**COAL LAB 10**

Q.1)

Include Irvine32.inc

.data

arr dword 6,1,4,3,2,5

space byte " ",0

.code

main proc

push offset arr

push lengthof arr

call bubblesort

mov edx, offset space

mov ecx, lengthof arr

mov esi, 0

L3:

mov eax, [arr+esi]

call writedec

call writestring

add esi, type arr

loop L3

call crlf

exit

main endp

bubblesort proc

push ebp

mov ebp, esp

mov esi, [ebp+12]

mov ecx, [ebp+8]

L1:

mov edx, ecx

mov ecx, [ebp+8]

dec ecx

L2:

mov eax, [esi]

cmp eax, [esi+4]

jb cont

xchg eax, [esi+4]

mov [esi], eax

cont: add esi, type arr

loop L2

mov esi, [ebp+12]

mov ecx, edx

loop L1

pop ebp

ret

bubblesort endp

end main



Q.2)

include irvine32.inc

.data

var1 dword ?

var2 dword ?

var3 dword ?

msg1 byte "Armstrong",0

msg2 byte "Not Armstrong",0

enters byte "Enter 3 digits separately: ",0

.code

main Proc

mov edx,offset enters

call writestring

call takeinput

call calArmstrong

exit

main endp

takeinput proc

mov eax,0

mov ebx,0

mov ecx,0

call readint

mov var1,eax

mov eax,0

call readint

mov var2,eax

mov eax,0

call readint

mov var3,eax

ret

takeinput endp

calArmstrong proc

mov eax,var1

imul eax,var1

imul eax,var1

mov ebx,var2

imul ebx,var2

imul ebx,var2

mov ecx,var3

imul ecx,var3

imul ecx,var3

add eax,ebx

add eax,ecx

mov ebx,var1

imul ebx,100

mov ecx,var2

imul ecx,10

add var3,ebx

add var3,ecx

cmp var3,eax

jne validn

mov edx,offset msg1

jmp exitt

validn:

mov edx,offset msg2

exitt:

call display

ret

calArmstrong Endp

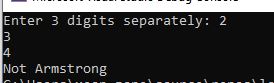
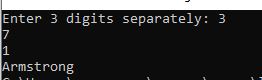
display proc

call writestring

ret

display endp

end main



Q.3)

INCLUDE Irvine32.inc

.data

msg byte " String",0

rev = ($-msg)-1

.code

main PROC

mov ebx,sizeof msg

sub ebx,1

call Reverse

exit

main ENDP

Reverse proc

mov al,msg[ebx]

call writechar

sub ebx,1

cmp ebx,0

jng exitt

call Reverse

exitt:

ret

Reverse ENDP

END main



Q.4)

INCLUDE Irvine32.inc

.data

var DWORD ?

msg1 BYTE "Enter number: ", 0

msg2 BYTE "Square is: ", 0

.code

main PROC

mov edx, OFFSET msg1

call Writestring

call readint

mov var, eax

mov edx, 0

call LocalSquare

mov edx, OFFSET msg2

call writestring

call writedec

exit

main ENDP

LocalSquare PROC

Enter 50, 0

mov [ebp - 8], eax

mov ebx, [ebp - 8]

imul ebx

Leave

ret

LocalSquare ENDP

END main



Q.5)

INCLUDE Irvine32.inc

.data

msg1 byte "Enter Number: ",0

msg2 byte "Factorial: ",0

n dword ?

.code

main PROC

mov edx, offset msg1

call writestring

call readint

mov n,eax

mov eax,1

mov ebx,n

call Fact

mov edx,offset msg2

call writestring

call writedec

exit

main ENDP

Fact PROC

cmp ebx,0

je exitt

mul ebx

sub ebx,1

call Fact

exitt:

ret

Fact ENDP

END main



Q.6)

INCLUDE Irvine32.inc

.data

arr DWORD 4 DUP(0)

count DWORD 0

pCount DWORD 0

msg1 BYTE "Enter Prime Number: ", 0

msg2 BYTE "Prime Number", 0

msg3 BYTE "Not Prime Number", 0

msg4 BYTE "Largest Prime Number: ", 0

.code

main PROC

mov ecx, 4

mov esi, 0

i:

mov edx, offset msg1

call writeString

call readDec

mov arr[esi], eax

add esi, 4

loop i

push offset arr

call checkPrime

exit

main ENDP

checkPrime PROC

pop ebx

pop esi

push ebx

mov edi, 0

mov ecx, lengthof arr

L1:

push ecx

mov ecx, [esi + edi \* 4]

push edi

mov edi, ecx

mov ebx, 1

mov count, 0

L2:

mov eax, edi

cdq

idiv ebx

cmp edx, 0

jne el

inc count

el:

inc ebx

loop L2

cmp count, 2

jne notPrime

mov edx, offset msg2

inc pCount

jmp endd

notPrime:

mov edx, offset msg3

endd:

call writeString

call crlf

pop edi

inc edi

pop ecx

loop L1

cmp pCount, 4

jne exitt

call largestPrime

exitt:

ret

checkPrime ENDP

largestPrime PROC

sub ecx,1

mov eax, [esi]

mov ecx, lengthof arr

L:

cmp eax, [esi+TYPE arr]

jnle next

mov eax, [esi+TYPE arr]

next:

add esi, 4

loop L

mov edx, offset msg4

call writeString

call writeDec

ret

largestPrime ENDP

END main

