ID NO 14306

Semester 5th

Subject Assemble language

Assignment NO 4

Submitted to Sir Amin

USe the following data for Questions 1-5

```
data
val1  BYTE  10h
val2  WORD  8000h
val3  DWORD 0FFFFh
val4  WORD  7FFFh
```

Q1 Write an instruction that increments val2.

Ans `inc val2`

Q2 write an instruction that Subtracts val3 from EAX.

Ans `Sub eax, val3`

Q3 Write instruction that Substract val4 from val2.

Ans
```
mov ax, val4
Sub val2, ax
```

Q4 If Val2 is incremented by 1 using the ADD instruction. What will be the values of the of CF and SF?

Ans CF = 0, SE = 1

Q5 If Val4 is incremented by 1 using the ADD instruction, what will be the value of the OF and SF?

Ans OF = 1 ; SF = 1

Q6 Where indicated write down the value of the: CF, SF, ZF, and OF after each instruction has executed.

```
mov ax, 7FF0h

add al, 10h ; a. CF = 1 SF = 0 ZF = 0
add ah, 1   ; b. CF = 0 SF = 1 ZF = 1
add ax, 2   ; c. CF = 0 SF = 0 ZF = 0
```

Q7 Fill the requested register value on the right side of the following instruction sequence

Ans

```
. mov esi, OFFSET myBytes
. mov al, [esi]      ; a. AL = 10h
. mov al, [esi+3]    ; b. AL = 40h
. mov esi, OFFSET myWords+2
. mov ax, [esi]      ; c. AX = 0032h
. 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 C. AX =
```

Q8 Fill in the requested values on the right side of the following instruction sequence.

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

```
. mov eax, DWORD PTR myWords
  ; b. EAX = 003B008Ah
. mov esi, myPointer
. mov ax, [esi+2]  ; c. AX = 0000
. mov ax, [esi+6]  ; d. AX = 0000
. mov ax, [esi-4]  ; e. AX = 0044h
```

Q9 What will be the final value of EAX in this example?

```
mov eax, 0
mov ecx, 10    ; outer loop counter.
L1:
mov eax, 3
mov eax, 5     ; inner loop counter
L2:
add eax, 5
Loop L2        ; repeat inner loop
Loop L1        ; repeat outer loop
```

Ans 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 repeat.

Q10 Revise the code from the per Preceding question So the outer loop Counter is not erased when the inner loop Starts.

```
.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:
add eax,5
loop L2; repeat inner loop
```

```
. mov ecx, count
: loop L1 ; repeat outer loop.
```

Q11 Write a sequence of MOV instructions that will exchange the upper and lower words in a doubleword variable named three.

Ans
```
. mov ax, word Ptr three
. mov bx, word Ptr three+2
. mov three, bx
. mov word Ptr three+2. ax
```

Q12 Using the XCHG instruction to move than three times, reorder the values in four 8-bit registers from the order A, B, C, D, to B, C, D, A.

Ans
```
. Xchg A, B
. Xchg A, C
. Xchg A, D
```

Q13 Transmitted message often include a parity bit whose value is combined with a data byte to produce an even number of 1 bits. Suppose a message byte in the AL register contains 01110101. Show how you could use the PF combined with an arithmetic instruction to determine if this message byte has even or odd parity.

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

* Parity flag (PF) will be zero for the message byte having an odd 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, these are five (5) odd number of ones in the AL register. Thus, PF=0.

Q14 write Code using byte operands that adds two negative integers and Cause the OF to be Set.

Ans Any non-Zero operand Cause the Carry Flag to be Set.

Example:

```
.data
Val B  BYTE  1,0
Val C  SBYTE -128
.Code
neg Val B      ; CF=1, OF=0
neg [Val B+1]  ; CF=0, OF=0
neg Val C      ; CF=1, OF=1
```

Q15 write a sequence of two instructions the use addition to set the ZF and CF at the same time.

Ans
```
mov al, 0FFh
add al, 1
```

Q16 write a sequence of two instructions that set the CF using Substraction.

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

Q17 Implement the following arithmetic expression in assembly language : EAX = - val2 + 7 - val3 + val1. Assume that val1, val2; and

Val3 are 32-bit integer variables.

Ans Solution:-

```
INCLUDE Irvine32.inc
.data
Val1 SDWORD 8
Val2 SDWORD -15
Val3 SDWORD 20

.Code
main PROC
mov eax val2
neg eax ; eax = -15
add eax ; 7 ; -Val2+7
mov ebx ; val3
add ebx ; val1 ; val3+val1
Sub eax , ebx
mov final val, eax
Call DumpRegs ; display the registers
```

```
exit
main ENDP
END main
```

Q18 write a loop that iterates a doubleword array and Calculates the Sum of its elements using a Scale factor with indexed addressing.

Ans

```
.Data
        intarray DWORD 10000h,
20000h, 30000h, 40000h
.Code
main Proc
mov edi, OFFSET intarray
mov ecx, LENGTHOF intarray
L1
add eax, [edi]
add edi, TYPE intarray
Loop L1
invoke ExitProcess, 0
```

```
main endp
end main
```

Q19 Write a Sequence of two instructions that set both the CF and OF at the Same time.

Ans
```
mov al, 80h
add al, 80h
```

Q20. Write a Sequence of instructions Showing how the ZF could be used to indicate unsigned overflow after executing INC and DEC instructions.

Ans
```
mov al, 0FFh
inc al
jZ INC -over flow

mov bl, 1
dec bl
```

```
jz DEC - overflow

INC - overflow
DEC - overflow
```

Q21 what will be the value of EAX after each of the following instruction execute?

```
mov eax, TYPE myBytes ; a. 1
mov eax, LENGTHOF myBytes ; b. 4
mov eax, SIZEOF myBytes ; c. 4
mov eax, TYPE myWords ; d. 2
mov eax, LENGTHOF myWords ; e. 4
mov eax, SIZEOF myWords ; f. 8
mov eax, SIZEOF myString ; g. 5
```

Q22 write a Single instruction that moves the first two bytes in myBytes to the DX register. The resulting value will be 2010h.

```
mov dx, WORDPTR myBytes
```

Q23 write an instruction that move the second byte in myBytes words to the EAX registes.
AL

Ans mov al, BYTE PTR myWords+1

Q24 write an instruction that moves all four bytes in myBytes to the EAX registes.

Ans mov eax, DWORD PTR myBytes.

Q25 Insert a LABEL directive in the given data that permits myWords to be moved directly to a 32-bit registes.

Ans myWords LABEL DWORD
myWords WORD 3 DUP(?),
2000h

```
. data
    mov eax, mywords D
```

Q26 Insert a LABEL directive in the given data Pexmints myBytes to be moved directly to a 16-bit registex.

Ans

```
. data
myByte BYTE 10h, 20h, 30h,
40h
mywordS WORD 3 DUP(?), 200h
mywordS DLABEL DWORD
mywords WORD 3 DUP(?), 200h
. Code
    mov eax, mywords D
```

Q27 Write a Program that uses the variable below and Mov instruction to COPY the value from big Endian to littleEndian.

reversing the order of the bytes. The number's 32bits value is understood to be 12345678 hexadecimal.

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

- Program Name:-
  BigEndian to Little Endian
- 386
  model Flat, StdCall
- Stack 4096
  ExitProcess PROTO, dwExitCode:
  DWORD

- Data
  big Endian BYTE 12h, 34h, 56h, 78h
  Little Endian DWORD?

- Code
  main PROC

```
mov al, [bigEndian+3]
mov BYTE PTR [littleEndian], al
mov al, [bigEndian+2]
mov BYTE PTR [littleEndian+1], al
mov al, [bigEndian+1]
mov BYTE PTR [littleEndian+2+1], al
mov al, [bigEndian]
mov BYTE PTR [littleEndian+3], al

INVOKE ExitProcess, 0
main ENDP
END main
```

Q28 write a Program that uses a loop to copy all the elements from an unsigned word (16-bit array into an unsigned doubleword (32bit) array.

Ans

```
.386
model Flat, StdCall
```

```
.Stack 4096
ExitProcess PROTO ,dwExitCode:
DWORD

.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
```

Q29 Use a loop with indirect or indexed addressing to reverse the elements of an integer array in place. Do not copy the elements to any other array. Use the SIZEOF, TYPE, and LENGTHOF operations to make the program as flexible as possible if the array size and type should be changed in the future.

Solution:-

```
.386
.model flat, stdcall
```

```
.Stack 4096
ExitProcess PROTO, dwExit
Code : DWORD

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

.Code
main PROC
MOV ESI, OFFSET decimalArray
MOV EDI, OFFSET decimalArray
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] , EAY
Mov  [EDI] , EBX

ADD  ESI , TYPE decimal Array
Sub  EDI , TYPE decimal Array
DEC  ECX

LOOP L2

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

Q30 write a program with a loop and indirect addressing that copies a string from source to target, reversing the character order in the process. Use the following variables:-

```
Source BYTE "This is the
Source String",0
target BYTE SIZEOF Source
DUP('#')
```

- Solution:-

```
.386
.model flat.StdCall
.Stack 4096
ExitProcess PROTO,dwExitCode:
DWORD
```

```
.data
Source BYTE "This is the Source
String",0
target BYTE SIZEOF Source
DUP('#')
```

```
.Code
main PROC
```

```
mov esi,0
mov edi, LENGTHOF Source-1
mov ecx, SIZEOF Source

L1:
    mov eax,0
    mov edial ,Source [esi]
    mov target [edi],al
    inc esi
    dec edi
    loop L1

INVOKE ExitProcess,0
main ENDP
END main
```