Ubaid Ali

I D #

Assignmant # 04

Assembly language

ANSWERS:-

1) inc vol 2

2) sub eax, vol 3

3) mov ax, vol 4
sub val 2, ax

4) CF = 0 , SF = 1

5) OF = 1 , SF = 1

6) mov ax , 7ffoh
add al, 10h, a: CF = 1
SF = 0 ZF = 1, OF = 0

add oh, 1; b: CF = 0 SF = 1 ZF = 0 OF = 1

add ax, 2; 2; C: CF = 0 SF = 0 ZF = 0 OF = 0

19) mov al 80h

add al, 80h

21)
```
mov eax, TYPE myBytes ; a. 1
mov eax, LENGTH myBytes ; b. 4
mov eax, SIZEOF myBytes ; c. 4
mov eax, TYPE myword ; d. 2
mov eax, LENGTHOF myword ; e. 4
mov eax, size of myword ; f. 8
mov eax, size of mystring ; g. 5
```

22) mov dx, word PTR myBytes

23) mov al, BYTES PTR myword+1

24) mov eax, DWORD PTR myBytes

7)
```
mov esi, OFFSET myBytes
mov al, [esi] ; a. AL = 10h
mov al, [esi+3] ; b AL = 40h
mov esi, OFFSET myword+2 ; c.
                AX = 0038h
mov ax, [esi]
mov edi, 8

mov edx, [myDoubles+edi] ; d
```

EDX=3
mov edx, my Doubles[edi] ; EEDX=3
mov ebx, my Pointer

mov eax, [ebx+4] ; EEAX=2

8) mov esi, OFFSET my Bytes
mov ax, [esi] ; a AX= 20 1ch

mov eax, DWORD PTR my words
; b EAX= 00 3B00 8Ah

x mov ax, [esi] x
x mov esi, my WORD x

mov esi, my Pointer
mov ax, [esi+2] ; c. AX=000

mov ax, [esi+6] ; e. AX=000
mov ax, [esi+4] ; e. AX=000

9) The Program does not stops. because the first loop instruction decrement Ecx to Zero.

The second loop instruction decrements ECX to FFFFFFFFH, causing the outer loop the repeat

10)
```
.DATA
count DWORD
.CODE
mov eax,0
mov ecx,Lo: outer loop counter
L1:
mov count, eax
mov eax, 3
mov eax, 5 inner loop counter
1
L2:
    add eax, 5
Loop L2; repeat inner loop
mov ecx, count
Loop L1; repeat outer loop
```

11)
```
mov ax, word ptr three
mov bx, word ptr three+2
mov three, bx
mov word ptr three+2 ax.
```

12)
```
xchg A,B
xchg A,C
xchg A,D
```

13)
```
mov al, 0FFh
add al, 1
```

20)
```
mov al 0FFh
inc al
jz INC.overflow

mov bc,1
dec bc
jz DEC.overflow
INC. overflow
DEC. overflow
```

18)
```
data:
    invarray DWORD 10000h,
```

(6)

2000h, 30000h, 40000h

code.

main Proc

mov edi, OFFSET inventory
mov ecx, LENGTHOF intarray
mov eax, 0

L1
add aex, [edi]
add edi, TYPE intarray
Loop L1

invoke ExitProcess, 0

main endP
end main.

2b)

data.
my byte 10h, 20h, 30h, 40h.
my words WORD 3DUP(3) 200h

my words DLABEL (DWORD)
my words WORD 3 DUP(?); 200h

code
mov eax, my words D

27) Programming Name: big Endian to little Endian.

```
.386
model flat, stdcall

.stack 4096
ExitProcess PROTO, dwExitCode: DWORD

.data
bigEndian BYTE 12h, 34h, 56h
          78h,
littleEndian DWORD ??

.code
main PROC

mov al, [bigEndian+3]
mov BYTE PTR [littleEndian], al

mov al, [bigEndian+1]
mov BYTE PTR [littleEndian+2], al

mov al, [bigEndian]
mov BYTE PTR [littleEndian+3], al

INVOKE ExitProcess, 0
```

main ENDP
END main

2) .386
.data
array WORD 0,2,5,9,10
new ARRAY DWORD [LENGTHOF array]
DUP(?)

.code
main PROC
mov ecx, LENGTHPE array
mov ESI, OFFSET array
mov EDI, OFFSET newArray

L1:

mov EAX, 0
MOV AX, [ESI]
mov [EDI], EAX
ADD ESI, TYPE array
ADD EDI, TYPE newarray

LOOP L1

INVOKE ExitProcess, 0
main ENDP

END main.

29) .386.

.model flat, stdcall
stack 4096
ExitProcess PROTO, dwExitCode:
DWORD

.data
decimal Array DWORD
1, 2, 3, 4, 5, 6, 7, 8

.code
main PROC

MOV ESI, OFFSET decimal Array
MOV EDI, OFFSET
MOV ECX, LENGTHOF decimal
Array-1

L1:
ADD EDI, TYPE decimalArray

L2:
MOV EAX, [ESI]
MOV EBX, [EDI]
XCHG EAX, EBX

MOV [ESI] , EAX
MOV [ESI] , EBX

ADD , ESI , TYPE decimal Array
SUB EDI , TYPE "
DEC ECX

LOOP L2

INVOKE ExitProcess, 0
main ENDP
END main

30) .386

.model flat , stdcall
.stack 4096
ExitProcess proto dwExitCode:
DWORD.

.data
source BYTE This is the
source string", 0 ,lagest
BYTE size EDF source DUP(x)

.code
main PROC

```
mov esi, 0
mov edi, LENTHOF source - 1
mov ecx, size of source
L1:
    mov eax, 0
    mov edial, source [esi]
    mov largest [edi], al
    inc esi
    dec edi
        Loop L1
```

5)

```
my words LABEL DWORD
my words, WORD 3 DUP (2) 2004
```

```
.data
    mov eax, my words D
```

7)

```
INCLUDE IRVINE 32 inc

.data
  val 1 SDWORD 8
  val 2 SDWORD -15
  val 3 SDWORD 20
  final val SDWORD = ?

.code
```

```
main PROC
mov eax, val 2
neg eax ; eax = -15
add eax ; 7 - val 2 + 7
mov ebx , val 3
add ebx , val 1 ; val 3 + val 1
Sub eax , ebx
mov final val, eax
call Dum Regs , displays the
registers
exit
main ENDP
END P main
```

6)

```
mov al, 0FFh
add al , 1 ; CF = 1 , AL = 00
; Try to go to below
    zero:
mov al , 0
Sub al, 1 ; CF = 1, AL = FF
```

3) Parity flag (PF) will be

Set if there is an even number of bits in the message bite.

* Parity flag (PE) will be zero for the message byte having an - add number of bits.

- code

```
mov al, 01110101b
add al, 00000000 ; AL = 01110101
                   PF = 0
```

After the execution of the ADD instruction AL contain the value of the message byte.

since, there are five [5] add number of ones in the all register Thus PF = 0

(4) Any non-zero operand comes the carry flag to be set

Example :-

```
.data
valB  BYTE  1, 0
valC  SBYTE -128
```

.code

```
neg  valB      ; CF = 1, OF = 0
neg  [valB+1]  ; CF = 0, OF = 0
neg  valC      ; CF = 1, OF = 1
```