROC

call minmaxarray ;calling minmaxarray function

exit

minmaxarray PROC

mov ecx,LENGTHOF arr

mov esi,0

mov edi,4

mov eax,arr[esi]

mov MIN,eax

l1:

mov eax,arr[esi]

add esi,4

cmp eax,MIN

jl n1

jmp n2

n1:

mov MIN,eax

jmp n2

n2:

loop l1

mov ecx,LENGTHOF arr

mov esi,0

mov edi,4

mov eax,arr[esi]

mov MAX,eax

l2:

mov eax,arr[esi]

add esi,4

cmp eax,MAX

jg p1

jmp p2

p1:

mov MAX,eax

jmp p2

p2:

loop l2

mov edx,OFFSET mi

call Writestring

mov eax,MIN

call Writedec

call crlf

mov edx,OFFSET mi

call writestring

mov eax,MAX

call Writedec

ret

minmaxarray ENDP

main ENDP

end main

**Text

Description automatically generated**

**Q3:**

INCLUDE Irvine32.inc

.data

msg BYTE "Enter number : ",0

msg1 BYTE "Square : ",0

x DWORD ?

.code

main PROC

call localsquare

exit

localsquare PROC

mov edx, offset msg

call writestring

call readint

mov x, eax

mov edx, offset msg1

call writestring

enter 8,0

mov DWORD PTR [ebp-4],eax

mov ebx, [ebp-4]

imul ebx

call writedec

leave

ret

localsquare ENDP

main ENDP

END main

**Graphical user interface, text

Description automatically generated**

**Q4:**

INCLUDE Irvine32.inc

.data

x DWORD ?

msg BYTE "Enter number: ",0

msg1 BYTE "Factorial: ",0

.code

main PROC

mov edx, offset msg

call writestring

call readint

mov x, eax

mov edx, offset msg1

call writestring

push x

call fact

call WriteDec

exit

main ENDP

fact PROC

push ebp

mov ebp, esp

mov eax, [ebp+8]

cmp eax, 0

jg L1

mov eax, 1

jmp L2

L1:

dec eax

push eax

call fact

ReturnFact:

mov ebx, [ebp+8]

mul ebx

L2:

pop ebp

ret 4

fact ENDP

END main

**Graphical user interface, text

Description automatically generated**

**Q5:**

INCLUDE Irvine32.inc

.data

var1 DWORD ?

starttime DWORD ?

msg BYTE " using recursion ",0;facorial

msg1 BYTE "Elapsed time : ", 0

msg2 BYTE " using Loop", 0;factorial

msg3 BYTE "Enter number : ", 0

.code

main PROC

call crlf

mov edx, offset msg

call writestring

call getseconds

mov starttime, eax

call readint

mov var1, eax

push var1

call fact

call WriteDec

call crlf

mov edx, offset msg2

call writestring

call loopfact

exit

main ENDP

getseconds PROC

sub eax, starttime

call crlf

mov edx, offset msg1

call writestring

call writedec

call crlf

mov edx, offset msg3

call writestring

ret

getseconds ENDP

fact PROC

push ebp

mov ebp, esp

mov eax, [ebp+8]

cmp eax, 0

jg L1

mov eax, 1

jmp L2

L1:

dec eax

push eax

call fact

ReturnFact:

mov ebx, [ebp+8]

mul ebx

L2:

pop ebp

ret 4

fact ENDP

loopfact PROC

call getseconds

mov starttime, eax

call readint

mov ecx, eax

mov edx, 1

mov ebx, 1

L1:

imul ebx, edx

inc edx

loop L1

mov eax, ebx

call writedec

ret

loopfact ENDP

END main

**Text

Description automatically generated**

**Q6:**

INCLUDE Irvine32.inc

.data

var1 DWORD 0

nuum DWORD 0

msg1 BYTE " is prime",0

msg2 BYTE " not prime",0

maa BYTE "Largest prime number : ",0

MAX DWORD 1000

arr DWORD 4 DUP(?)

.code

main PROC

mov edi, 0

call readint

mov arr[edi], eax

add edi, 4

call checkprime

call crlf

mov var1, 0

call readint

mov arr[edi], eax

add edi, 4

call checkprime

call crlf

mov var1, 0

call readint

mov arr[edi], eax

add edi, 4

call checkprime

call crlf

mov var1, 0

call readint

mov arr[edi], eax

add edi, 4

call checkprime

call crlf

mov var1, 0

call largestprime

exit

main ENDP

checkprime PROC

mov esi, eax

mov nuum, eax

mov ebx, 1

L1:

cmp esi, 0

je outt

mov edx, 0

mov eax, nuum

mov ecx, ebx

div ecx

cmp edx, 0

jne here

inc var1

here:

dec esi

inc ebx

jmp L1

outt:

cmp var1, 2

je ans

mov edx, offset msg2

call writestring

exit

ans:

mov edx, offset msg1

call writestring

ret

checkprime ENDP

largestprime PROC

mov ecx,LENGTHOF arr

mov esi,0

mov edi,4

mov eax,arr[esi]

mov MAX,eax

l2:

mov eax,arr[esi]

add esi,4

cmp eax,MAX

jg p1

jmp p2

p1:

mov MAX,eax

jmp p2

p2:

loop l2

mov edx,OFFSET maa

call writestring

mov eax,MAX

call Writedec

ret

largestprime ENDP

END main

**Text

Description automatically generated**

**Q7:**

INCLUDE Irvine32.inc

.data

msg BYTE " bubble sort: ", 0

arr word 1,5,8,51,30,16,3,19,29,55,23

.code

main PROC

call bubblesort

main ENDP

bubblesort PROC

mov esi, 0

mov edi, 0

mov edx, 11

mov ecx, 10

l1:

push ecx

mov ecx, edx

l2:

movzx eax, [arr+esi]

movzx ebx, [arr+edi]

cmp eax, ebx

jl en

xchg eax, ebx

mov [arr+esi], ax

mov [arr+edi], bx

en:

add edi, 2

loop l2

pop ecx

add esi, 2

sub edx, 1

mov edi, esi

loop l1

mov edx, OFFSET msg

call writestring

call crlf

mov ecx, 11

mov esi, 0

l3:

movzx eax, arr[esi]

call writedec

call crlf

add esi, 2

loop l3

exit

bubblesort ENDP

END main

**Text

Description automatically generated**

**Q8:**

INCLUDE Irvine32.inc

.data

num DWORD ?

a DWORD ?

x DWORD 1

; now all output messages

str4 BYTE "Enter 3 digit Number : ",0

str1 BYTE " an armstrong ", 0

str2 BYTE " not an armstrong ", 0

str3 BYTE "ESP value : ",0

.code

main PROC

call takeinput

exit

main ENDP

takeinput PROC

mov edx, offset str4

call writestring

call readint

mov a, eax

mov num, eax

mov edx, offset str3

call writestring

mov eax, esp

call writedec

call crlf

call armstrong

ret

takeinput ENDP

armstrong PROC

mov esi, 0

top:

cmp num, 0

je ouut

mov edx, 0

mov eax, num

mov ecx, 10

div ecx

mov ebx, edx

mov x, ebx

imul ebx, ebx

imul ebx, x

add esi, ebx

mov edx, 0

mov eax, num

mov ecx, 10

div ecx

mov num, eax

jmp top

ouut:

cmp esi, a

je hh

mov edx, offset str2

call writestring

call crlf

mov edx, offset str3

call writestring

mov eax, esp

call writedec

exit

hh:

mov edx, offset str1

call writestring

call crlf

mov edx, offset str3

call writestring

mov eax, esp

call writedec

ret

armstrong ENDP

END main

main ENDP

TakeInput PROC

mov edx, offset msg1

call writestring

call readint

mov var, eax

mov orignalnum, eax

mov edx, offset var1

call writestring

mov eax, esp

call writedec

call crlf

call armstrong

ret

TakeInput ENDP

armstrong PROC

mov esi, 0

top:

cmp orignalnum, 0

je ouut

mov edx, 0

mov eax, orignalnum

mov ecx, 10

div ecx

mov ebx, edx

mov m, ebx

imul ebx, ebx

imul ebx, m

add esi, ebx

mov edx, 0

mov eax, orignalnum

mov ecx, 10

div ecx

mov orignalnum, eax

jmp top

ouut:

cmp esi, var

je hh

mov edx, offset msg3

call writestring

call crlf

mov edx, offset var1

call writestring

mov eax, esp

call writedec

exit

hh:

mov edx, offset msg2

call writestring

call crlf

mov edx, offset var1

call writestring

mov eax, esp

call writedec

ret

armstrong ENDP

END main

**Text

Description automatically generated**