LAB FILE

8086 ASSEMBLY PROGRAMMING

Q.1 Write a program to display ouput in LED.

Ans:

```
#start=led_display.exe#
#make_bin#
name "led"
mov ax,5
out 199,ax
hlt
```

Q.2 Program to add two number and display output in LED.

Ans:

```
#start=led_display.exe#
#make_bin#
name "led"
mov al,10
out 199, ax
mov bl, 20
add al,bl
out 199,ax
hlt
```

Q.3 Program to find avg of 3 number.

Ans:

```
DATA SEGMENT
```

```
    NUM1 DB
    NUM2 DB 10
    NUM3 DB 10
    AVG DB ?
ENDS
CODE SEGMENT
   ASSUME DS:DATA CS:CODE
   START:
   MOV AX,DATA
   MOV DS,AX
   MOV AL,NUM1
   ADD AL,NUM2
   ADD AL,NUM3
   MOV AH,0
   MOV DL,3
   DIV DL
   MOV AVG,AL
    #start=led_display.exe#



out 199,ax
   MOV AH,4CH
   INT 21H
ENDS
END START
```

Q4.**write a program to swap two numbers**

Ans:

```
DATA SEGMENT
    NUM1 DB 5
    NUM2 DB 10

ENDS
CODE SEGMENT
   ASSUME DS:DATA CS:CODE
   START:
    #start=led_display.exe#
   MOV AX,@DATA
   MOV DS,AX
   MOV AL,NUM1
   MOV BL,NUM2
   MOV NUM1,BL
   MOV NUM2,AL
   MOV AH,4CH
   INT 21H
ENDS
END START
```

Q5 – **write a program to subract 16 bit number**

```
data segment
a db 2Ah
b db 13h
c dw ?
data ends

code segment
assume cs:code,ds:data
start:
mov ax,data
mov ds,ax
mov al,a
mov bl,b
sub al,bl
mov c,ax
int 3
code ends
end start
```

Q6-**program to arrange numbers in accending order**

```
DATA SEGMENT
STRING1 DB 99H,12H,56H,45H,36H
DATA ENDS
```

```
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START: MOV AX,DATA
MOV DS,AX

MOV CH,04H

UP2: MOV CL,04H
LEA SI,STRING1

UP1: MOV AL,[SI]
MOV BL,[SI+1]
CMP AL,BL
JC DOWN
MOV DL,[SI+1]
XCHG [SI],DL
MOV [SI+1],DL

DOWN: INC SI
DEC CL
JNZ UP1
DEC CH
JNZ UP2

INT 3
CODE ENDS
END START
```

Q-7 **Write a program to input a string from user and display the string on the center of the** screen

```
PRNT MACRO MSG
   MOV AH,09h
   LEA DX,MSG
   INT 21H
ENDM

DATA SEGMENT
   MSG1 DB "ABC PQR XYZ$"
DATA ENDS

CODE SEGMENT
   ASSUME CS:CODE,DS:DATA
   START:
   MOV AX,DATA
   MOV DS,AX
   MOV AX,0600H
   MOV BH,71H
   MOV CX,0000H ;UPPER LEFT ROW,COLUMN
   MOV DX,0184H ;LOWER RIGHT ROW,COLUMN
   INT 10H
   MOV AH,02H
   MOV BH,00H
   MOV DH,0CH
   MOV DL,23H
```

```
    INT 10H
    PRNT MSG1
    MOV AH,4CH
    INT 21H
CODE ENDS
    END START
```

Q8- **Write a program to reverse a string entered by the user.**

```
Data Segment
  str1 db 'String_Reverse','$'
      strlen1 dw $-str1
  strrev db 20 dup(' ')
Data Ends

Code Segment
  Assume cs:code, ds:data
  Begin:
    mov ax, data
    mov ds, ax
    mov es, ax
    mov cx, strlen1
    add cx, -2
    lea si, str1
    lea di, strrev
    add si, strlen1
    add si, -2
    L1:
```

```
    mov al, [si]
    mov [di], al
    dec si
    inc di
    loop L1
    mov al, [si]
    mov [di], al
    inc di
    mov dl, '$'
    mov [di], dl
  Print:
    mov ah, 09h
    lea dx, strrev
    int 21h
Exit:
    mov ax, 4c00h
    int 21h
Code Ends
End Begin
```

Q9. **Write a program to count number of 1's in the number**

```
DATA SEGMENT
NO DW 5648H
Z DW ?
O DW ?
DATA ENDS
CODE SEGMENT
```

```
ASSUME CS:CODE, DS:DATA
START:
MOV AX, DATA
MOV DS, AX
MOV AX, NO
MOV BX, 00H
MOV CX, 10H
MOV DX, 00H
UP:
ROL AX,1
JC ONE
INC BX
JMP NXT
ONE:
INC DX
NXT:
DEC CX
JNZ UP
MOV Z, BX
MOV O, DX
INT 3
CODE ENDS
END START
```

Q10 write a program to check wheather the string is palindrome or not

```
DATA SEGMENT

STRING DB ‘abba’, ‘$’ #STRING TO BE CHECKED, DOLLAR SYM TO INDICATE THE END OF THE STRING

STRING1 DB ‘STRING IS A PALINDROME’, ‘$’

STRING2 DB ‘STRING IS NOT A PALINDROME’, ‘$’

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA

START:

MAIN PROC FAR # MAIN FUNCTION KEY WORD FAR IS USED BECAUSE IT WILL BE IN ANOTHER

SEGMENT

MOV AX, DATA #INTIALIZATION OF THE DS REGISTER

MOV DS, AX

CALL PALINDROME # PROCEDURE CALLED AS PALINDROME FUNCTION WILL BE CALLED

MOV AH,4CH # TERMINATION OF THE PROGRAM AND THE CONTROL WILL BE RETURNED

INT21H

MAIN ENDP

CODE ENDS

END START

PALINDROME PROC

MOV SI, OFFSET STRING # THE OFFSET ADDRESS OF THE STRING ‘ABBA’ WILL BE ASSIGNED TO SI

LOOP1:

MOV AL, [SI] # [SI] MEANS THE CONTENT OF SI IS MOVED TO AL , AL=’A’

JE LABEL1 #JUMP IF EQUAL TO LABEL1, BUT THEY ARE NOT EQUAL THUS SI IS INCREMENTED

INC SI

JMP LOOP1 # CONTINUE WITH THE LOOP TILL EOS ‘$’ IS ENCOUNTERED
```

```
LABEL1:
MOV DI, OFFSET STRING # DI INTIALISED TO THE FIRST ADDRESS OF THE STRING
DEC SI # SI IS POINTING TO DOLLAR THUS DECREMENT SI
LOOP2:
CMP SI,DI
JL OUTPUT1 #JUMP IF LESS TO OUTPUT1 THAT IS WHEN THEY CROOS MIDPOINT
MOV AL, [SI]
MOV BL, [DI]
CMP AL, BL # COMPARE AL AND BL
JNE OUTPUT2 # JUMP IF NOT EQUAL TO OUTPUT2
DEC SI
INC DI
JMP LOOP2
OUTPUT1:
LEA DX, STRING1 #LOAD EFFECTIVE ADDRESS OF STRING1 TO DX USED TO LOAD THE OFFSET
VALUE TO A REGISTER
MOV AH, 09H 3 TO DISPLAY THE MESSAGE OF STRING 1 ON THE DISPLAY
INT 21H
RET
OUTPUT2:
LEA DX, STRING2
MOV AH, 09H
INT 21H
RET
PALINDROME ENDP
END MAIN
```