SSN College of Engineering
Department of Computer Science and Engineering
UCS1512 – Microprocessors Lab
Sorting

Name: Prasanna Kumaran D
Registration Number: 185001110
Semester: V

30 September 2020

1 AIM:

To write and execute 8086 programs for sorting an array in ascending and descending order.

2 PROCEDURE:

- Write the program in a text editor and save it as a ***.asm*** file under the MASM directory.
- Launch DOSBOX application and mount the MASM folder using the command prompt.
- Use the following syntax for mounting : ' **mount [LOCAL DRIVE] FILEPATH** '. Enter into the local drive('LOCAL DRIVE:').
- The code file can be edited using the command **edit FILENAME.asm**. Save the changes and exit.
- Assemble the code using the command '**masm FILENAME.asm**' to generate the object file. The object file is in the format 'FILENAME.obj'
- Add dynamic libraries using the syntax **'link FILENAME.obj'** to generate the executeable(**.exe**) file.
- Enter the debug mode using debug FILENAME.exe to execute and analuse the memory contents. The various commands used in debug mode are as follows:-
  - U :- Displays unassembled code.
  - D :- Refers to the offset from which contents in the memory are displayed.
  - E :- Change the value in memory.
  - G :- execute the code.
  - Q :- Quit debug mode.

3 Algorithm & Program

INITIALIZATION:

- Declare and initialize the operands and the code and data segments.

3.1 Sorting - Ascending:

To sort an array in ascending order.

1. Initialize data and code segment and variables
2. Move the starting address of data segment to DS
3. Initialize AH to 0 and transfer the contents of count to AL register
4. JUMP CONDITION: HERE:- Move AX to CX register, and starting address of matrix 1 to SI register
5. JUMP CONDITION: HERE:- Move the contents pointed by SI register to BL register
6. Compare value in BL register and value pointed at location SI+1
   - if BL > [Si + 1] then JUMP to NEXT, else
   - swap BL and [SI + 1] and transfer the contents of BL register to [SI]
7. JUMP CONDITION: NEXT:-
   - Increment SI, loop HERE1 until CX becomes 0
   - Decrement AX
8. Terminate the program

3.1.1 Sorting - Ascending: Program

Program	Comments
assume cs:code, ds:data	
data segment count db 08h org 0010h matrix1 db 77h,33h,22h,11h,44h,55h,66h,00h data ends	Initialize data segment and variables End data segment

code segment
org 0100h
start:
mov ax,data
mov ds,ax
mov al,count
mov ah,00h
dec ax
here:
mov cx,ax
mov si, offset matrix1
here1:
mov bl, [si]
cmp bl,[si+1]
jge next
xchg bl,[si+1]
mov [si],bl
next:
inc si
loop here1
dec ax
jnz here
mov ah,4ch
int 21h
 	
code ends
end start
	Initialize code segment
	
	Transfer address of data segment to DS
	Transfer the contents of count to AL register and initialize AH to 0
	
	Decrement AX by 1
	
	Move the contents of the AX register to CX register
	Initialize SI register with the starting address of matrix 1
	
	Transfer the contents of the current memory address pointed by SI register to BL register.
	Compare the successor with the value in the BL register, if BL is greater jump to NEXT, else swap BL with the successor, then move the contents of BL register to the current memory address.
	
	Increment SI by 1 (Move to the next location)
	loop until CX becomes 0 (loop here1)
	Decrement AX by 1
	Jump to HERE and continue the next iteration.
	
	Termination of execution
	End of the code segment
	Terminate program

```
P:\>debug sortdesc.exe
-u
076C:0100 B86A07        MOV     AX,076A
076C:0103 8ED8          MOV     DS,AX
076C:0105 B400          MOV     AH,00
076C:0107 A00000        MOV     AL,[0000]
076C:010A 8A1E0100      MOV     BL,[0001]
076C:010E F6E3          MUL     BL
076C:0110 48            DEC     AX
076C:0111 8BC8          MOV     CX,AX
076C:0113 BE1000        MOV     SI,0010
076C:0116 8A1C          MOV     BL,[SI]
076C:0118 3A5C01        CMP     BL,[SI+01]
076C:011B 7E05          JLE     0122
076C:011D 865C01        XCHG    BL,[SI+01]
-_
```

Figure 1: Sorting (Ascending) - unassembled

```
-d 076a:0000
076A:0000  02 04 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  77 33 22 11 44 55 66 00-00 00 00 00 00 00 00 00   w3".DUf.........
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  02 04 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  00 11 22 33 44 55 66 77-00 00 00 00 00 00 00 00   .."3DUfw........
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   ................
-
```

Figure 2: Sorting (Ascending) - Output

3.2 Sorting (Descending):

To sort an array in descending order.

1. Initialize data and code segment and variables
2. Move the starting address of data segment to DS
3. Initialize AH to 0 and transfer the contents of count to AL register
4. JUMP CONDITION: HERE:- Move AX to CX register, and starting address of matrix 1 to SI register
5. JUMP CONDITION: HERE:- Move the contents pointed by SI register to BL register
6. Compare value in BL register and value pointed at location SI+1
   - if BL $<$ [Si + 1] then JUMP to NEXT, else

- swap BL and [SI + 1] and transfer the contents of BL register to [SI]

7. JUMP CONDITION: NEXT:-
   - Increment SI, loop HERE1 until CX becomes 0
   - Decrement AX
8. Terminate the program

3.2.1 Sorting(Descending): Program

Program	**Comments**
assume cs:code, ds:data	
data segment row db 02h col db 04h org 0010h matrix1 db 77h,33h,22h,11h,44h,55h,66h,00h data ends	Initialize data segment and variables End data segment

code segment
org 0100h
start: mov ax,data
mov ds,ax
mov ah,00h
dec ax
here:
mov cx,ax
mov si, offset matrix1
here1:
mov bl, [si]
cmp bl,[si+1]
jle next
xchg bl,[si+1]
mov [si],bl
next:
inc si
loop here1
dec ax
jnz here
mov ah,4ch
int 21h
code ends
end start
	Initialize code segment
	
	Transfer address of data segment to DS
	Transfer the contents of count to AL register and
	initialize AH to 0
	
	Decrement AX by 1
	
	Move the contents of the AX register to CX register
	Initialize SI register with the starting address of matrix 1
	
	Transfer the contents of the current memory address
	pointed by SI register to BL register.
	Compare the successor with the value in the BL register,
	if BL is lesser jump to NEXT, else swap BL with the
	successor, then move the contents of BL register to the
	current memory address.
	
	Increment SI by 1 (Move to the next location)
	loop until CX becomes 0(loop here1)
	Decrement AX by 1
	Jump to HERE and continue the next iteration.
	
	Termination of execution
	End of the code segment
	Terminate program

```
P:\>debug sortasc.exe
-u
076C:0100 B86A07        MOV     AX,076A
076C:0103 8ED8          MOV     DS,AX
076C:0105 A00000        MOV     AL,[0000]
076C:0108 B400          MOV     AH,00
076C:010A 48            DEC     AX
076C:010B 8BC8          MOV     CX,AX
076C:010D BE1000        MOV     SI,0010
076C:0110 8A1C          MOV     BL,[SI]
076C:0112 3A5C01        CMP     BL,[SI+01]
076C:0115 7D05          JGE     011C
076C:0117 865C01        XCHG    BL,[SI+01]
076C:011A 881C          MOV     [SI],BL
076C:011C 46            INC     SI
076C:011D E2F1          LOOP    0110
076C:011F 48            DEC     AX
-_
```

Figure 3: Sorting (Descending) - unassembled

```
-d 076a:0000
076A:0000  08 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  77 33 22 11 44 55 66 00-00 00 00 00 00 00 00 00   w3".DUf.........
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  08 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  77 66 55 44 33 22 11 00-00 00 00 00 00 00 00 00   wfUD3"..........
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   ................
-
```

Figure 4: Sorting (Descending) - Output

4 RESULT:

Thus, 8086 programs to sort an array in ascending and descending order was implemented.