Bubble sort

code:

```
.model small
.data
n   db  5
a   db  05,07,04,03,06
.code
      MOV  ax,@data
      MOV  DS,ax
      MOV  cl,n
       dec  cl
outloop : MOV ch,cl
          MOV SI,00h
inloop :  MOV al,a[SI]        -> zero, not '0'
          inc  SI
          CMP  al,a[SI]
          JC   NOexch         if al < a[SI] CF = 1
                              (CF set to 1)
          xchg al, a[SI]      if al > a[SI]
          MOV  a[SI-1], al    CF set to zero.
noexch :  DEC  ch
          JNZ  inloop
          DEC  cl
                  -> PTO
```

JNZ outloop
MOV ah, 4ch
INT 21h

~~Menu driven prog for bubble sort~~

2-10-20

Lab program WAP to find ASCII value of

.model small
.data
msg1 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
ADD AL, 07h

digit: add al, 30h
mov res, al
and bl, 0fh
cmp bl, 0ah
jc digit1
add bl, 07h
digit1: add bl, 30h
mov res+1, bl

```
mov ah, 00h
mov al, 03h      ; TEXT MODE
int 10h

mov ah, 02h      ; SET THE CURSOR position
mov bh, 00h      ; page no
mov dh, 0ch      ; ROWS (00 is Top)
mov dl, 28h      ; column val
int 10h

mov res+2, '$'
lea dx, res
call disp
mov ah, 4Ch
int 21h

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

92) WAP to read a string & check if it is palindrome or not.

- MODEL small
- ~~data~~ 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 a
msg4 DB 0DH, 0AH, "NON palindrom $", palindrom
                                    $"
string DB 80h DUP (?)
resstring DB 80h DUP (?)
.code
start : MOV AX, @data
        MOV DS, AX
        Display msg1
        ; Take the string from keyboard
            character by character
A       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 & store it in resstring
        MOV DI, OFFSET resstring
BACK    : MOV AL, [SI]
          MOV [DI], AL
```

dec SI
inc DI
dec CH
JNZ BACK
MOV [DI], byte ptr '$'
display Msg 2
display Restring
MOV SI, offset String
MOV DI, offset Restring

AG: MOV AL, [SI] ; BMSCE madam
CMP AL, [DI] ; ECSMB madam
JNE FAIL ;
inc SI
inc DI
dec CX
JZ success
JMP AG

FAIL: Display msg4
JMP Final
Success: display msg3

Final: MOV AH, 4ch
int 21H

END