Department of Computer Science

Microprocessor & Assembly Language

Assignment Number: 04

Name	_____	**Mohammad Basir**
ID	_____	**13142**
INSTRUCTOR Name	_____	**Mohammad Amin**
Semester	_____	**8th**

Use the following data for Questions 1-5:

.data val1 BYTE

10h val2 WORD 8000h

val3 DWORD 0FFFFh

val4 WORD 7FFFh

Q.1

Write an instruction that increments val2 ?

ANSWER :

inc val2 .

Q.2

Write an instruction that subtracts val3 from EAX?

ANSWER :

sub eax,val3

Q.3

Write instructions that subtract val4 from val2?

ANSWER

mov ax,val4 sub val2, ax .

Q.4

If **val2** is incremented by 1 using the ADD instruction, what will be the values of the CF and SF?

ANSWER

CF = 0, SF = 1 .

Q.5

If **val4** is incremented by 1 using the ADD instruction, what will be the values of the OF and SF?

ANSWER:

OF = 1, SF = 1 .

Q.6

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

```
mov ax,7FF0h add al,10h
;  a. CF = 1 SF = 0 ZF = 1 OF = 0 add ah,1
;  b. CF = 0 SF = 1 ZF = 0 OF = 1 add ax,2
;  c. CF = 0 SF = 0 ZF = 0 OF = 0
```

Use the following data definitions for Questions 7 and 8:

```
myBytes BYTE
10h,20h,30h,40h
myWords WORD
8Ah,3Bh,72h,44h,66h
myDoubles DWORD
1,2,3,4,5 myPointer
DWORD myDoubles
```

Q.7

Fill in the requested register values on the right side of the following instruction sequence:

```
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 = 003Bh
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
```

Q.8

Fill in the requested register values on the right side of the following instruction sequence:

```
mov esi,OFFSET myBytes
```

```
  mov ax,[esi]                    ; a. AX = 2010h
  mov eax,DWORD PTR               ; b. EAX =
myWords  mov      esi,myPointer 003B008Ah
  mov ax,[esi+2]                  ; c. AX = 0000
  mov ax,[esi+6]                  ; d. AX = 000
  mov ax,[esi-4]                  ; e. AX = 0044h
```

Q.9

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 ecx,5         ; inner loop
counter L2:
add eax,5
loop L2      ; repeat inner loop
loop L1      ; repeat outer loop
```

ANSWER:

The program does not stop, because the first LOOP instruction decrements ECX to zero. The second LOOP instruction decrements ECX to FFFFFFFFh, causing the outer loop to repeat.

Q.10

Revise the code from the preceding question so the outer loop counter is not erased

```
.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
```

Q.11

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

ANSWER

```
Mov ax, word ptr three

Mov bx, word ptr three+2

Mov three, bx

Mov word ptr three+2, ax
```

Q.12

Using the XCHG instruction no more than three times, reorder the values in four 8-bit registers from the order A, B, C, D to B, C, D, A.

ANSWER :

```
xchg A, B
xchg A, C
xchg A, D
```

Q.13

Transmitted messages 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.

ANSWER:

```
mov al,01110101b
add al,00000010b
```

Q.14

Write code using byte operands that adds two negative integers and causes the OF to be set.

ANSWER:

```
mov eax,-1
add eax,1
```

Q.15 Write a sequence of two instructions that use addition to set the ZF and CF at the same time.

ANSWER

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

Q.16

Write a sequence of two instructions that set the CF using subtraction.

ANSWER

```
mov al, 3
sub al, 4
```

Q.17

Implement the following arithmetic expression in assembly language: EAX = –val2 + 7 –val3 + val1. Assume that val1, val2, and val3 are 32-bit integer variables.

```
data
Uarray WORD   1000h, 2000h, 3000h, 4000h
```

```
Sarry WORD      -1, -2, -3, -4
val1 SDWORD   -8
val2 SDWORD   -15
val3 SDEORD     20
.code
main PROC


mov eax, va12; EAX = -15
neg eax, EAX = 15 0000000fh
add eax, 7; EAX = 7 00000007h
sub eax, va13; EAX = 2 00000002h
add eax, va11; EAX = -13 DDDDDDD0h
call DumpRegs
exit
main ENDP
```

Q.18 Write a loop that iterates through a doubleword array and calculates the sum of its elements using a scale factor with indexed addressing.

```
mov edi, OFFSET intarray
mov ecx, LENGTHOF intarray
```

```
mov eax, 0
L1:
add eax, [edi]
add edi, TYPE intarray
loop L1
invoke ExitProcess,0
main endp
end main
```

Q.19

Write a sequence of two instructions that set both the CF and OF at the same time.

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

Q.20

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

ANSWER :

Setting the Zero flag after INC and DEC to indicate unsigned overflow:

```
mov al,0FFh
inc al
jz overflow_occurred
mov bl,1
dec bl
jz overflow_occurred
```

Use the following data definitions for Questions 21–26:

```
.data
myBytes BYTE 10h,20h,30h,40h
myWords WORD 3 DUP(?),2000h
myString BYTE "ABCDE"
```

Q.21

What will be the value of EAX after each of the following instructions execute?

ANSWER

```
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
```

Q.22

Write a single instruction that moves the first two bytes in **myBytes** to the DX register. The resulting value will be 2010h.

ANSWER :

MOV DX, WORD PTR myBytes

Q.23

Write an instruction that moves the second byte in **myWords** to the AL register.

ANSWER:

MOV AL, BYTE PTR myWords+1

Q.24

Write an instruction that moves all four bytes in myBytes to the EAX register.

ANSWER :

MOV EAX,DWORD PTR myBytes

Q.25

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

ANSWER**:**

```
myWords LABEL DWORD
        myWords WORD 3
DUP(?),2000h
   .data
    MOV
EAX,myWordsD
```

Q.26

Insert a LABEL directive in the given data that permits **myBytes** to be moved directly to a 16-bit register.

ANSWER:

```
myBytesW LABEL DWORD
  myBytes BYTE 10h,20h,30h,40h
```

.data
MOV EAX,myBytesW

Q.27

Write a program that uses the variables below and MOV instructions to copy the value from **bigEndian** to **littleEndian**, reversing the order of the bytes. The number's 32-bit value is understood to be 12345678 hexadecimal.

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

Q.28

Write a program that uses a loop to copy all the elements from an unsigned Word (16-bit) array into an unsigned doubleword (32-bit) array.

ANSWER :

```
; Program Name:  bigEndian to LittleEndian

.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+2]
  mov BYTE PTR [littleEndian+1],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
```

Q.29

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 operators to make the program as flexible as possible if the array size and type should be changed in the future.

ANSWER **:**

```
.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 decimalArray
    ADD EDI, SIZEOF decimalArray
    SUB EDI, TYPE decimalArray
    Mov ecx, LENGTHOF decimalArray
  L2:
    MOV EAX, [ESI]
    MOV EBX, [EDI]
    XCHG EAX, EBX
    MOV [ESI], EAX
    MOV [EDI], EBX
    ADD ESI, TYPE decimalArray
    SUB EDI, TYPE decimalArray
    DEC ECX
```

```
    LOOP L2


INVOKE ExitProcess,0

main ENDP

END main
```

Q.30

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('#')
```

ANSWER **:**

```
.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 al,source[esi]
    mov target[edi],al
    inc esi
    dec edi
    loop L1

INVOKE ExitProcess,0
main ENDP
END main
```

---

the end ***