Sorting

Expt No:	6	**Name:**	Mahesh Bharadwaj K
Date :	01/10/2020	**Reg No:**	185001089

Aim:

To write and execute 8086 programs for sorting array in ascending and descending order using bubble sort.

Procedure:

- Mount masm folder to a drive on DOSBOX.
- Navigate to mounted drive using 'dir' .
- Save 8086 program with the extension **'.asm'** in the same folder using the command **'edit'**.
- Assemble the **.asm** file using the command **'masm filename.asm'**.
- Link the assmebled **.obj** file using the command **'link filename.obj'**.
- Debug the executable file **.exe** with the **'debug filename.exe'** command.
    i. **U:** To view the un-assembled code.
    ii. **D:** Used as 'D segment:offset' to see the content of memory locations starting from segment:offset address.
    iii. **E:** To change the values in memory.
    iv. **G:** Execute the program using command.
    v. **Q** exits from the debug session.

Algorithm:

1. Ascending Order

* Load effective address of Array into BX register
* initialise SI register to 0h
* BEGIN OUTER LOOP
    - Compare SI and size of array, if SI is not lesser end loop.
    - Initialise DI register to 0h
    - Load size of array into CX register.
    - Subtract value of SI from CX.

- BEGIN INNER LOOP
  - Compare DI and CX, if DI is not lesser, end loop
  - Move [BX + DI] into AL register.
  - Move [BX + DI + 01h] into AH register.
  - Compare AL & AH
  - IF AL > AH, swap AL and AH using XCHG and move values into array
  - Increment DI
- Increment SI

2. Descending Order

- Load effective address of Array into BX register
- initialise SI register to 0h
- BEGIN OUTER LOOP
  - Compare SI and size of array, if SI is not lesser end loop.
  - Initialise DI register to 0h
  - Load size of array into CX register.
  - Subtract value of SI from CX.
  - BEGIN INNER LOOP
    - Compare DI and CX, if DI is not lesser, end loop
    - Move [BX + DI] into AL register.
    - Move [BX + DI + 01h] into AH register.
    - Compare AL & AH
    - IF AL < AH, swap AL and AH using XCHG and move values into array
    - Increment DI
  - Increment SI

1. Bubble sort: Ascending

Program:

Program		Comments
start:	MOV AX,data	Move data segment address contents to AX register
	MOV ds,AX	Move data in AX register to DS register
	LEA BX, arr	Load Effective Address of array into BX
	MOV SI, 0000H	i = 0
outer:	CMP SI, size_arr	CMP i and n - 1
	JNL **stop**	if i >= n-1, exit loop
	MOV DI,0000h	j = 0
	MOV CX, size_arr	load 'n-1' to CX
	SUB CX, SI	CX is now n - i - 1
inner:	CMP DI, CX	Compare j and n - i - 1
	JNL **next**	if j >= n - i - 1 exit inner loop
	MOV AL, [BX + DI]	arr[j] into AL
	MOV AH, [BX + DI + 0001h]	arr[j+1] into AH
	CMP AL, AH	Compare AL & AH
	JB **skip**	if AL < AH , do nothing
	XCHG AH, AL	Swap AL, AH if AL > AH
	MOV [BX + DI], AL	Move updated values into array
	MOV [BX + DI + 0001h], AH	
skip:	INC DI	j = j + 1
	JMP **inner**	Next iteration of j loop
next:	INC SI	i = i + 1
	JMP **outer**	Next iteration of i loop
stop:	MOV ah,4ch	
	INT 21h	Request interrupt routine

Unassembled Code:

```
-U 076C:0100
076C:0100 B86A07        MOV     AX,076A
076C:0103 8ED8          MOV     DS,AX
076C:0105 8D1E0000      LEA     BX,[0000]
076C:0109 BE0000        MOV     SI,0000
076C:010C 3B361000      CMP     SI,[0010]
076C:0110 7D23          JGE     0135
076C:0112 BF0000        MOV     DI,0000
076C:0115 8B0E1000      MOV     CX,[0010]
076C:0119 2BCE          SUB     CX,SI
076C:011B 3BF9          CMP     DI,CX
076C:011D 7D13          JGE     0132
076C:011F 8A01          MOV     AL,[BX+DI]
```

Input and Output:

Figure 1: **Input:** {04h, 08h, 02h, 0Ah, 01h, 03h}, size_arr = 6 - 1 = 5
Output: {01h, 02h, 03h, 04h, 08h, 0Ah}

```
-d 076A:0000
076A:0000  04 08 02 0A 01 03 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  05 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-g

Program terminated normally
-d 076A:0000
076A:0000  01 02 03 04 08 0A 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  05 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

2. Bubble sort: Descending

Program:

Program		Comments
start:	MOV AX,data	Move data segment address contents to AX register
	MOV ds,AX	Move data in AX register to DS register
	LEA BX, arr	Load Effective Address of array into BX
	MOV SI, 0000H	i = 0
outer:	CMP SI, size_arr	CMP i and n - 1
	JNL **stop**	if i >= n-1, exit loop
	MOV DI,0000h	j = 0
	MOV CX, size_arr	load 'n-1' to CX
	SUB CX, SI	CX is now n - i - 1
inner:	CMP DI, CX	Compare j and n - i - 1
	JNL **next**	if j >= n - i - 1 exit inner loop
	MOV AL, [BX + DI]	arr[j] into AL
	MOV AH, [BX + DI + 0001h]	arr[j+1] into AH
	CMP AL, AH	Compare AL & AH
	JA **skip**	if AL > AH , do nothing
	XCHG AH, AL	Swap AL, AH if AL > AH
	MOV [BX + DI], AL	Move updated values into array
	MOV [BX + DI + 0001h], AH	
skip:	INC DI	j = j + 1
	JMP **inner**	Next iteration of j loop
next:	INC SI	i = i + 1
	JMP **outer**	Next iteration of i loop
stop:	MOV ah,4ch	
	INT 21h	Request interrupt routine

Unassembled Code:

```
D:\>debug 6-B.EXE
-u
076C:0100 B86A07        MOV     AX,076A
076C:0103 8ED8          MOV     DS,AX
076C:0105 8D1E0000      LEA     BX,[0000]
076C:0109 BE0000        MOV     SI,0000
076C:010C 3B361000      CMP     SI,[0010]
076C:0110 7D23          JGE     0135
076C:0112 BF0000        MOV     DI,0000
076C:0115 8B0E1000      MOV     CX,[0010]
076C:0119 2BCE          SUB     CX,SI
076C:011B 3BF9          CMP     DI,CX
076C:011D 7D13          JGE     0132
076C:011F 8A01          MOV     AL,[BX+DI]
```

Input and Output:

Figure 2: **Input:** {04h, 08h, 02h, 0Ah, 01h, 03h}, size_arr = 6 - 1 = 5
Output: {0Ah, 08h, 04h, 03h, 02h, 01h}

```
-d 076A:0000
076A:0000  04 08 02 0A 01 03 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  05 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-g

Program terminated normally
-d 076A:0000
076A:0000  0A 08 04 03 02 01 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  05 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

Result:

8086 ASL programs for bubble sort in ascending and descending order have been executed successfully using MS - DOSBox.