NAME SAEEDA NAZ

ID 14332

Semester ~~4th~~ 5th

Department BS (CS)

Assignment

No

(4)

Page 1 Assignment No 4

1) Question No 1

Ans inc val 2

2) Question No 2

Ans Sub eax, val 3

3) Question No 3

Ans mov ax, val 4
Sub val 2, ax

4) Question No 4

Ans CF = 0, SF = 1

5) Question No 5

Ans OF = 1, SF = 1

6) Question 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
```

7) Question No 19

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

8) Question No 21

Ans :-

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

9) Question No 22

Ans :-
```
mov dx, WORD PTR myBytes
```

10) Question No 23

Ans :-
```
mov al, BYTE PTR myWords+1
```

11) Question No 24

Ans :- mov eax,

```
mov eax, DWORD PTR myBytes
```

12) Question No 7

Ans :-

```
mov esi , OFFSET myBytes
mov al , [esi]                ; a. AL = 10h
mov al , [esi+3]              ; b. AL = 40h
mov esi , OFFSET myWord + 2   ; c. AX = 003Bh
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
```

13) Question No 8

Ans :-

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

```
mov  eax, DWORD PTR myWords   ; b. EAX =
x mov  ax, [esi] x                 003B002Ah
x mov  esi , myWORD x

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

14) Question No 9

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.

15) Question No 10

Ans :-

```
.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.
```

16) Question No 11

Ans :-

```
mov ax, word ptr three
mov bx, word ptr three + 2
 mov three, bx
mov word ptr three + 2, ax
```

17) Question No 12

Ans :-

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

18) Question No 15

Ans
```
mov al, OFFh
add al, 1
```

19) Question No. 30.

Ans :-

```
mov al, OFFh
inc al
jz INC_overflow

mov bl, 1
dec bl
jz DEC_overflow

INC_overflow:
DEC_overflow.
```

20) Question No 18

Ans :-

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

```
• code

main proc

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
```

21) Question No 26

Ans :-

```
• data
myByte BYTE 10h, 20h, 30h, 40h
myWords WORD 3 DUP(?), 2000h

myWords DLABEL DWORD
myWords WORD 3 DUP (?), 2000h
```

```
• code
  mov  eax, myWords D
```

33) Question No 27

Ans :-

Programming Name : bigEndian to LittSe Endian

```
• 386
  .model Flat , Std call
  .Stack 4096
  ExitProcess PROTO , dwExitCode : DWORD

• data
  bigEndian BYTE 12h , 34h , 56h , 78h
  littSe Endiagn DWORD ?

• code
  main PROC
    mov al, [bigEndian + 3]
    mov BYTE PTR [littSe Endian], al

    mov al, [bigEndian + 2]
    mov BYTE PTR [littSe Endian + 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
```

23) Question No 28

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
```

Question No 29

Ans :- Solution :-

```
• 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
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 decimalArray
  Sub EDI, TYPE decimalArray
  DEC ECX

  Loop L2

  INVOKE ExitProcess, 0
main ENDP
END main
```

Question No 30

Ans :- 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 edi al , Source [ esi ]
 mov target [ edi ], al
 inc esi
 dec edi
 loop L1

INVOKE ExitProcess , 0
main ENDP
END main
```

Question No 25

Ans

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

.data
mov eax, myWords D
```

Question No 17

Ans :- solution :-

```
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 ; -val2 +7

mov ebx , val 3
add ebx , val 1 ; val 3 + val 1
```

```
Sub   eax , ebx

mov  finalVal , eax
call DumpRegs ; display the registers

exit
main  ENDP
END   main.
```

Question No 16.

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
```

Question No 13

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

Question No 14

Ans :- 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
```