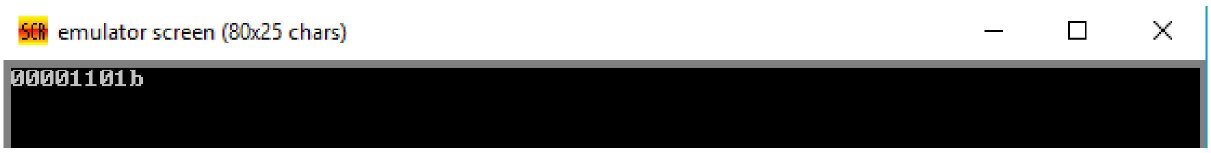
Experiment 1	Write Assembly Language program in 8086 for demonstrating the addition and subtraction Instructions.
Date: 01/11/2018	

Program

```
name "add_sub"
      org 100h
      mov al,09H
      mov bl,05h
      add bl,al
      sub bl,01h

     mov cx,08h
print:  mov
ah,02h
      mov dl,'0'
      test
bl,10000000b
      jz zero
      mov dl,'1'
 zero: int 21H
      shl bl,01h
      loop print

      mov dl,'b'
      int 21h
      mov al,0
      int 16h
      Ret
```

Output:



```
00001101b
```

Result:-

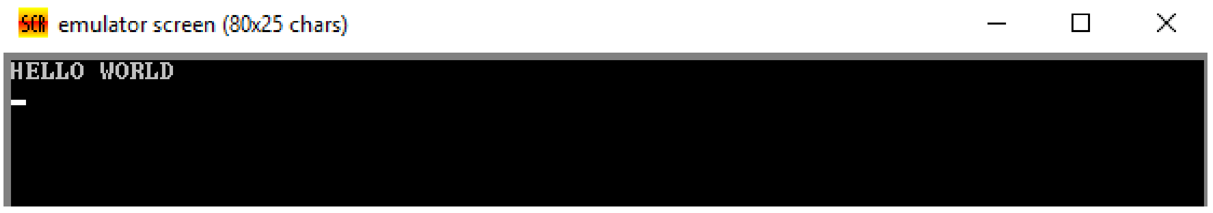
The program was executed successfully.

Experiment 2	Write Assembly Language program in 8086 to print “HELLO WORLD”.
Date: 01/11/2018	

Program:

```
NAME "hello"
           ORG  100h
            JMP START
MSG:     DB "HELLO WORLD",0DH,0AH,24H
START:  MOV DX,MSG
            MOV AH,09H
            INT 21H
            MOV AH,0
            INT 16H
    RET
```

Output :



```
HELLO WORLD
_
```

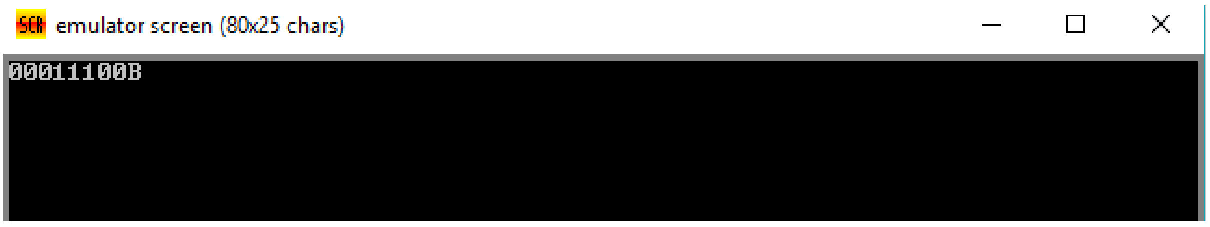
Result: - The program was executed successfully.

Experiment 3	*Write Assembly Language program in 8086 for counting the number of characters in a given string of a zero terminated string.*
Date: 01/11/2018	

Program: -

```
Name "counter"
       org 100H
       JMP start
       str DB 'abcdefg hijklmnop qrstuvwxyz',0
start:   LEA BX,str
         mov AX,0
compare: cmp [BX],0
         JE done
         INC AX
         INC BX
         JMP compare

done:    MOV BX,AX
         MOV CX,08
print:   MOV AH,2
         MOV Dl,'0'
         TEST BL,10000000B
         JZ zero
         MOV DL,'1'
zero:    INT 21H
         SHL BL,01H
         LOOP print
         MOV DL,'B'
         INT 21H
         MOV AL,0
         INT 16H
    RET
```

Output: -



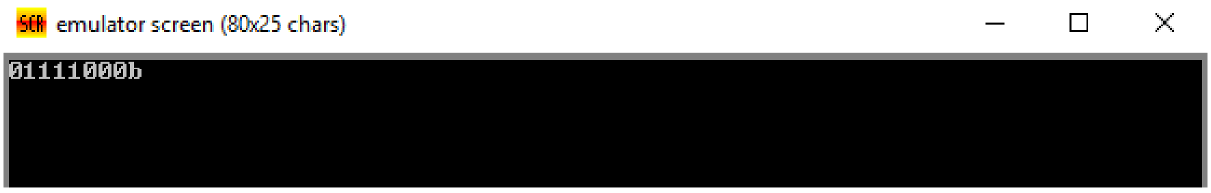
```
00011100B
```

Experiment 5	Write Assembly Language program in 8086 for finding the factorial of 5.
Date: 01/11/2018	

Program: -

```
name  "fact"
          org 100h
          jmp start
          n db 5
start:  mov al,01h
          mov cl,00h
          mov dl,n
next:   cmp cl,dl
          je done
          inc cl
          mul cl
          jmp next
done:  mov bl,al
          mov cx,08h
print:  mov ah,02h
          mov dl,'0'
          test bl,10000000b
          jz zero
          mov dl ,'1'
zero:   int 21h
          shl bl,01h
          loop print
          mov dl,'b'
          int 21h
          mov al,0
          int 16h
   ret
```

Output: -



```
01111000b
```

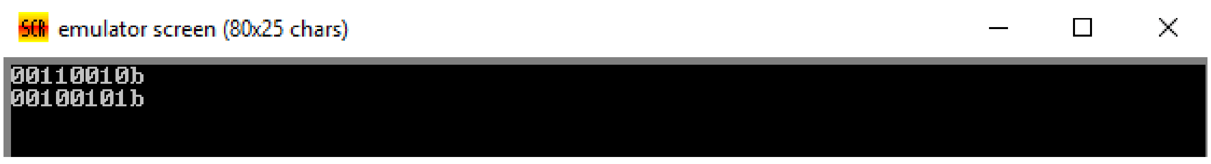
Result: - The program was executed successfully.

Experiment 6	*Write Assembly Language program in 8086 for the conversion from centigrade (Celsius) to Fahrenheit calculation and vice-versa.*
Date:	

Program: -

```
name"celsius"
        org 100h
        jmp start
        tc db 10
        tf db 100
        result1 db  "result in farenheit"
        result2 db  "result in celcius"

 start: mov cl,tc
        mov al,09h
        imul cl
        mov cl,05h
        idiv cl
        add al,20h
        mov result1,al
        mov bl,result1
        call print
        mov cl,tf
        sub cl,20h
        mov al,05h
        imul cl
        mov cl,09h
        idiv cl
        mov result2,al
        mov bl,result2
        call print
        mov ah,0
        int 16h
   ret


        print proc
        pusha
        mov cx,08h
   p1:  mov ah,02h
        mov dl,'0'
        test bl,10000000b
        jz zero
        mov dl,'1'
zero:   int 21h
        shl bl,1
        loop p1
        mov dl,'b'
        int 21h
        mov dl,0dh
        int 21h
        mov dl,0ah
        int 21h
        popa
        ret
```

Output: -



```
00110010b
00100101b
```

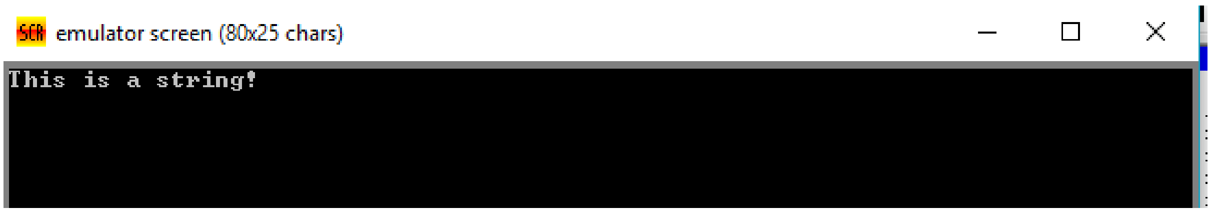
Result: - The program was executed successfully.

Experiment 7	*Write Assembly Language program in 8086 for reversing a string.*
Date: 01/11/2018	

Program: -

```
            name"reverse"
            org 100h
            jmp start
  string:   db '!gnirts a si siht $'
   start:   lea bx,string
            mov si,bx
            next_byte:cmp [si],'$'
            je end
            inc si
            jmp next_byte
     end:   dec si
            do_reverse:cmp bx,si
            jae done
            mov al,[bx]
            mov ah,[si]
            mov [si],al
            mov [bx],ah
            inc bx
            dec si
            jmp do_reverse
    done:   lea dx,string
            mov ah,09h
            int 21h
            mov ah,00h
            int 16h
            ret
```

Output: -



```
This is a string!
```

Result: - The program was executed successfully.

Experiment 8

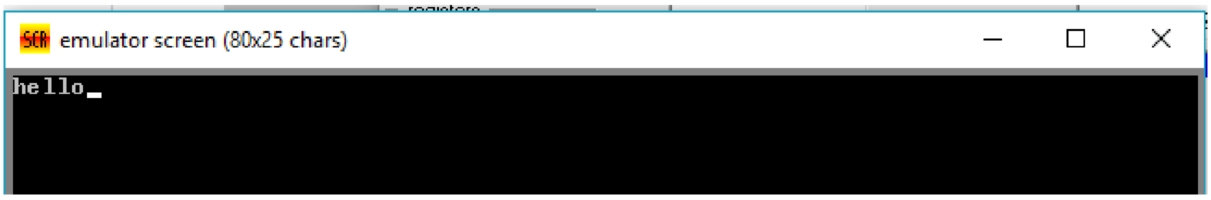
Date: 01/11/2018

Write Assembly Language program in 8086 to make lowercase to uppercase string.

Program: -

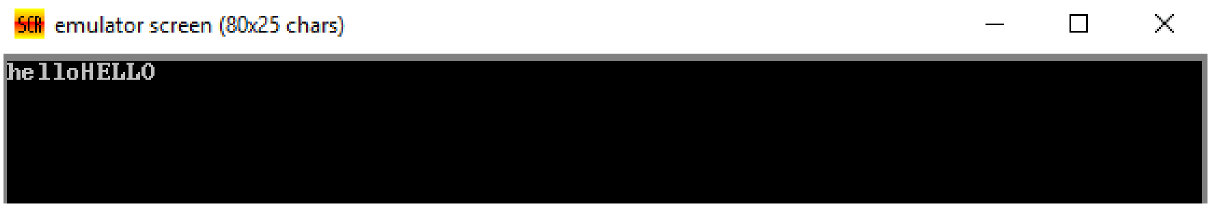
```
  name"lower_upper"
              org 100h
              jmp start
              string db 20h,22h dup('?')
              new_line db0dh,0ah,'$'

       start: lea dx,string
              mov ah,0ah
              int 21h

              mov bx,dx
              mov ah,00h
              mov al,ds:[bx+1]
              add bx,ax
              mov byte ptr[bx+2],'$'

                 lea dx,new_line
                 mov ah,09h
                 lea bx,string
                 mov ch,0h
                 mov cl,[bx+1]
                 jcxz null
                 add bx,2

    upper_case:  cmp byte ptr[bx],'a'
                 jb ok
                 cmp byte ptr[bx],'z'
                 ja ok
                 and byte ptr[bx],11011111b

            ok:   inc bx
                  loop upper_case


                  lea dx,string+2
                  mov ah,09h
                  int 21h
                  mov ah,0
                  int 16h

          null:   ret
```

Input:-



```
hello_
```

<u>Output:</u> -



```
helloHELLO
```

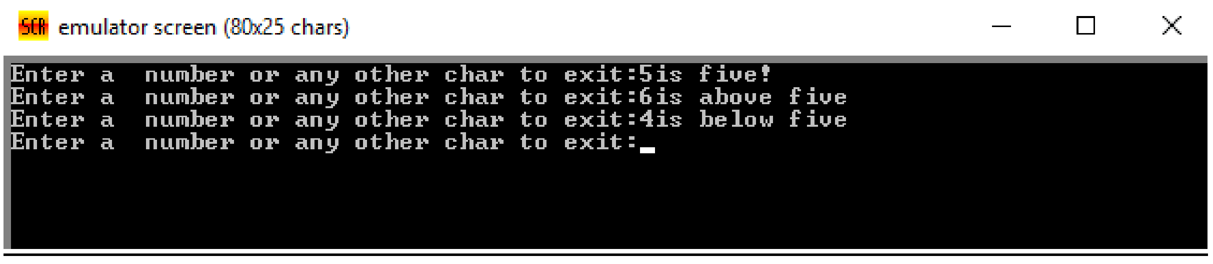
Result: - The program is executed successfully.

Experiment 9	Write Assembly Language program in 8086 to check whether inputting the number is greater,smaller or equal to the 5.
Date: 01/11/2018	

Program: -

```
       Name"cmpwithfive”
       org 100h
       jmp start
       msg db "Enter a  number or any other char to exicute:$"
          equal_5 db "is five!",0dh,0ah,"$"
          below_5 db "is below five ",0dh,0ah,"$"
          above_5 db "is above five ",0dh,0ah,"$"
   start:
   game:  mov dx,offset msg
          mov ah,09h
          int 21h
          mov ah,01h
          int 21h
          cmp al,'0'
          jb stop
          cmp al,'9'
          ja stop
          cmp al,'5'
          jb below
          ja above
          mov dx,offset equal_5
          jmp print
          below:mov dx,offset below_5
          jmp print
   above: mov dx,offset above_5
   print: mov ah,09h
          int 21h
          jmp game
   stop:  ret
```

Output: -



```
Enter a  number or any other char to exit:5is five!
Enter a  number or any other char to exit:6is above five
Enter a  number or any other char to exit:4is below five
Enter a  number or any other char to exit:_
```

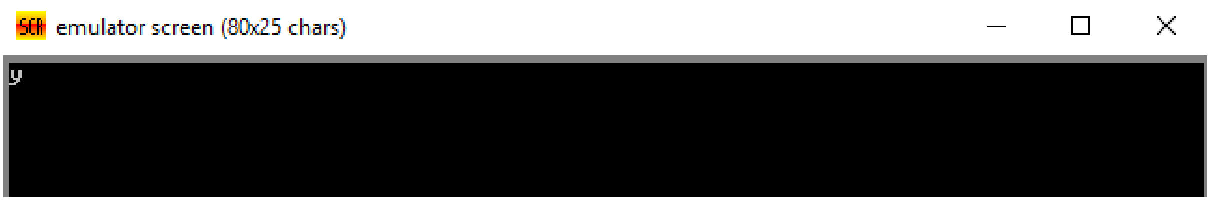
Result: - The program is executed successfully.

Experiment 10	Write Assembly Language program in 8086 for comparison of two string.
Date: 01/11/2018	

Program: -

```
      name "strcmp"
      org 100h
      jmp start

  x1:    str1 db 'test string'
         str2 db 'test string'
         size=($-x1)/2

 start:  cld
         mov ax,cs
         mov ds,ax
         mov es,ax
         lea SI,str1
          lea DI,str2
          mov cx,size

          repe cmpsb
          jnz not_equal
          mov al,'y'
          mov ah,0eh
          int 10h
          jmp exit_here

 not_equal:  mov al,'n'
             mov ah,0eh
             int 10h

 exit_here:   mov ah,0h
              int 16h
           ret
```

Output: -



```
y
```

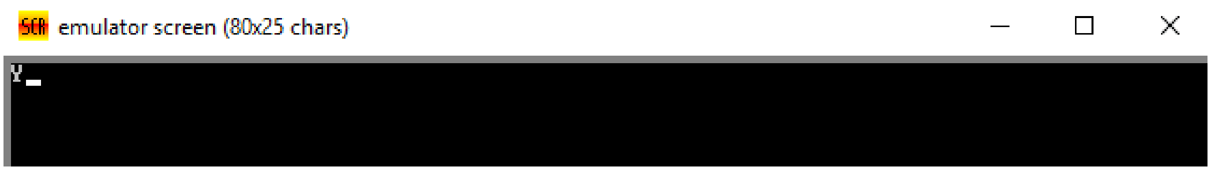
Result: **-**_The program is executed successfully.

Experiment 11	*Write Assembly Language program in 8086 for comparison of two string word.*
Date: 01/11/2018	

Program: -

```
            NAME "COMPSW"
             ORG 100H

            X: DATA1 DW 1234H,5678H,9012H,3456H
               DATA2 DW 1234H,5678H,9012H,3456H
               SIZE=($-X)/4

             CLD
            MOV AX,CS
            MOV DS,AX
            MOV ES,AX
            LEA SI,DATA1
            LEA DI,DATA2
            MOV CX,SIZE
            REPE CMPSW
            JNZ NOT_EQUAL
            MOV AL,'Y'
            MOV AH,0EH
            INT 10H
            JMP EXIT_HERE

    NOT_EQUAL:  MOV AL,'N'
                MOV AH,0EH
                INT 10H
    EXIT_HERE:  MOV AH,0H
                INT 16H
            RET
```

Output: -


emulator screen (80x25 chars)

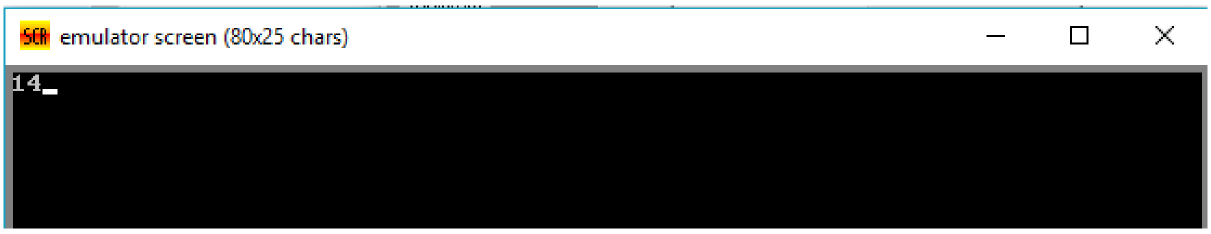
```
Y_
```

Experiment 12	*Write Assembly Language program in 8086 to adding two number using XOR(AAA).*
Date: 01/11/2018	

Program: -

```
NAME "AAA"
ORG 100H
CLD
MOV AH,09H
MOV AL,05H
ADD AL,AH
XOR AH,AH
AAA
MOV DX,AX
MOV AH,0EH
OR DH,30H
MOV AL,DH
INT 10H
OR DL,30H
MOV AL,DL
INT 10H
MOV AH,0
INT 16H
RET
```

Output: -



```
14_
```

Result:- The program was executed successfully.

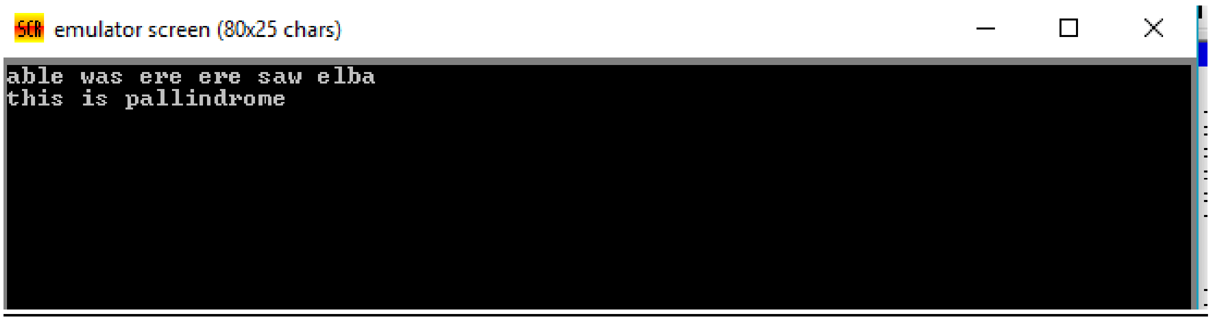
Experiment 13	Write Assembly Language program in 8086 for checking a string for a palindrome.
Date:	

Program: -

```
   name "pallindrome"
  org 100h
  jmp start
  msg1 db "this is pallindrome $"
  msg2 db "this is not pallindrome $"
  m1: s db "able was ere ere sawa elba"
   s_size=$-m1
   db 0dh,0ah,'$'

start: mov ah,09h
      mov dx,offset s
      int 21h
      lea di,s
      mov si,di
      add si,s_size
      dec si
      mov cx,s_size
      cmp cx,01h
      je  is_pallindrome
      shr cx,01

next_char:  mov al,[di]
          mov bl,[si]
          cmp al,bl
          jne not_pallindrome
          inc di
          dec si
          loop next_char

is_pallindrome: mov ah,09h
```

```
            mov dx,offset msg1
            int 21h
            jmp stop

not_pallindrome:
                              mov ah,09h
                              mov dx,offset msg2
                              int 21h
                              jmp stop

       stop:          mov ah,00h
                      int 16h

                    ret
```

<u>Output: -</u>



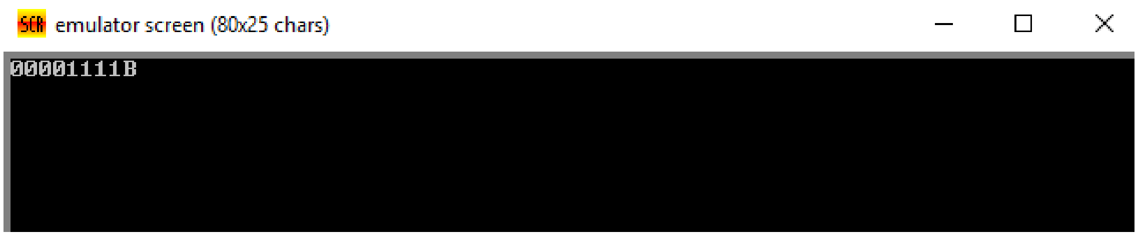
```
able was ere ere saw elba
this is pallindrome
```

Experiment 4	*Write Assembly Language program in 8086 for calculation of the sum of a vector.*
Date: 01/11/2018	

Program:-

```
name "cal_sum"
        org 100H
        jmp start
        vector db 5,4,3,2,1
start:   mov cx,05H
         mov al,00H
         mov bx,00H
 next:  add al, vector [bx]
        inc bx
        loop next
        mov bl,al
        mov cx,08H

print: mov ah,02H
        mov dl,'0'
        test bl,10000000B
        jz zero
        mov dl,'1'

zero:   int 21H
        shl bl,01H
        loop print
        mov dl,'B'
        int 21H

        mov al,0
        int 16H
```

Output:-



```
00001111B
```