

**Q3.(i)**

**a.**

Include Irvine32.inc

.data

op1 sdword 1

op2 sdword 2

op3 sdword 50

X sdword 1

Y sdword 1

.code

main proc

while\_loop:

mov eax, op1

cmp eax, op2

jge end\_while\_loop

inc op1

mov eax, op2

cmp eax, op3

je plus\_two

mov eax, Y

add eax, 10

mov X, eax

jmp while\_loop

plus\_two:

mov eax, Y

add eax, 2

mov X, eax

jmp while\_loop

end\_while\_loop:

exit

main endp

end main

**b.**

Include Irvine32.inc

.data

VAL1 SDWORD 2

VAL2 SDWORD 1

VAL3 SDWORD 1

X SDWORD 2

.code

main PROC

MOV eax, VAL1

CMP eax, VAL2

JLE else\_statement

MOV eax, VAL2

CMP eax, VAL3

JLE else\_statement

MOV X, 10

JMP endd

else\_statement: MOV X, 20

endd: exit

main ENDP

END main

**(ii)**

Minimum PROC uses eax ebx ecx

CMP edx, eax

JLE L1

XCHG edx, eax

L1: CMP eax, ebx

JLE L2

XCHG edx, ebx

L2: CMP edx, ecx

JLE L3

XCHG edx, ecx

L3: ret

Minimum ENDP

**(iii)**

**a.**

CopyTables PROC USES esi edi ecx

MOV esi, OFFSET Table1

MOV edi, OFFSET Table2

MOV ecx, LENGTHOF Table1

CLD

REP MOVSB

ret

CopyTables ENDP

**b.**

Search PROC USES ecx esi eax

MOV ecx, [esp + 16]

MOV esi, [esp + 20]

MOV eax, [esp + 24]

MOV edi, -1

L1:

INC edi

CMP al, [esi + edi]

LOOPNZ L1

JZ L2

INC edi

L2:

ret 12

Search ENDP

**(iv) a.**

Include Irvine32.inc

.code

Binomial PROC n:DWORD, k:DWORD

CMP k, 0

JE Base

MOV ecx, n

CMP ecx, k

JE Base

DEC n

INVOKE Binomial, n, k

DEC k

INVOKE Binomial, n, k

ret

Base: INC eax

ret

Binomial ENDP

main PROC

MOV eax, 0

INVOKE Binomial, 5, 2

CALL WriteDec

exit

main ENDP

end main

**b.**

Include Irvine32.inc

.code

Power PROC x:DWORD, n:DWORD

CMP n, 0

JE Base

DEC n

INVOKE Power, x, n

MUL X

Base:

ret

Power ENDP

main PROC

MOV eax, 1

MOV edx, 0

INVOKE Power, 7, 3

CALL WriteDec

exit

main ENDP

end main

**(v).**

Include Irvine32.inc

.code

Fibonacci PROC n:DWORD

CMP n, 0

JE Base

CMP n, 1

JE Base

DEC n

INVOKE Fibonacci, n

DEC n

INVOKE Fibonacci, n

ret

Base:

ADD eax, n

ret

Fibonacci ENDP

main PROC

CALL ReadInt

MOV ebx, eax

MOV eax, 0

INVOKE Fibonacci, ebx

CALL WriteDec

CALL CRLF

exit

main ENDP

end main

**(vi).**

Include Irvine32.inc

.data

array DWORD 7, 2, 5, 9, 1, 8

.code

main PROC

MOV ecx, LENGTHOF array - 1

L1: MOV esi, 0

MOV edx, LENGTHOF array - 1

L2: CMP edx, 0

JE L5

MOV eax, array[esi]

ADD esi, TYPE array

CMP eax, array[esi]

JB L3

MOV ebx, array[esi]

MOV array[esi], eax

SUB esi, TYPE array

MOV array[esi], ebx

DEC edx

ADD esi, TYPE array

JMP L2

L3: DEC edx

JMP L2

L5: LOOP L1

MOV esi, OFFSET array

MOV ecx, LENGTHOF array

MOV ebx, TYPE array

CALL DumpMem

exit

main ENDP

END main

**(vii)**

Include Irvine32.inc

.data

array DWORD 7, 2, 5, 9, 1, 8

i DWORD 0

j DWORD 0

.code

main PROC

MOV ecx, LENGTHOF array - 1

L1:

MOV esi, i

MOV edi, i

MOV j, esi

MOV edx, LENGTHOF array

L2: CMP edx, esi

JE L4

MOV edi, j

MOV eax, array[edi \* TYPE array]

CMP array[esi \* TYPE array], eax

JAE L3

MOV j, esi

MOV edi, esi

L3: INC esi

JMP L2

L4: MOV esi, i

PUSH array[edi \* TYPE array]

PUSH array[esi \* TYPE array]

POP array[edi \* TYPE array]

POP array[esi \* TYPE array]

INC i

LOOP L1

MOV esi, OFFSET array

MOV ecx, LENGTHOF array

MOV ebx, TYPE array

CALL DumpMem

exit

main endp

END main

**Q4.**

**(i).**

Include Irvine32.inc

.data

Sequence\_Number WORD ?

Revision\_Count BYTE ?

Status BYTE ?

Sensor\_Data WORD ?

.code

main PROC

MOV Sequence\_Number, ax

AND Sequence\_Number, 0000111111111111b

SHR eax, 12

MOV Revision\_Count, al

AND Revision\_Count, 00000111b

SHR eax, 3

MOV Status, al

AND Status, 00000001b

SHR eax, 1

MOV Sensor\_Data, ax

exit

main endp

END main

**(ii).**

mul\_dec PROC USES ebx ecx

MOV cx, ax

SHL ax, 4

MOV bx, cx

SHL bx, 2

ADD ax, bx

MOV bx, cx

SHL bx, 1

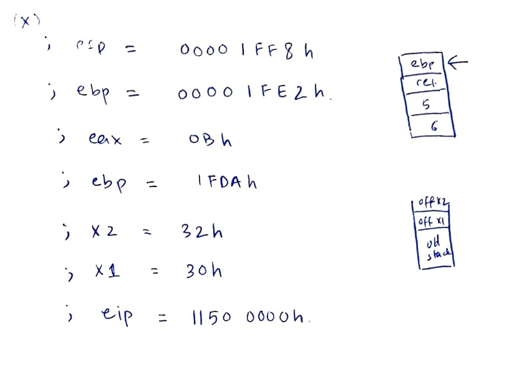
ADD ax, bx

ADD ax, cx

ret

mul\_dec ENDP

**(iii).**

****

**(iv).**

Include Irvine32.inc

.data

str1 BYTE "This is the source string", 0

t BYTE LENGTHOF str1 DUP(0)

flags BYTE 32 DUP(0)

.code

copy PROC USES eax ecx esi edi

CLD

MOV eax, 0

Top: CMP BYTE PTR [esi], 0

JE OutB

MOV al, [esi]

MOV ch, 8

DIV ch

MOV cl, ah

MOV ch, 1

SHL ch, cl

MOV ah, 0

TEST flags[eax], ch

JNZ NoCopy

OR flags[eax], ch

MOVSB

JMP Top

NoCopy: INC esi

JMP Top

OutB: ret

copy ENDP

main PROC

MOV esi, OFFSET str1

MOV edi, OFFSET t

CALL copy

exit

main ENDP

END main

**(v).**

Include Irvine32.inc

.data

arr DWORD 1000 DUP(?)

foundAt BYTE " the number was found at: ", 0

message1 BYTE "Enter your number: ", 0

.code

WriteFound PROC USES eax, val:DWORD, index:DWORD

MOV eax, val

CALL WriteDec

MOV edx, OFFSET foundAt

CALL WriteString

MOV eax, index

CALL WriteDec

CALL CRLF

ret

WriteFound ENDP

search PROC, val:DWORD, index:DWORD

MOV eax, index

CMP eax, LENGTHOF arr

JE Base

MOV eax, val

MOV esi, index

CMP arr[esi \* TYPE arr], eax

JNE Next

INVOKE WriteFound, val, index

Next: INC index

INVOKE search, val, index

Base:

ret

search ENDP

main PROC

MOV edx, OFFSET message1

CALL WriteString

CALL ReadInt

INVOKE search, eax, 0

exit

main ENDP

end main

**(vi).**

Include Irvine32.inc

.data

moon BYTE 20 DUP('0'), 0

.code

star\_array PROC

ENTER 20, 0

LEA esi, [ebp + 20]

MOV edi, OFFSET moon

MOV ecx, 20

L1:

MOV BYTE PTR [esi + ecx - 1], '\*'

MOV BYTE PTR [edi + ecx - 1], 'x'

LOOP L1

LEAVE

ret

star\_array ENDP

main proc

call star\_array

exit

main endp

end main

**(vii).**

Include Irvine32.inc

.data

.code

MyFun PROC, a:DWORD, b:DWORD

MOV eax, a

CMP eax, 5h

JLE Base

DIV b

INVOKE MyFun, eax, b

Base: ret

MyFun ENDP

main proc

exit

main endp

invoke MyFun, 1, 2

end main

**(viii).**

Include Irvine32.inc

.data

ArraySearchValues DWORD 20 DUP(?)

ArrayValues DWORD 1000 DUP(?)

foundAt BYTE " found at index ", 0

.code

WriteFound PROC USES eax edx

CALL WriteDec

MOV edx, OFFSET foundAt

CALL WriteString

MOV eax, LENGTHOF ArrayValues

SUB eax, ecx

CALL WriteDec

CALL CRLF

ret

WriteFound ENDP

search PROC

MOV ecx, LENGTHOF ArraySearchValues

MOV esi, OFFSET ArraySearchValues

L1: PUSH ecx

MOV eax, [esi]

MOV ecx, LENGTHOF ArrayValues

MOV edi, OFFSET ArrayValues

L2: REPNE SCASD

JNE L3

CALL WriteFound

JMP L2

L3: POP ecx

ADD esi, TYPE ArraySearchValues

LOOP L1

ret

search ENDP

main proc

call search

exit

main endp

end main

**(ix).**

Include Irvine32.inc

.data

uni\_name BYTE "FAST NATIONAL UNIVERSITY", 0

ascii BYTE LENGTHOF uni\_name DUP(0)

sitya BYTE "SITYA", 0

higher BYTE "ABCD SITYA ABCD", 0

.code

searchSITYA PROC

MOV al, 'S'

MOV edi, OFFSET higher

MOV ecx, LENGTHOF higher

CLD

REPNE SCASB

JNZ NotFound

DEC edi

MOV esi, OFFSET SITYA

REPE CMPSB

NotFound: ret

searchSITYA ENDP

toASCII PROC

MOV esi, OFFSET uni\_name

MOV edi, OFFSET ascii

MOV ecx, LENGTHOF uni\_name

CLD

REP MOVSB

MOV esi, OFFSET ascii

MOV ecx, LENGTHOF ascii

MOV ebx, TYPE ascii

call DumpMem

ret

toASCII ENDP

main proc

call searchSITYA

main endp

end main