Date: .....

NAME :- FAYYAZ MUHAMMAD

ID :- 14294

Subject :- Microprocessor And Assembly language

Department :- BS(CS)

Teacher :- Sir Amin

ASSIGNMENT NUMBER

04

ASSIGNMENT NO:- 4

1) Q:- No 1

Ans) inc val 2.

2) Q:- No 2

Ans) Sub eax, val 3

3) Q:- No 3

Ans) MOV ax, val 4
Sub val 2, ax

4) Q:- No 4

Ans) CF=0, SF=1

5) Q:- No 5

Ans) OF=1, SF=1

6) Q:- No 6

```
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
```

Q. No 7

b)
```
mov esi, OFFSET myBytes
mov al, [esi]          ; a. AL = 10h
mov al, [esi+3]        ; b. AL = 40h
mov esi, OFFSET myword+2 ; c. AX = 0036h
mov ax, [esi]
mov edi, 8

mov edx, [myDoubles+edi]   ; d. EDx = 3
mov edx, myDoubles[edi]    ; e. EDx = 3

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

Q. No 8

b)
```
mov esi, OFFSET myBytes
mov ax, [esi]          ; a. Ax = 2010h
mov eax, DWORD PTR mywords ; b. EAx = 00100020h
x mov ax, [esi]x
x mov esi, mywordx

mov esi, myprinter
mov ax, [esi+2]        ; c. Ax = 0000
mov ax, [esi+6]        ; d. Ax = 0000
mov ax, [esi-4]        ; e. Ax = 0044h
```

Q. No 9

A) The program does not stop, because the first loop instruction decrement ECX to Zero. The second Loop instruction decrements ECX to FFFFFFFFh, causing the outer loop to repeat.

Q. No 10

A) .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 No 11

A) mov ax, word ptr Three
mov bx, word ptr Three + 2
mov Three, bx
mov word ptr Three + 2, ax

Q. No 12

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

Q. No 13

A)
- 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 there are five (5) odd number of ones in the AL register. Thus, PF = 0

Q. No. 14

A) Any non-Zero operand causes the carry flag to be set.

Example:

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

```
.Code
    neg  val B        ; CF=1, OF=0
    neg  [val B+1]    ; CF=0, OF=0
    neg  valc         ; CF=1, OF=1
```

Q No 15

b)
```
mov al, OFFh
add al, 1
```

Q No 16

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

Q No 17

Ans) Solution:-

```
INCLUDE Irvine32.inc
.data
val1 SDWORD 8
val2 SDWORD -15
val3 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 Dumpregs ; display the registers

exit
main ENDP
END main
```

Q. No 18

b)

```
.data
    intarray DWORD 10000h , 20000h ,
    30000h , 40000h

.Code
    main Proc

mov edi , OFFSET intarray
mov ecx , LENGTH OF intarray
mov eax , 0

L1
add eax , [edi]
add edi , TYPE intarray
```

```
LOOP L1
invoke ExitProcess, 0

main endp
end main
```

Q No: 19

b)
```
mov al, 80h
add al, 80h
```

Q No: 20

b)
```
mov al, 0FFh
inc al
JZ INC_overflow

mov bl, 1
dec bl
JZ DEC_overflow

INC_overflow
DEC_overflow
```

Q No 21

a)
```
mov eax, TYPE myBytes;      a. 1
mov eax, LENGTHOF myByte;   b. 4
mov eax, SIZEOF myByte;     c. 4
mov eax, TYPE myWord;       d. 2
```

mov eax, LENGTHOF myword ; e. 4
mov eax, SIZEOF myword ; F. 8
mov eax, SIZE OF my String ; g. 5.

Q No 22.

A). mov dx, WORD PTR ~~myword+1~~ myBytes

Q No 23.

A) mov al, BYTE PTR myWords + 1

Q No 24

A) mov eax, DWORD PTR myBytes

Q No 25

A) myWords LABEL DWORD
myWords WORD 3 DUP (?), 2cccch

.data
mov eax, myWords D

Q No 26.

A). .data
myBytes BYTE 10h, 20h, 30h, 40h
myWords WORD 3 DUP(?), 200h

my words DI ABEL DWORD
my words WORD 3 DUP (?), 2ooch

.Code
mov eax, my words D.

Q. No 27

B).

Programming Name: big Endian to little End.

• 386
model Flat, std call
• Stack 4096
Exit Proten PROTO, dw Exit Code: DWORD

• data
big Endian Byte 12h, 34h, 56h, 78h
Little Endian DWORD?

• Code
main PROC
mov al, [big Endian + 3]
mov Byte PTR [little Endian], 01

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
```

Q No 28.

Q).

```
.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
```

Q No 29

B) Solution:

```
.386
.model flat, std
.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
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 decimal Array
Dec Ecx

Loop L2

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

Q. No 30

Ans) Solution:

```
.386
.model flat, stdcall
.stack 4096
Exit Process PROTO, dw exit code : 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 Exit Process,
main ENDP
END main
```