Ayesha Mehmood
ID #6832

Assignment #04

Assembly language

Answers

1) inc val 2

2) Sub eax, val 3

3) mov ax, val 4
Sub val 2, ax

4) CF = 0, SE = 1

5) OF = 1, SE = 1

6) mov ax, 7FFoh
add al, 10h; a: CF=1 SE = 0 ZF = 1, OF=0

add ah, 1; b: CF = 0 SE = 1 ZF = 0 OF = 1

add ax, 2; c: CF = 0 SE = 0 ZF = 0 OF = 0

②

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, LENGTHOFmyword ; 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, BYTE PTR mywords+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, myDoubles [edi] ; e. EDX = 3
mov ebx, myPointer

mov eax, [ebx+4] ; f. EAx = 2
```

8)

```
mov esi, OFFSET myBytes
mov ax, [esi] ; a. Ax = 2010h

mov eax, DWORD PTR mywords ; b. EAx =
                           003B008Ah

xmov ax, [esi] x
x mov esi, myWORDx

mov esi, myPointer ;
mov ax, [esi+2] ; c. Ax = 0000
```

mov ax, [esi+6] ; d. AX = 0000
mov ax, [esi+4]; e. AX = 0044h

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 to report.

10) • DATA

Count DWORD ?

• CODE
mov eax, 0
mov ecx, 10: outer loop counter
L1:
mov Count, ecx
mov eax, 3
mov ecx, 5 inner loop counter.

L2:

odd 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 bl, 1
dec bl
jz DEC.overflow
```

INC. overflow
DEC. overflow

18)

```
.data
inventory DWORD 10000h, 2000h,
30000h, 40000h

.code
main proc

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

L1

add eax, [edi]
add edi, TYPE inventory
Loop L1

invoke ExitProcess, 0

main endp
end main
```

26)

```
.data
myByte BYTE 10h, 20h, 30h, 40h

mywords : WORD . 3 DUP (?) . 2004

mywords DLABEL DWORD
mywords WORD 3 DUP (?) ; 2000h

• Code
mov eax, mywordsD
```

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,

Little Endian DWORD ?
```

```
.code
 main PROC
mov al , [big Endian+3]
mov BYTE PTR [little Endian], al

mov al, [big Endian +2]
mov BYTE PTR [little Endian+1], al

mov al, [ big Endian +1]
 mov BYTE ; PTR [little Endian
        +2], al

 mov al, [ big Endian]
 mov BYTE PTR [little Endian+3], al

 INVOKE ExitProcess , 0

 main ENDP

 END main
```

28)

```
 .386
 .model Flat ; stdcall
 . stack 4096
 ExitProcess PROTO, dwExitCod : DWORD
```

(9)

```
.data
array WORD 0,2,5,9,10
newArray DWORD (LENGTHOF array
          DUP(?)

.code
 main PROC

 mov ecx, LENGTHOF 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
decimalArray DWORD 1,2,3,4,5,6,7,8

.code
main PROC

mov ESI , OFFSET decimalArray
mov EDI , OFFSET  "
mov ecx , LENGTHOF .decimalArray-1

L1 ;
   ADD EDI , TYPE decimalArray
   Loop L1

mov ecx ; LENGTHOF decimalArray

L2 ;

   mov eAx , (ESI)
   mov EBx (EDI)
   XCHG EAx , EBx
```

```
mov [ESI], EAX
mov [EDI], EBX

ADD , ESI , TYPE decimal Array
Sub EDI , TYPE "
DEC ECX
.
Loop L2

INVOKE Exit process, 0
main ENDP
END main
```

30)

- 38 b
- model flat , stdcall
- Stack : 4096

```
Exit process proto , dw Exit code
DWORD

.data
source BYTE " This is the
source string" 0 largest
BYTE size EOF source DUP (x)

.Code
main PROC
```

```
mov    esi , 0
mov    edi , LENTHOf   source - 1
mov    ecx . size OF   source

L1
    mov    ean ,0
    mov    edial ,source   [esi]
    mov    largest   [edi] , al
    inc    esi
    dec    edi
    loop   L1

INVOKE   Exit process , 0
main     ENDP
END   main
```

25)

```
my words   LABEL   DWORD
my words   .WORD   3 Dup (?) ,2000h

. data
  mov   eax , my words   D
```

27)

```
INCLUDE   Irvine 32 inc

. data
Val 1   SDWORD   8
Val 2   SDWORD   -15
```

(13)

```
Val 3    SDWORD  20

Finalval    SDWORD =?

.   code

    main   PROC

mov   eax   val 2
neg   eax  :  eax  = -15
odd   eax  :  7  - val 2  +7

mov    ebx , vol 3
odd    ebx , val 1 , vol 3 + val 1

Sub  eax , ebx

mov  finalval, eax
call  Dum Regs ; displays  the registers

exit
main ENDP
END main
```

16) mov al, OFFh
add al, 1 ; CF = 1, AL = 00
; Try to go to below zero:
mov al, 0
Sub al, 1 ; CF = 1, AL = FF

13) * Parity flag (PF) will be set if there is an even number of 1 bits in the message bite.

* Parity flag (PE) will be zero; for the message byte having an add number of 1 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) odd numbers of ones in the AL register. Thus PF=0

14) Any non-zero operand causes 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
```