Exp No: 6

Sorting

Date: 21/9/2020

Name: Gokhulnath T
Register Number: 185001051

Aim: To write assembly program to do following sorting operations:

a) Sorting in ascending order
b) Sorting in descending order

Procedure:

1. Install all the required file for executing MASM programs.(Masm, edit, link, debug etc..).
2. Write the assembly program in any editor before mounting the folder to the MASM.
3. Mount the folder that contains the assembly program with any name such as "d".
   - mount d e:\masm
4. Create the object file of the assembly program using masm.
   - masm 16BITADD.asm
5. Use the link to create the executable file of the object file created from the above step.
   - Link 16BITADD.obj
6. Run the executable file using debug.
   - debug 16BITADD.exe
7. By un-assembling the program you can check the code segment of the program
   - u 076b:0100
8. To check the data memory segment, you can use the memory option to view the data stored.
   - d 076a:0000
9. To enter your own values, you can use the enter option which will prompt for new values.
   - e 076a:0000
10. To execute the program, you can use go option
    - G
11. After successful execution and termination of the program, you can check the result by checking the data memory segment
    - d 076a:0000
12. The result can be viewed in the respective address mentioned in the program.

6 a) Sorting in ascending order

Algorithm:

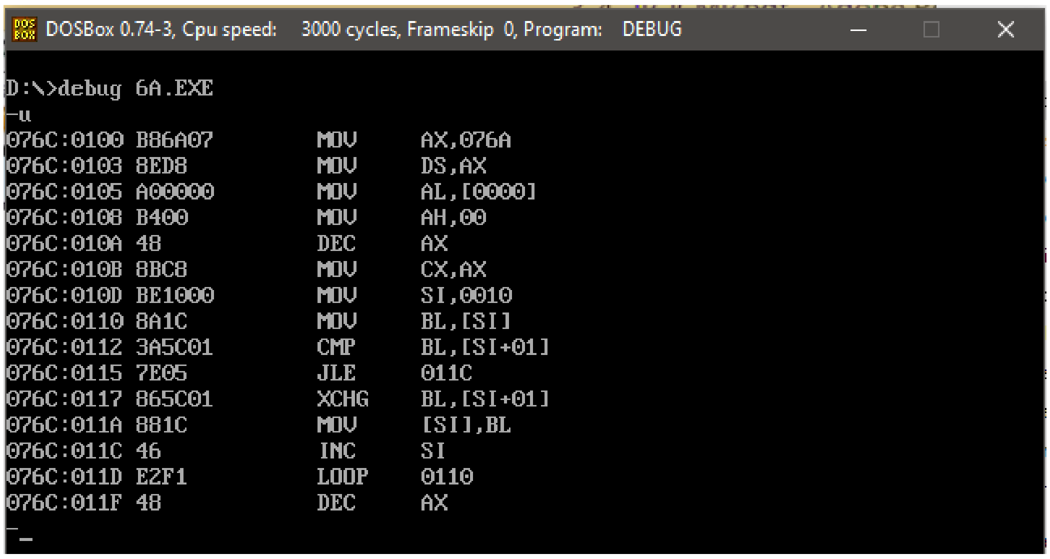
a) Assign data to ax register
b) Load contents of memory location ax in register ds
c) Load contents of memory location count in register cl
d) Load content 00h to register ah
e) Decrement ax
f) Here loop starts
g) Load contents of memory location ax in register cx
h) Load contents of memory location offset list in register si
i) Here1 loop starts
j) Load contents of memory location [si] in register bl
k) Compare bl and [si+1]
l) Jump to next loop if lesser or equal to, else continue
m) Exchange bl and [si+1]
n) Load contents of memory location bl in register [si]
o) Next loop starts
p) Increment si
q) Loop to here1
r) Decrement ax
s) If zero flag cleared jump to here else continue
t) Load content 4ch termination code to ah register
u) Stops execution of the program

Program:

Program	**Comments**
assume cs:code,ds:data,es:extra	Initializing the code, data and extra segments to assembler
data segment	Data segment
count db 08h	count is declared and initialized to 02h
org 0010h	list data segment starting from address range 0010h
list db 55h,66h,00h,77h,33h,22h,11h,44h	list is declared and initialized to 55h,66h,00h,77h,33h,22h,11h,44h
data ends	
code segment	Code segment
org 0100h	assemble the code starting from address range 0100h
start: mov ax,data	Transferring the data from memory location data to ax
mov ds,ax	Transferring the data from memory location ax to ds

mov al,count	Transferring the data from memory location count to al
mov ah,00h	Transferring the data 00h to ah
dec ax	Decrement ax
Here: mov cx,ax	Here loop Transferring the data from memory location ax to cx
mov si, offset list	Transferring the data from memory location offset list to si
here1: mov bl, [si]	Here1 loop Transferring the data from memory location [si] to bl
cmp bl,[si+1]	Compare bl and [si+1]
jle next	Jump to next loop if lesser or equal to, else continue
xchg bl,[si+1]	Exchange bl and [si+1]
mov [si],bl	Transferring the data from memory location bl to [si]
next: inc si	Next loop Increment si
loop here1	Loop to here1
dec ax	Decrement ax
jnz here	If zero flag cleared jump to here else continue
mov ah,4ch	Transferring the termination code 4ch to ah
int 21h	Termination
code ends	Code ends
end start	

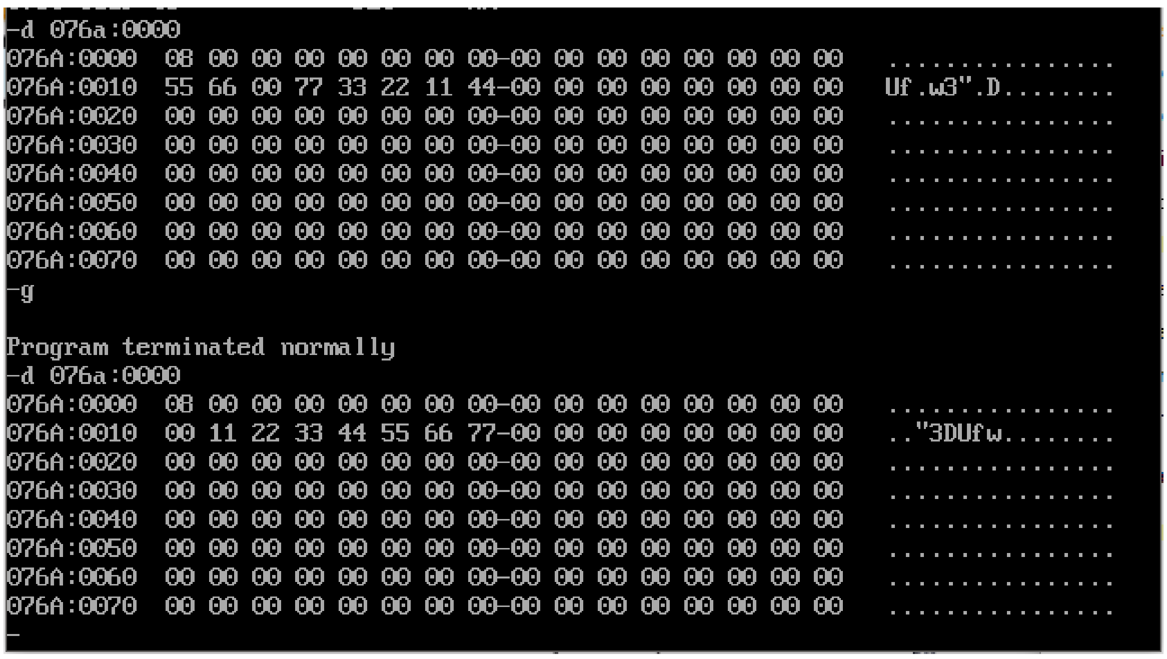
Unassembled code:



DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG

```
D:\>debug 6A.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 7E05          JLE     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
-_
```

Sample Input and output:



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

Result:

Thus, the assembly program for Sorting in ascending order is written and executed.

6 b) Sorting in descending order

Algorithm:

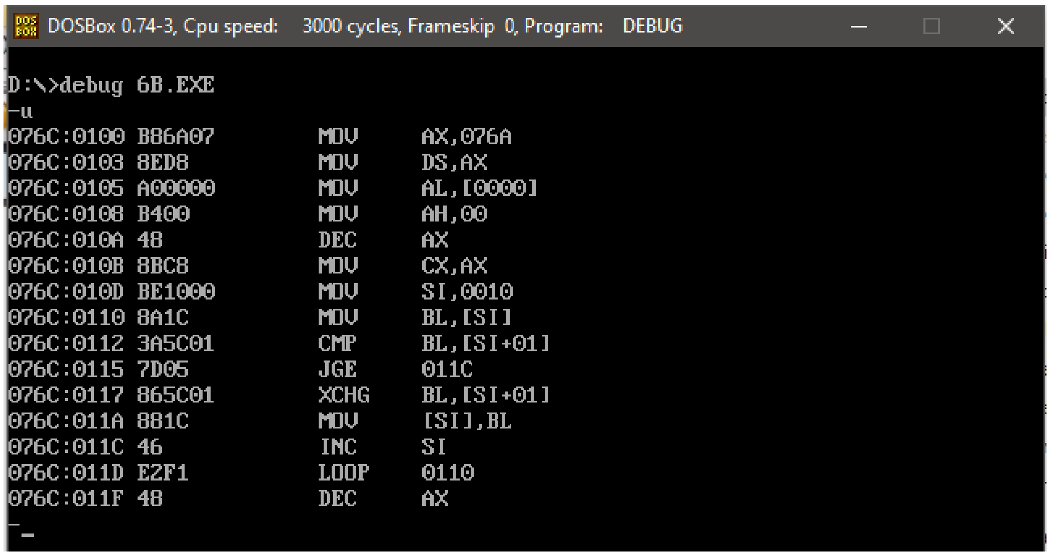
a) Assign data to ax register
b) Load contents of memory location ax in register ds
c) Load contents of memory location count in register cl
d) Load content 00h to register ah
e) Decrement ax
f) Here loop starts
g) Load contents of memory location ax in register cx
h) Load contents of memory location offset list in register si
i) Here1 loop starts
j) Load contents of memory location [si] in register bl
k) Compare bl and [si+1]
l) Jump to next loop if greater or equal to, else continue
m) Exchange bl and [si+1]
n) Load contents of memory location bl in register [si]
o) Next loop starts
p) Increment si
q) Loop to here1
r) Decrement ax
s) If zero flag cleared jump to here else continue
t) Load content 4ch termination code to ah register
u) Stops execution of the program

Program:

Program	**Comments**
assume cs:code,ds:data,es:extra	Initializing the code, data and extra segments to assembler
data segment	Data segment
count db 08h	count is declared and initialized to 02h
org 0010h	list data segment starting from address range 0010h
list db 55h,66h,00h,77h,33h,22h,11h,44h	list is declared and initialized to 55h,66h,00h,77h,33h,22h,11h,44h
data ends	
code segment	Code segment
org 0100h	assemble the code starting from address range 0100h
start: mov ax,data	Transferring the data from memory location data to ax
mov ds,ax	Transferring the data from memory location ax to ds

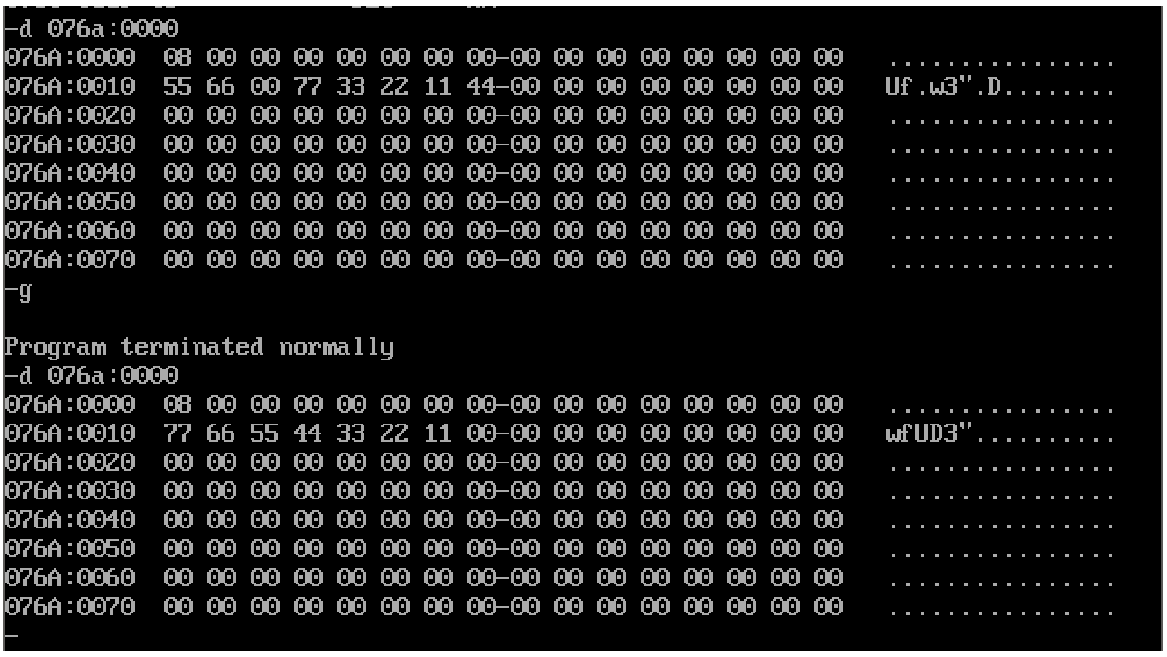
mov al,count	Transferring the data from memory location count to al
mov ah,00h	Transferring the data 00h to ah
dec ax	Decrement ax
Here: mov cx,ax	Here loop Transferring the data from memory location ax to cx
mov si, offset list	Transferring the data from memory location offset list to si
here1: mov bl, [si]	Here1 loop Transferring the data from memory location [si] to bl
cmp bl,[si+1]	Compare bl and [si+1]
jge next	Jump to next loop if greater or equal to, else continue
xchg bl,[si+1]	Exchange bl and [si+1]
mov [si],bl	Transferring the data from memory location bl to [si]
next: inc si	Next loop Increment si
loop here1	Loop to here1
dec ax	Decrement ax
jnz here	If zero flag cleared jump to here else continue
mov ah,4ch	Transferring the termination code 4ch to ah
int 21h	Termination
code ends	Code ends
end start	

Unassembled code:



```
D:\>debug 6B.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
-
```

Sample Input and output:



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

Result:

Thus, the assembly program for Sorting in descending order is written and executed.