Lab program -1
Binary Search program.

```
.model small
;MACRO TO DISPLAY THE MESSAGE....
DISPLAY  MACRO MSG
         LEA DX, MSG
         MOV AH, 09H
         INT 21H
ENDM

.DATA
LIST DB  01H, 05H, 07H, 10H, 12H, 14H.
NUMBER   EQU ($-LIST)
KEY DB   03H
MSG1 DB  0DH, 0AH "Element found in the list
```

```
.CODE
START: MOV  AX, @DATA
       MOV  DS, AX
       MOV  CH, NUMBER-1
       MOV  CL, 00H.
AGAIN: MOV  SI, OFFSET LIST
       XOR  AX, AX
       CMP  CL, CH
       JE   NEXT
       JNC  FAILED.
NEXT:  MOV  AL, CL
       ADD  AL, CH
       SHR  AL, 01H
       MOV  BL, AL
       XOR  AH, AH
```

```
        MOV BP, AX
        MOV AL, DS; [BP][SI]
        CMP AL, KEY
        JE SUCCESS
        JC INCLOW
        MOV CH, BL
        DEC CH
        JMP AGAIN
INCLOW: MOV CL, BL
        INC CL
        JMP AGAIN
SUCCESS: DISPLAY MSG1
        JMP FINAL
FAILED : DISPLAY MSG2.
FINAL  : MOV AH, 4CH
        INT 21H
END START
```

TWISTER

Date Page

Lab program-2
Bubble sort

```
.MODEL SMALL
DISPLAY MACRO MSG
  LEA DX, MSG
  MOV AH, 09H
  INT 21H
ENDM
.DATA
LIST DB 02H, 01H, 34H, 0F4H, 09H, 05H
NUMBER EQU $-LIST
MSG1 DB 0DH,0AH, "1 >> SORT In Ascending order$"
MSG2 DB 0DH,0AH, "2 >> Sort in descending order$"
MSG3 DB 0DH,0AH, "3 >> Exit $"
MSG4 DB 0DH,0AH, "Enter your choice: $"
MSG5 DB 0DH,0AH, "Invalid
     choice entered... $"
.CODE
START: MOV AX, @DATA
  MOV DS, AX
LEA SI, LIST
  MOV CH, NUMBER-1 ; CL STORES THE NUMBER
                     OF ELEMENTS IN LIST
DISPLAY MSG1  ; DISPLAY THE MENU....
DISPLAY MSG2
DISPLAY MSG3
DISPLAY MSG4
MOV AH, 01H
INT 21H
SUB AL, 30H
CMP AL, 01H  ; INPUT=1? SORT IN ASCENDING
                ORDER
```

```
JE ASCSORT
CMP AL,02H ;INPUT=2? SORT IN DESENDIN
                                ORDER

JE DESSORT
CMP AL,03H ;INPUT=3? EXIT
JE FINAL
DISPLAY MSG5
JMP FINAL
ASCSORT: MOV BL,00H
AGAIN : MOV SI,OFFSET LIST
 MOV CL,00H ;JVALUE
 MOV BH,CH
 SUB BH,BL ;N-1-i
NPASS: CMP CL,BH
 JNC NEXT
 MOV AL[SI]
 MOV BP,01H
 CMP AL,DS:[BP][SI]
 JC NOPE
 XCHG AL,[SI+1]
 XCHG [SI],AL
 NOPE: INC CL
 INC SI
 JMP NPASS
NEXT : INC BL
 CMP BL,CH
 JC AGAIN
 JMP FINAL

DESSORT : MOV BL,00H
AGAINI ; MOV SI ,OFFSET LIST
MOV ~~BH~~, ~~CH~~ CL,00H ; JValue
```

```
MOV BH,CH
SUB BH,BL      ; N-1-i
NPASS1 :CMP CL BH
JNC NEXT
MOV AL, [SI]
MOV BP ,01H
CMP AL ,DS :[BP][SI]
JNC _NOPE1
XCHG AL, [SI+1]
XCHG [SI], AL
_NOPE1 : INC CL
 INC SI
 JMP NPASS1
 NEXT1 : INCBL
 CMP BL ,CH
  JC AGAIN1
FINAL : MOV AH, 4CH
 INT 21H
END START
```

Lab program 2

Display the ASCII equivalent

```
.model small
.data
msg db 0dh, 0ah "Enter alphanumeric
                 character $"
res db 02 dup(0)
.code
mov ax, @data
mov ds, ax
lea dx, msg1
call disp
mov ah, 01h
int 21h
mov bl, al
mov cl, 4
shr al, cl
cmp al, 0ah
jc digit
digit1: add bl, 30h
        mov res+1, bl

mov ah, 00h } For displaying
mov al, 03h } Text MODE
int 10h

mov ah, 02h → SET The Cursor Position
mov bh, 00h → PAGE Number
mov dh, 0ch → ROW
mov dl, 28h → Column VAL.
int 10h.
```

```
mov rest+2, '$'
lea dx, res
call disp.
mov ah, 4ch
int 21 h.

disp proc near
mov ah, 09h.
int 21h
ret
disp endp.
end.
```

Lab Program - 3
Palindrome

```
.Model small
display macro msg
    lea dx, msg
    mov ah, 09h
    int 21h.

endm
.data
msg1 db 0DH, 0AH. "Enter String: $"
msg2 db 0DH, 0AH "Reverse string: $"
msg3 db 0DH, 0AH, "Input string is palindrome$"
msg4 db 0DH, 0AH, "Input string is not a palindrome string: $"
Rstring db 80H dup (?)
RRstring db 80H dup (?)
.code
start : mov ax, @data.
        mov ds, ax
        display msg1
        ; take the String from Keyboard charactery By character.
        Mov si, offset string
        XOR cl, cl
Again : Mov ah, 01h
        Int 21h.
        cmp al, 0DH
        JE NEXT
        MOV [SI], AL.
```

```
        INC SI
        INC CL
        JMP AGAIN
NEXT:   MOV[SI],BYTE PTR '$' ;STRING INPUT OVER....
        DEC SI
        MOV CH,CL ;REVERSE THE STRING AND STORE IN RSTRING
        MOV DI,OFFSET RSTRING
BACK:   MOV AL,[SI]
        MOV [DI],AL
        DEC SI
        INC DI
        DEC CH
        JNZ BACK
        MOV [DI],BYTE PTR '$'
        DISPLAY MSG2
        DISPLAY RSTRING
        MOV SI,OFFSET STRING
        MOV DI,OFFSET RSTRING
AG:     MOV AL,[SI]
        CMP AL,[DI]
        JNE FAIL
        INC DI
        DEC CX
        JZ SUCCESS
        JMP AG
FAIL:   DISPLAY MSG4
        JMP FINAL
SUCCESS: DISPLAY MSG3
FINAL:  MOV AH,4CH
        INT 21H
END
```

Lab program - 4
Comparison if two strings

```
.MODEL SMALL
DISPLAY MACRO MSG
        LEA DX, MSG
        MOV AH, 09H
        INT 21H
ENDM
.DATA
MSG1 DB 0DH,0AH, "Enter first string: $"
MSG2 DB 0DH,0AH, "Enter second string: $"
MSG3 DB 0DH,0AH, "Enter length of first
                  string: $"
MSG4 DB 0DH,0AH, "Enter length of
                  second string: $"
MSG5 DB 0DH,0AH, "--- Strings are equal
                  ---$"
MSG6 DB 0DH,0AH, "--- Strings are not
                  equal --- $"
STRING1 DB 80H DUP(?)
STRING2 DB 80H DUP(?)

.CODE
START: MOV AX, @DATA
       MOV DS, AX
       DISPLAY MSG3
       MOV SI, OFFSET STRING1
       CALL READSTR
       MOV BL, CL ; STORE THE
                    LENGTH OF
                    FIRST STRING
```

TWISTER
Date Page

```
DISPLAY MSG2
MOV SI, OFFSET STRING2.
CALL READSER.
PUSH BX
PUSH CX
DISPLAY MSG3
MOV AL, BL
CALL LEN-DIS
DISPLAY MSG4
MOV AL, CL
CALL LEN-DIS
POP CX
POP BX
CMP CL, BL ; COMPARE THE LENGTHS
JNE FAIL ; IF LENGTH ARE EQUAL,
          PROCESS NEXT STATEMENT
MOV SI OFFSET STRING1
MOV DI OFFSET STRING2
CLD
CHK : MOV AL, [SI]
      CMP AL, [DI] ; COMPARE BOTH
                     THE STRINGS
      JNE FAIL
      INC SI
      INC DI
      DEC CL
      JNZ CHK
      DISPLAY MSG5
      JMP FINAL

LEN_DIS PROC NEAR.
        XOR AH, AH
```

```
        ADD AL,00H
        AAM
        ADD AX,3030H
        MOV BH,AL
        MOV DL,AH
        MOV AH 02H
        INT 21H
        MOV DL,BH
        MOV AH,02H
        INT 21H
RET
LEN D'S ENDP
READSTR PROC NEAR
        XOR CL,CL.
BACK:   MOV AH,01H
        INT 21H
        CMP AL,0DH
        JE FINISH
        MOV[SI] AL
        INC SI
        INC CL
        JMP BACK
FINISH: MOV[SI], BYTE PTR '$'
        RET
READSTR ENDP
FAIL:   DISPLAY MSG6
FINAL:  MOV AH,4CH
        INT 21H
END START
```

Lab program-6

NCR

```
.model small
.data
n   dw 4
r   dw 2
nor dw 0

.code
mov ax, @data
mov ds, ax

mov ax,n
mov bx, r
call disp
call ncrpro
jmp final

ncrpro  proc near
        cmp ax, bx ; r=n
        je res1
        cmp bx, 0 ; r=0
        je res1
        cmp bx, 1 ; r=1
        dec ax
        cmp bx, ax
        je incr
        push ax
        push bx
        call ncrpro
pop bx
```

```
pop ax
dec bx
push ax
push bx
call norpro
pop bx
pop ax
ret

res1: inc ncr.
ret
incr: inc ncr.
ress: add ncr, ax   ; 1+2 3+3 = 6.
ret
norpro endp

disp proc near
  mov bx, ncr
  add bx, 3030h.
  mov dl, bh.
  mov ah, 02h.
  int 21h.
  mov dl, bl
  mov ah, 02h.
  int 21h.
  ret
  disp endp.

final. mov ah, 4ch.
    int 21h.
    end.
```

TWISTER

Date Page

Lab program 5

Display the system time

```
.model small
.code
mov ah, 2ch.
int 21h.
mov ch
mov al, ch
aam
mov bx, ax
call disp
mov dl, ':'
mov ah, 02h.
int 21h.

mov al, cl
aam
mov bx, ax
call disp
mov dl, ':'
mov ah, 02h.
int 21h.

mov al, dh.
aam
mov bx, ax
call disp.

mov ah, 4ch
int 21h.
```

```
disp proc near
mov dl, 6h.
add dl, 30h.
mov ah, 02h.
int 21h.
mov dl, bl
add dl, 30h.
mov ah, 02h.
int 21h.
iret
disp endp.
end
```

Lab program - 8
0-99 binary count.

```
.model small
.code
mov cl, 00
mov ah, 00h
mov al, 03h
int 10h

back : mov bh, 00h
       mov dh, 00h
       mov dl, 00h
       mov ah, 02h
       int 10h

       mov al, cl
       add al, 00h
       aam
       add ax, 3030h
       mov ch, al
       mov dl, ah
       mov ah, 02h
       int 21h
       mov dl, ch
       mov ah, 02h
       int 21h
       Call delay
       inc cl
       xor ax, ax
       cmp cl, 100d
       jne back
```

```
    je last.

delay proc near.
push ax
push bx
push cx
mov cx, 0affh.
ag: mov bx, 0fffh.
ag1: nop
dec bx
jnz ag1.
dec cx
jnz ag
pop cx
pop bx
pop ax
ret
delay endp.
last: mov ah, 4ch.
int 21h.
end.
```

Lab program - 9.
Cursor location

```
.model small
disp macro msg
lea dx, msg
mov ah, 09h
int 21h
endm
.data
row db 02 dup(0)
col db 02 dup(0)
msg1 db 0dh, 0ah, "Enter X-co-ordinate: $"
msg2 db 0dh, 0ah, "Enter Y-co-ordinate: $"
msg3 db 0dh, 0ah, "Cursor displayed at
.code
mov ax, @data
mov ds, ax
disp msg1
mov si, offset row
call read
disp msg2
mov si, offset col
call read
mov si, offset row
mov dh, [si]
inc si
mov al, [si]
sub ax, 3030h
add
mov dh, al
mov si, offset col.
```

```
mov ah, [si].
inc si
mov al, [si].
Sub ax, 3030h.
aad
mov al, al.
mov ah, 00
mov al, 03h.
int 10h.
mov ah, 02h.
int 10h.
jmp final.

read proc near.
mov cx, 02h.

back: mov ah, 01h.
int 21h.
mov [si], al.
inc si
dec cx
jnz back.
ret
read endp

final: mov ah, 01h.
int 21h.
mov ah, 4ch
int 21h.
end.
```

Lab program - 10.

File creation and deletion.

```
.model small
display macro msg.
lea dx, msg
mov ah, 09h.
int 21h.
endm

.data
msg1 db 0dh, 0ah, "Enter the filename for creation: $"
mg2 db 0dh, 0ah, "File created Successfully: $"
mgsg3 db 0dh, 0ah, "File creation unsuccessful: $"
msg4 db 0dh, 0ah, "Enter the filename for deletion: $"
mg5 db 0dh, 0ah, "File deleted Successfully: $"
mgsg6 db 0dh, 0ah "File deletion failed: $"
fname1 db 10 dup(0)
fname2 db 10 dup(0)

.code
mov ax, @data
mov ds, ax
display msg1
mov si, 00
back1: mov ah, 01h.
```

```
int 21h.
cmp al, 0dh.
je next1
mov frame1[si], al
inc si
jmp back1

next1: mov frame1[si], '$'
       lea dx, frame1
       mov cx, 00
       mov ah, 3ch.
       int 21h.
       jc cfail.
       display msg2
       jmp del.

cfail: display msg3.
del : display msg4.
      mov si, 00

back2: mov ah, 01h.
       int 21h.
       cmp al, 0dh.
       je next2.
       mov frame2[si], al
       inc si
       jmp back2

next2: mov frame2[si], '$'.
       lea dx, frame2
       mov ah, 4fh.
       int 21h.
```

```
jc dfail
display msg5
jmp final

dfail : display msg6

final : mov ah, 4ch
        int 21h

end
```