

Experiment No. 6: Sorting

Date: 21-09-2020

NAME: Harshini S

REG.NO: 185001058

A. AIM:

Program for sorting in ascending order.

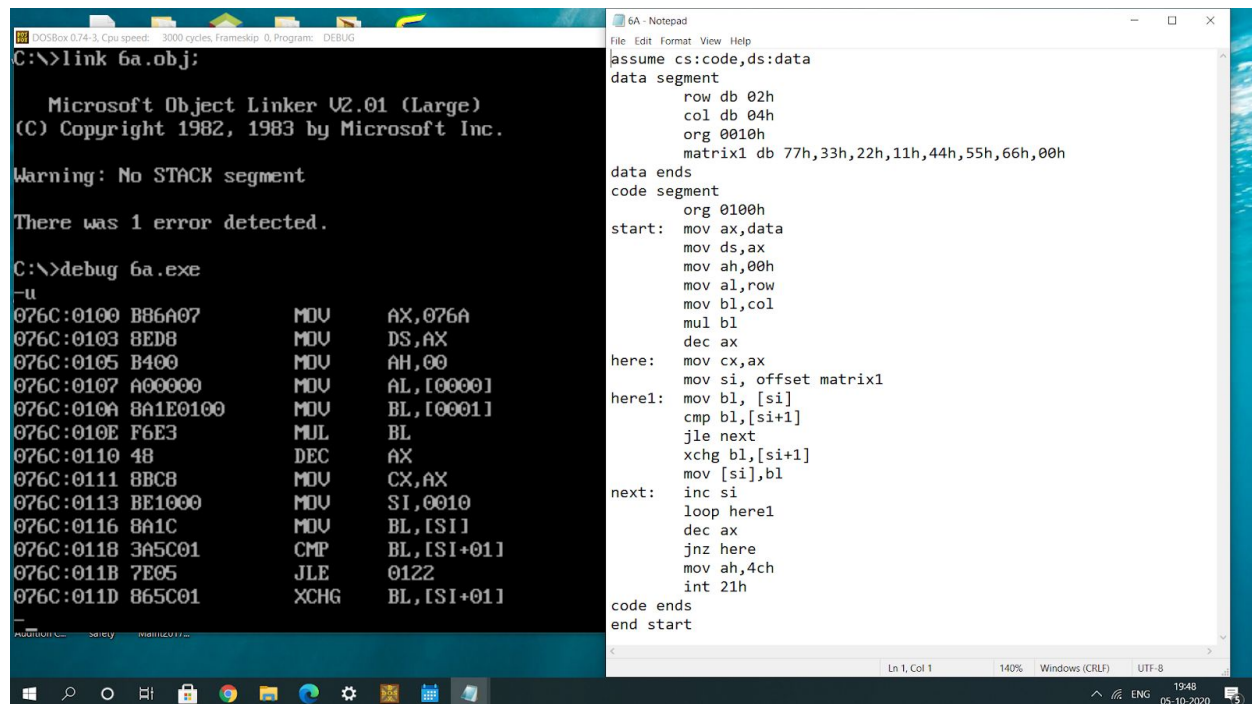
ALGORITHM:

- Initialize the data segment.
- Move data segment address to ds
- Initialize ah with 00h.
- Move row value to al and col value to bl.
- Multiply al with bl.
- Decrement ax.
- Here :
 - Move ax value to cx
 - Load offset of matrix1 to si
- Here1 :
 - Move contents pointed by si to bl
 - Compare contents pointed by si+1 with bl
 - If bl is less than or equal to [si+1] jump to next
 - Exchange values of bl and [si+1]
 - Move bl to matrix1
- next :
 - Increment si
 - Loop here1
 - Decrement ax
 - Jump to here if not equal to zero
 - Terminate the program

PROGRAM:

PROGRAM	COMMENTS
start: mov ax,data mov ds,ax mov ah,00h mov al,row mov bl,col mul bl dec ax	Load data segment to ds Initialise ah with 00h Move row value to ah Move col value to bl Multiply al with bl Decrement ax
Here: mov cx,ax mov si, offset matrix1	Move contents of ax to cx. Move offset of matrix1 to si.
here1: mov bl, [si] cmp bl,[si+1] jle next xchg bl,[si+1] mov [si],bl	Move contents pointed by si to bl Move contents pointed by si+1 to bl If bl is less than or equal to [si+1] jump to next Exchange values of bl and [si+1] Move bl to matrix1
next: inc si loop here1 dec ax jnz here mov ah,4ch int 21h	Increment si Start loop here1 Decrement ax Jump to here if not equal to 0 Terminate the program

UNASSEMBLED CODE:



The screenshot shows two windows. On the left is DOSBox 0.74-3 running the linker 'link 6a.obj'. It displays the 'Microsoft Object Linker V2.01 (Large)' version, copyright 1982, 1983 by Microsoft Inc. A warning states 'No STACK segment' and reports 'There was 1 error detected.' Below this, the debugger 'debug 6a.exe' is run, showing assembly instructions for the 076C segment from offset 0100 to 011D. On the right is a Notepad window titled '6A - Notepad' containing the assembly source code. The code defines a data segment with 'row' (02h), 'col' (04h), and 'matrix1' (77h, 33h, 22h, 11h, 44h, 55h, 66h, 00h). The code segment starts at 0100h, moves 'data' to 'ax', sets 'ds' to 'ax', and initializes 'ah' to 00h. It then enters a loop where it moves 'row' to 'al', 'col' to 'bl', multiplies them, decrements 'ax', and moves 'cx' to 'ax'. It then moves the offset of 'matrix1' to 'si', moves 'bl' to '[si]', compares '[si+1]', and jumps left if less or equal to 'next'. It then swaps '[si+1]' with 'bl', increments 'si', loops back to 'here1', decrements 'ax', jumps right if not zero to 'here', moves 'ah' by 4 bytes, and finally interrupts at 21h.

```
C:\>link 6a.obj:

Microsoft Object Linker V2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

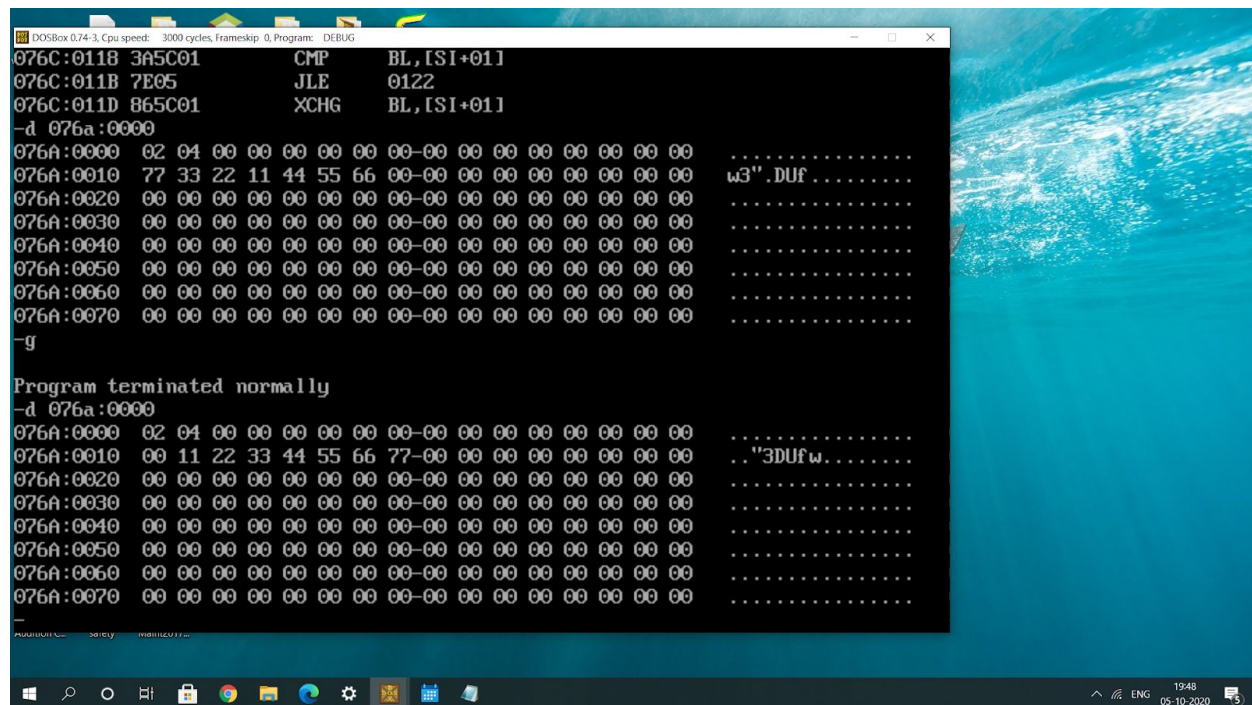
Warning: No STACK segment

There was 1 error detected.

C:\>debug 6a.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]

C:\>
Ln 1, Col 1    140%    Windows (CRLF)    UTF-8
1948
05-10-2020
```

SAMPLE INPUT/OUTPUT:



The screenshot shows DOSBox 0.74-3 in DEBUG mode. It displays the same assembly instructions as the previous image. Below the instructions, it shows a memory dump starting at 076A:0000. The dump shows a sequence of bytes: 02, 04, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00. To the right of the dump, the text 'w3".DUF' is visible. The program terminates normally, and the memory dump is repeated. The DOSBox status bar at the bottom shows '1948' and '05-10-2020'.

```
076C:0118 3A5C01     CMP     BL,[SI+01]
076C:011B 7E05        JLE     0122
076C:011D 865C01     XCHG    BL,[SI+01]
-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 77-00 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 .....
Ln 1, Col 1    140%    Windows (CRLF)    UTF-8
1948
05-10-2020
```

RESULT:

Thus the array has been sorted in ascending order.

B. AIM:

Program for sorting in descending order

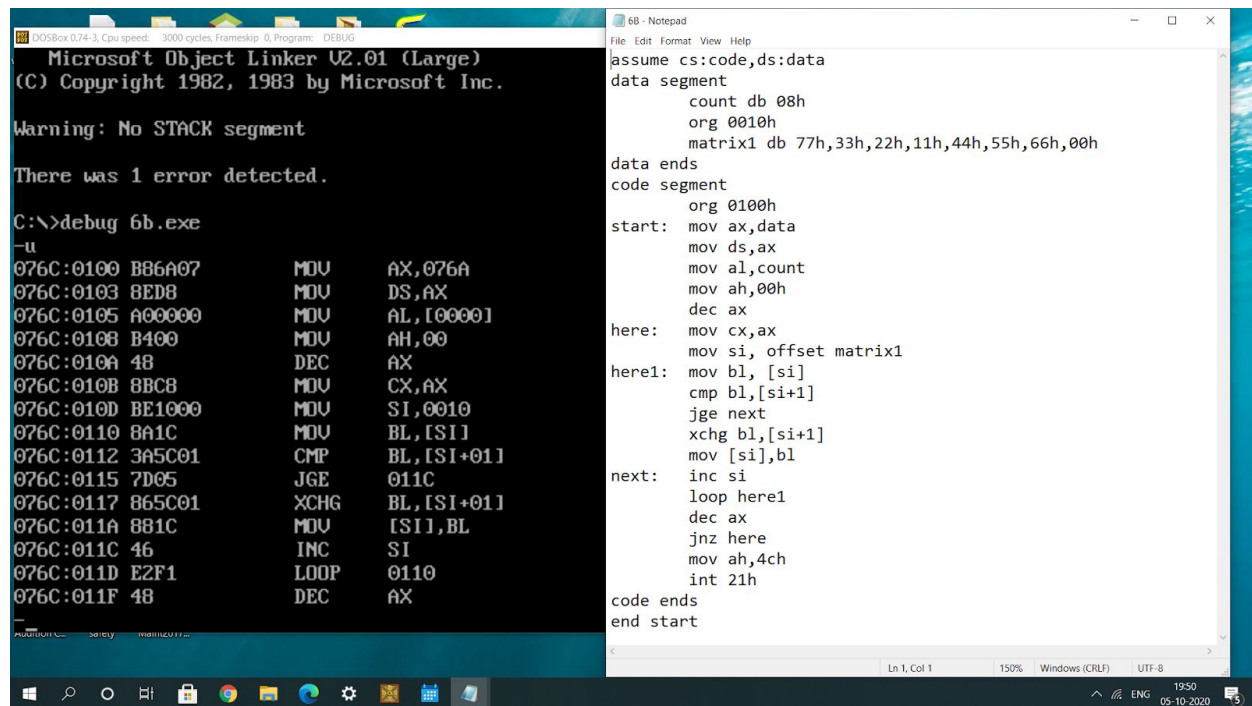
ALGORITHM:

- Initialize the data segment.
- Move data segment address to ds
- Load al with count value
- Initialize ah with 00h.
- Decrement ax.
- Here :
 - Move ax value to cx
 - Load offset of matrix1 to si
- Here1 :
 - Move contents pointed by si to bl
 - Compare contents pointed by si+1 with bl
 - If bl is greater than or equal to [si+1] jump to next
 - Exchange values of bl and [si+1]
 - Move bl to matrix1
- next :
 - Increment si
 - Loop here1
 - Decrement ax
 - Jump to here if not equal to zero
 - Terminate the program

PROGRAM:

PROGRAM	COMMENTS
start: mov ax,data mov ds,ax mov al,count mov ah,00h dec ax	Load data segment to ds Load al with count. Initialise ah with 00h Decrement ax
Here: mov cx,ax mov si, offset matrix1	Move contents of ax to cx. Move offset of matrix1 to si.
here1: mov bl, [si] cmp bl,[si+1] jge next xchg bl,[si+1] mov [si],bl	Move contents pointed by si to bl Move contents pointed by si+1 to bl If bl is greater than or equal to [si+1] jump to next Exchange values of bl and [si+1] Move bl to matrix1
next: inc si loop here1 dec ax jnz here mov ah,4ch int 21h	Increment si Start loop here1 Decrement ax Jump to here if not equal to 0 Terminate the program

UNASSEMBLED CODE:



The screenshot shows two windows. On the left is DOSBox 0.74-3 in DEBUG mode, displaying the assembly of 6b.exe. On the right is a Notepad window showing the source code for 6b.asm.

```
Microsoft Object Linker V2.01 (Large)
(C) Copyright 1982, 1983 by Microsoft Inc.

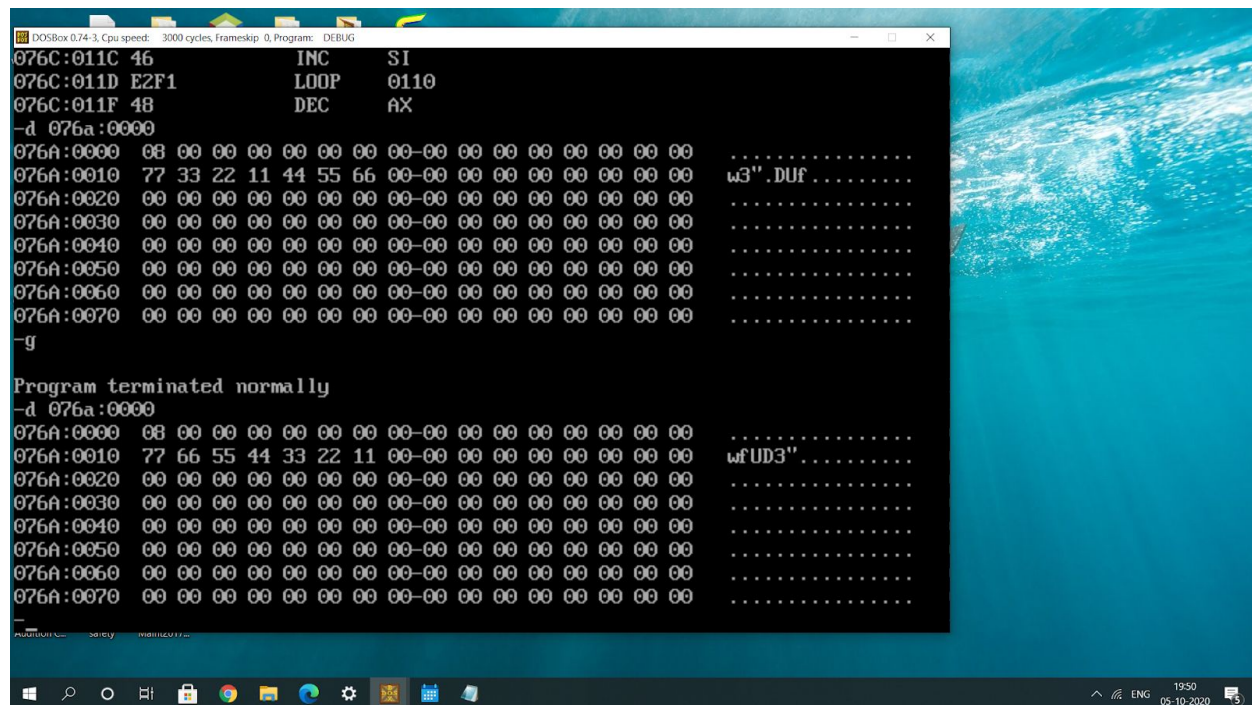
Warning: No STACK segment

There was 1 error detected.

C:\>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 4B          DEC     AX
076C:010B BBC8        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 8B1C        MOV     [SI],BL
076C:011C 46          INC     SI
076C:011D E2F1        LOOP    0110
076C:011F 4B          DEC     AX

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip: 0, Program: 6b.exe
6B - Notepad
File Edit Format View Help
assume cs:code,ds:data
data segment
    count db 08h
    org 0010h
    matrix1 db 77h,33h,22h,11h,44h,55h,66h,00h
data ends
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
```

SAMPLE INPUT/OUTPUT



The screenshot shows DOSBox 0.74-3 in DEBUG mode. It displays the continuation of the assembly from the previous block, followed by a memory dump of the data segment and a message indicating the program terminated normally.

```
076C:011C 46          INC     SI
076C:011D E2F1        LOOP    0110
076C:011F 4B          DEC     AX
-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 .....
```

RESULT:

Thus the array has been sorted in descending order.