**1. Apply Assembly Language Programming to enter and display 8-bit no’s**

.model small

.stack

.data

M1 DB 10, 13, "Enter 8 bit number : $"

M2 DB 10, 13, "Display 8 bit number : $"

num DB ?

.code

.startup

MOV AH, 09 ;Display Message on Screen

LEA DX, M1

INT 21H

MOV AH, 01 ;Enter Single Character

INT 21H

SUB AL, 30H

MOV CL, 4 ;Move the character shift 4 bits left

SHL AL, CL

MOV BH, AL ;Move to Memory Location

MOV AH, 01 ;Enter Single Character

INT 21H

SUB AL, 30H

ADD BH, AL ;Add both the characters

MOV num, BH

MOV AH, 09 ;Display Message on screen

LEA DX, M2

INT 21H

MOV BH, num

AND BH, 0F0H ;Performing AND operation to seperate the character

MOV CL, 4

SHR BH, CL ;Shift Right the character 4 bits

ADD BH, 30H

MOV DL, BH ;Display Single Character on screen

MOV AH, 02

INT 21H

MOV BH, num

AND BH, 0FH ;Performing AND operation to seperate the character

ADD BH, 30H

MOV DL, BH ;Display Single character on screen

MOV AH, 02

INT 21H

.exit

End

**2. Apply Assembly Language Programming to covert HEX to BCD.**

.MODEL SMALL

.STACK

.DATA

M1 DB 10, 13, "HEX TO BCD IS : $"

NO DW 0FFFFH

.CODE

DISP MACRO XX

MOV AH, 09

LEA DX, XX

INT 21H

ENDM

.STARTUP

DISP M1

MOV AX, NO

MOV DH, 0

MOV BX, 0AH

MOV CL, 0

BACK:

MOV DX, 0

DIV BX

PUSH DX

INC CL

CMP AX, 0

JNZ BACK

BACK1:

POP DX

ADD DL, 30H

MOV AH, 02

INT 21H

DEC CL

JNZ BACK1

.EXIT

END

**3. Apply Assembly Language Programming to perform addition and subtraction of two 16 bits numbers using macros and procedure.**

.model small

.stack

.data

M1 DB 10,13,"Addition is : $"

M2 DB 10,13,"Subtraction is :$"

NO1 DW 4536H

NO2 DW 2312H

RES DW ?

.code

DISP MACRO XX

MOV AH,09

LEA DX,XX

INT 21H

ENDM

.startup

DISP M1

MOV AX,NO1

ADD AX,NO2

MOV RES,AX

CALL DISP1

DISP M2

MOV AX,NO1

SUB AX,NO2

MOV RES,AX

CALL DISP1

JMP LAST

DISP1 PROC

MOV BX,RES

AND BH,0F0H

MOV CL,4

SHR BH,CL

ADD BH,30H

MOV DL,BH

MOV AH,02

INT 21H ;first digit display ends here

MOV BX,RES

AND BH,0FH

ADD BH,30H

MOV DL,BH

MOV AH,02

INT 21H ;second digit display ends here

MOV BX,RES

AND BL,0F0H

MOV CL,4

SHR BL,CL

ADD BL,30H

MOV DL,BL

MOV AH,02

INT 21H ;third digit display ends here

MOV BX,RES

AND BL,0FH

ADD BL,30H

MOV DL,BL

MOV AH,02

INT 21H ;fourth digit display ends here

RET

DISP1 ENDP

LAST:

.exit

End

**4. Write an ALP To find negatives numbers in the given array.**

.MODEL SMALL

.STACK

.DATA

MSG1 DB 10,13,"NEGATIVE NUMBERS: $"

SPACE DB " $"

ARRAY DB 37H,15H,0AAH,058H,084H ;initialized array

N\_ARRAY DB 5 DUP(0) ;Uninitialized array

.CODE

DISP MACRO XX

MOV AH,09

LEA DX,XX

INT 21H

ENDM

.STARTUP

LEA SI,ARRAY ;Source Array (+ve Array)

LEA DI,N\_ARRAY ;Destination Array (-ve Array)

MOV CL,5 ;+ve Array Length

MOV BL,0 ;-ve Array Length

BACK:

MOV AL,[SI]

AND AL,80H ;To check Positive or negative; 80 -> 1000 0000; Checking First Bit

JZ POSITIVE ;Jump if Number is Positive

MOV AL,[SI]

MOV [DI],AL

INC DI

INC BL

POSITIVE:

INC SI

DEC CL

JNZ BACK

DISP MSG1

LEA DI,N\_ARRAY

BACK1:

MOV BH,[DI]

AND BH,0F0H

MOV CL,4

SHR BH,CL

CMP BH,9

JG AA

ADD BH,30H

JMP AA1

AA:

ADD BH,37H

AA1:

MOV DL,BH

MOV AH,02

INT 21H

MOV BH,[DI]

AND BH,0FH

CMP BH,09

JG AA2

ADD BH,30H

JMP AA3

AA2:

ADD BH,37H

AA3:

MOV DL,BH

MOV AH,02

INT 21H

;MOV AH,13

;INT 21H

INC DI

DEC BL

DISP SPACE

JNZ BACK1

.EXIT

END

**5. Apply Assembly Language Programming to perform string operations. (i)Accept, (ii) Display, (iii) Concatenation (iv) Compare**

.MODEL SMALL

.STACK

.DATA

M1 DB 10,13, "ENTER STRING 1: $"

M2 DB 10,13, "ENTER LENGTH OF STRING 1: $"

M3 DB 10,13, "DISPLAY STRING 1: $"

M4 DB 10,13, "ENTER STRING 2: $"

M5 DB 10,13, "ENTER LENGTH OF STRING 2: $"

M6 DB 10,13, "DISPLAY STRING 2: $"

M7 DB 10,13, "COMPARE STRINGS: $"

M8 DB 10,13, "STRING NOT EQUAL $"

M9 DB 10,13, "EQUAL STRING $"

M10 DB 10,13, "CONCATINATE STRINGS: $"

STR1 DB 50,?,50 DUP(?)

STR2 DB 50,?,50 DUP(?)

L1 DB ?

L2 DB ?

.CODE

DISP MACRO XX

MOV AH,09

LEA DX,XX

INT 21H

ENDM

.STARTUP

DISP M1 ;ENTER STRING 1

MOV AH,0AH ;Take Input of Whole string together

LEA DX,STR1

INT 21H

DISP M2 ;LENGTH OF THE STRING 1

LEA SI, STR1+1

MOV CL,[SI]

MOV L1,CL

ADD CL,30H

MOV DL,CL

MOV AH,02

INT 21H

DISP M3 ;DISPLAY STRING 1

LEA SI,STR1+2

MOV CL,L1

BACK1:

MOV DL,[SI]

MOV AH,02

INT 21H

INC SI

DEC CL

JNZ BACK1

DISP M4 ;ENTER STRING 2

MOV AH,0AH

LEA DX,STR2

INT 21H

DISP M5 ;LENGTH OF THE STRING 2

LEA DI, STR2+1

MOV CL,[DI]

MOV L2,CL

ADD CL,30H

MOV DL,CL

MOV AH,02

INT 21H

DISP M6 ;DISPLAY STRING 2

LEA DI,STR2+2

MOV CL,L2

BACK2:

MOV DL,[DI]

MOV AH,02

INT 21H

INC DI

DEC CL

JNZ BACK2

DISP M7 ;COMPARE STRINGS 1 & 2

MOV CL,L1

MOV CH,L2

CMP CL,CH

JNZ N\_EQUAL

LEA SI,STR1+2

LEA DI,STR2+2

BACK3:

MOV DL,[SI]

CMP DL,[DI]

JNZ N\_EQUAL

INC SI

INC DI

DEC CL

JNZ BACK3

DISP M9 ;CONCATE

JMP CONCATE

N\_EQUAL:

DISP M8

DISP M10

CONCATE:

MOV CL,L1

MOV BL,CL

MOV CH,L2

ADD BL,CH

LEA SI,STR1+2

LEA DI,STR2+2

BACK4:

INC SI

DEC CL

JNZ BACK4

BACK5:

MOV DL,[DI]

MOV [SI],DL

INC SI

INC DI

DEC CH

JNZ BACK5

LEA SI,STR1+2

BACK6:

MOV DL,[SI]

MOV AH,02

INT 21H

INC SI

DEC BL

JNZ BACK6

.EXIT

END

**6. Apply Mixed Language Programming to design a calculator**

#include <iostream.h>

#include <conio.h>

void main()

{

clrscr();

int a,b,result;

int ch;

cout<<"--------------------CALCULATOR------------------\n";

cout<<"\nEnter first number : ";

cin>>a;

cout<<"\nEnter second number : ";

cin>>b;

cout<<"----------------------------------------\n";

do

{

cout<<"\n1.Addition \n2.Subtraction \n3.Multiplication \n4.Division \n5.Reset \n6.Exit"<<endl;

cout<<"----------------------------------------\n";

cout<<"Enter your choice : ";

cin>>ch;

switch(ch)

{

case 1:

{

asm mov ax,a;

asm mov bx,b;

asm add ax,bx;

asm mov result,ax;

cout<<"----------------------------------------\n";

cout<<"Addition of "<<a<<" and "<<b<<" = "<<result<<endl;

cout<<"----------------------------------------\n";

break;

}

case 2:

{

asm mov ax,a;

asm mov bx,b;

asm sub ax,bx;

asm mov result,ax;

cout<<"----------------------------------------\n";

cout<<"Subtraction of "<<a<<" and "<<b<<" = "<<result<<endl;

cout<<"----------------------------------------\n";

break;

}

case 3:

{

asm mov ax,a;

asm mov bx,b;

asm mul bx;

asm mov result,ax;

cout<<"----------------------------------------\n";

cout<<"Multiplication of "<<a<<" and "<<b<<" = "<<result<<endl;

cout<<"----------------------------------------\n";

break;

}

case 4:

{

asm mov ax,a;

asm mov bx,b;

asm div bx;

asm mov result,ax;

cout<<"----------------------------------------\n";

cout<<"Division of "<<a<<" by "<<b<<" = "<<result<<endl;

cout<<"----------------------------------------\n";

break;

}

case 5:

{

cout<<"----------------------------------------\n";

cout<<"\nEnter first number : ";

cin>>a;

cout<<"\nEnter second number : ";

cin>>b;

cout<<"\n----------------------------------------\n";

break;

}

case 6:

{

break;

}

default:

{

cout<<"\n----------------------------------------\n";

cout<<"Wrong Input";

cout<<"\n----------------------------------------\n";

break;

}

}

}while(ch!=6);

getch();

}

**7. Develop program to interface mouse driver drivers.**

.MODEL SMALL

.STACK

.DATA

M1 DB 10,13,"MOUSE DRIVER PRESENT$"

.CODE

DISP MACRO XX

MOV AH,09

LEA DX,XX

INT 21H

ENDM

.STARTUP

MOV AX,0000

INT 33H

CMP AX,0

JE LAST

DISP M1

MOV AX,0004H

MOV CX,0000H

MOV DX,0000H

INT 33H

MOV AX,0007H

MOV CX,0000

MOV DX,00FFH

INT 33H

MOV AX,0008H

MOV CX,0000

MOV DX,00FFH

INT 33H

P:

MOV AX,0001H

INT 33H

MOV AX,0003H

INT 33H

CMP BX,01

JE LEFT

JMP RIGHT

LEFT:

MOV AX,0011H

INT 10H

MOV AH,0CH

INT 10H

RIGHT:

MOV AX,0001H

INT 33H

CMP BX,02

JE LAST

JMP P

LAST:

MOV AX,0000

INT 10H

.EXIT

END