**SSN College of Engineering**

**Department of Computer Science and Engineering**

**UCS1512 – Microprocessors Lab**

**EX:06 - Sorting**

**Exp No: 06 Name : Kshitij Sharma**

**Date: 01/09/2020 Reg No: 185001080**

# 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.
  1. **U:** To view the un-assembled code.
  2. **D:** Used as ‘D segment:offset’ to see the content of memory locations starting from segment:offset address.
  3. **E:** To change the values in memory.
  4. **G:** Execute the program using command.
  5. **Q** exits from the debug session.

# Algorithm:

## 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

## Descending Order

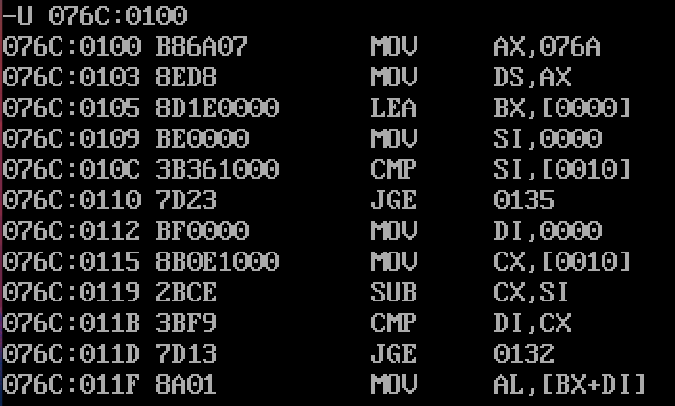
* + Load effective address of Array into BX register
  + initialise SI register to 0h
  + BEGIN OUTER LOOP
    - Compare SI and size of array, if SI is not lesser end loop.
    - Initialise DI register to 0h
    - Load size of array into CX register.
    - Subtract value of SI from CX.
    - BEGIN INNER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

# 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 |

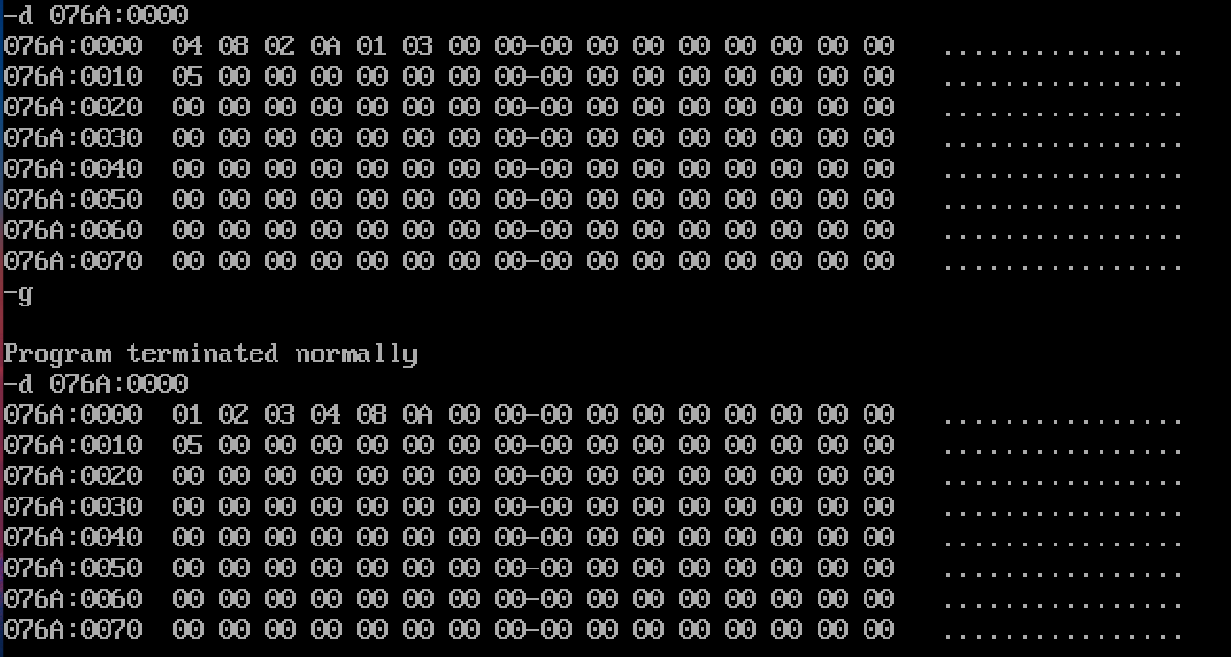
**Unassembled Code:**



**Input and Output:**

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

**Output:** {01h, 02h, 03h, 04h, 08h, 0Ah}

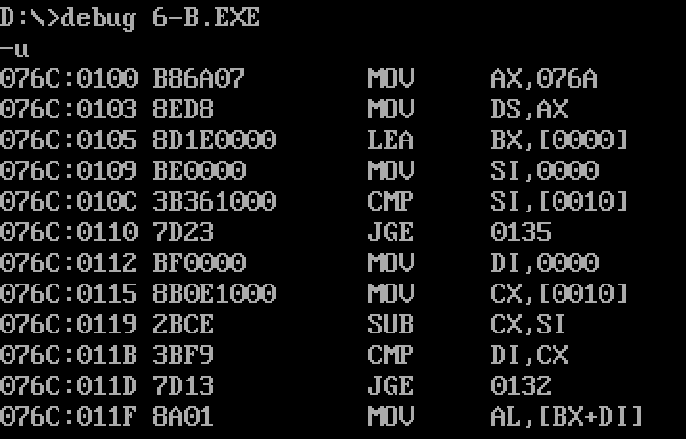


# Bubble sort: Descending

## 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 |

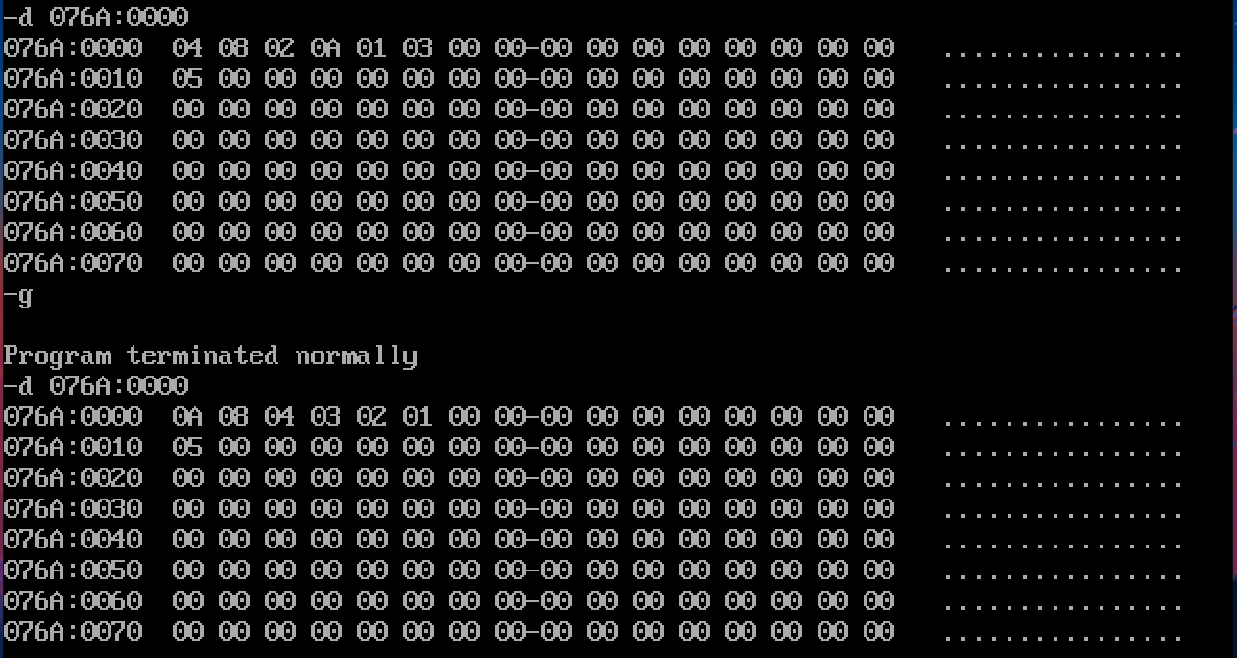
**Unassembled Code:**



**Input and Output:**

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

**Output:** {0Ah, 08h, 04h, 03h, 02h, 01h}



# Result:

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