Sorting

Expt No:	6	**Name:**	Shivanirudh S G
Date :	09/10/2020	**Reg No:**	185001146

Aim:

To perform sorting operations in 8086.

Ascending Order

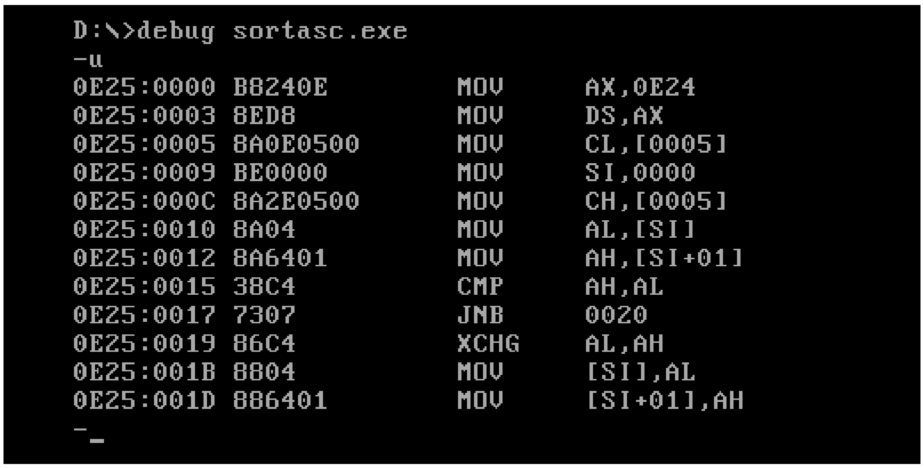
Algorithm:

- Move the data segment to the AX register and then move it to the DS register.
- Move value of count to CL register.
- Move offset of arr into SI register under label OUTER.
- Move value of count to CH register.
- Move value at [SI] to AL register, [SI+1] to AH register, under label INNER.
- Compare AH, AL with CMP AH, AL.
- If CF = 0, jump to label NOSWAP.
- Swap values of AH, AL with XCHG AH, AL
- Move value in AL to [SI] register, AH to [SI+1].
- Increment SI, decrement CH under label NOSWAP.
- Jump to INNER if ZF = 0.
- Decrement CL and jump to OUTER if ZF = 0.

Program:

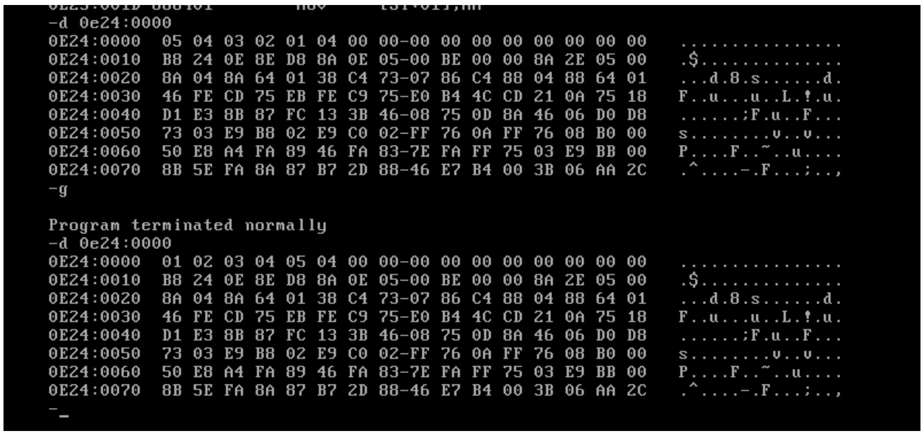
Program	**Comments**
assume cs:code, ds:data	Declare code and data segments
data segment	Start of data segment
arr db 05H, 04H, 03H, 02H, 01H	Define array of values arr
count db 04H	Define byte count with hex value 04
data ends	End of data segment
code segment	Start of code segment
start: mov ax, data	Move data to AX register
mov ds, ax	Move contents of AX register to DS register
mov cl, count	Move value of count to CL register
outer: mov si, offset arr	Move offset of arr to SI register
mov ch, count	Move value of count to CH register
inner: mov al, [si]	Move value at offset in SI to AL register
mov ah, [si+1]	Move value at offset in SI register +1 to AH
cmp ah, al	Compare values in AH, AL registers
jnc noswap	Jump to NOSWAP if CF = 0
xchg al, ah	Swap values in AL, AH registers
mov [si], al	Move value in AL register to offset at [SI]
mov [si+1], ah	Move value in AH register to offset at [SI]+1
noswap: inc si	Increment value of SI
dec ch	Decrement value of CH
jnz inner	Jump to INNER if ZF = 0
dec cl	Decrement value of CL
jnz outer	Jump to OUTER if ZF = 0
mov ah, 4ch	To request interrupt
int 21h	Request interrupt routine
code ends	End of code segment
end start	

Unassembled code:



```
D:\>debug sortasc.exe
-u
0E25:0000 B8240E        MOV     AX,0E24
0E25:0003 8ED8          MOV     DS,AX
0E25:0005 8A0E0500      MOV     CL,[0005]
0E25:0009 BE0000        MOV     SI,0000
0E25:000C 8A2E0500      MOV     CH,[0005]
0E25:0010 8A04          MOV     AL,[SI]
0E25:0012 8A6401        MOV     AH,[SI+01]
0E25:0015 38C4          CMP     AH,AL
0E25:0017 7307          JNB     0020
0E25:0019 86C4          XCHG    AL,AH
0E25:001B 8804          MOV     [SI],AL
0E25:001D 886401        MOV     [SI+01],AH
-_
```

Input and Output:



```
-d 0e24:0000
0E24:0000  05 04 03 02 01 04 00 00-00 00 00 00 00 00 00 00   ................
0E24:0010  B8 24 0E 8E D8 8A 0E 05-00 BE 00 00 8A 2E 05 00   .$..............
0E24:0020  8A 04 8A 64 01 38 C4 73-07 86 C4 88 04 88 64 01   ...d.8.s......d.
0E24:0030  46 FE CD 75 EB FE C9 75-E0 B4 4C CD 21 0A 75 18   F..u...u..L.!.u.
0E24:0040  D1 E3 8B 87 FC 13 3B 46-08 75 0D 8A 46 06 D0 D8   ......;F.u..F...
0E24:0050  73 03 E9 B8 02 E9 C0 02-FF 76 0A FF 76 08 B0 00   s........v..v...
0E24:0060  50 E8 A4 FA 89 46 FA 83-7E FA FF 75 03 E9 BB 00   P....F..~..u....
0E24:0070  8B 5E FA 8A 87 B7 2D 88-46 E7 B4 00 3B 06 AA 2C   .^....-.F...;..,
-g

Program terminated normally
-d 0e24:0000
0E24:0000  01 02 03 04 05 04 00 00-00 00 00 00 00 00 00 00   ................
0E24:0010  B8 24 0E 8E D8 8A 0E 05-00 BE 00 00 8A 2E 05 00   .$..............
0E24:0020  8A 04 8A 64 01 38 C4 73-07 86 C4 88 04 88 64 01   ...d.8.s......d.
0E24:0030  46 FE CD 75 EB FE C9 75-E0 B4 4C CD 21 0A 75 18   F..u...u..L.!.u.
0E24:0040  D1 E3 8B 87 FC 13 3B 46-08 75 0D 8A 46 06 D0 D8   ......;F.u..F...
0E24:0050  73 03 E9 B8 02 E9 C0 02-FF 76 0A FF 76 08 B0 00   s........v..v...
0E24:0060  50 E8 A4 FA 89 46 FA 83-7E FA FF 75 03 E9 BB 00   P....F..~..u....
0E24:0070  8B 5E FA 8A 87 B7 2D 88-46 E7 B4 00 3B 06 AA 2C   .^....-.F...;..,
-_
```

Figure 1: **Input:** 05H, 04H, 03H, 02H, 01H;
Output: 01H, 02H, 03H, 04H, 05H

Descending Order

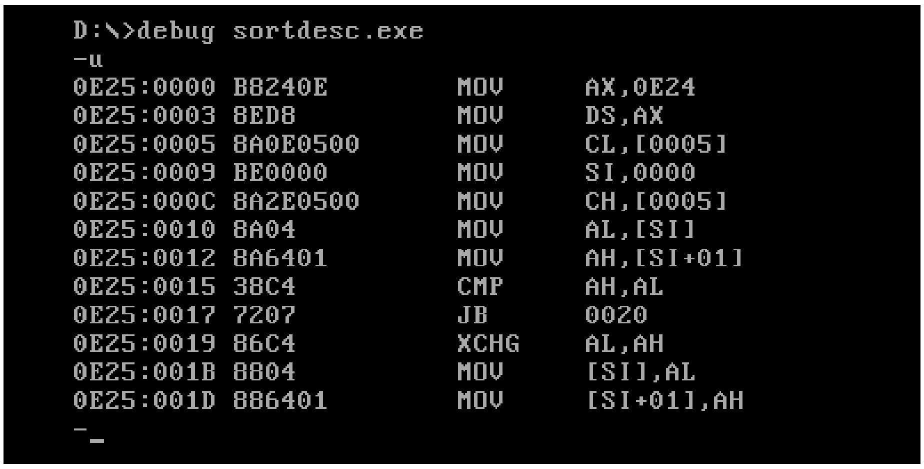
Algorithm:

- Move the data segment to the AX register and then move it to the DS register.
- Move value of count to CL register.
- Move offset of arr into SI register under label OUTER.
- Move value of count to CH register.
- Move value at [SI] to AL register, [SI+1] to AH register, under label INNER.
- Compare AH, AL with CMP AH, AL.
- If CF = 1, jump to label NOSWAP.
- Swap values of AH, AL with XCHG AH, AL
- Move value in AL to [SI] register, AH to [SI+1].
- Increment SI, decrement CH under label NOSWAP.
- Jump to INNER if ZF = 0.
- Decrement CL and jump to OUTER if ZF = 0.

Program:

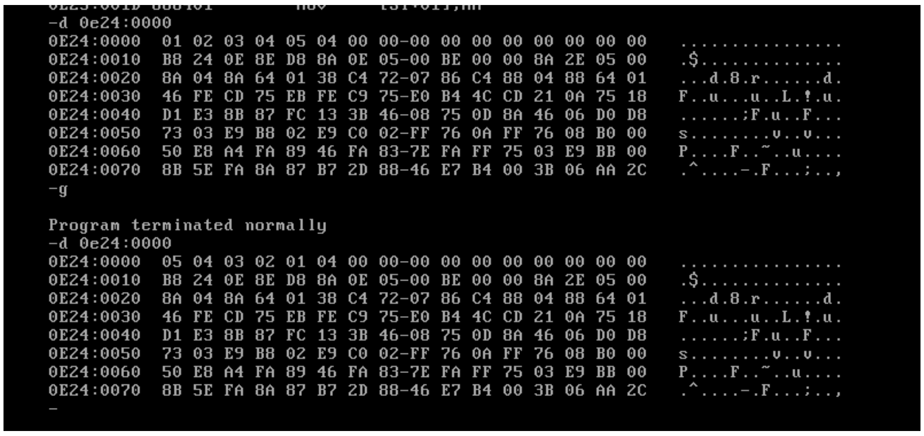
Program	**Comments**
assume cs:code, ds:data	Declare code and data segments
data segment	Start of data segment
arr db 05H, 04H, 03H, 02H, 01H	Define array of values arr
count db 04H	Define byte count with hex value 04
data ends	End of data segment
code segment	Start of code segment
start: mov ax, data	Move data to AX register
mov ds, ax	Move contents of AX register to DS register
mov cl, count	Move value of count to CL register
outer: mov si, offset arr	Move offset of arr to SI register
mov ch, count	Move value of count to CH register
inner: mov al, [si]	Move value at offset in SI to AL register
mov ah, [si+1]	Move value at offset in SI register +1 to AH
cmp ah, al	Compare values in AH, AL registers
jc noswap	Jump to NOSWAP if CF = 1
xchg al, ah	Swap values in AL, AH registers
mov [si], al	Move value in AL register to offset at [SI]
mov [si+1], ah	Move value in AH register to offset at [SI]+1
noswap: inc si	Increment value of SI
dec ch	Decrement value of CH
jnz inner	Jump to INNER if ZF = 0
dec cl	Decrement value of CL
jnz outer	Jump to OUTER if ZF = 0
mov ah, 4ch	To request interrupt
int 21h	Request interrupt routine
code ends	End of code segment
end start	

Unassembled code:



```
D:\>debug sortdesc.exe
-u
0E25:0000 B8240E        MOV     AX,0E24
0E25:0003 8ED8          MOV     DS,AX
0E25:0005 8A0E0500      MOV     CL,[0005]
0E25:0009 BE0000        MOV     SI,0000
0E25:000C 8A2E0500      MOV     CH,[0005]
0E25:0010 8A04          MOV     AL,[SI]
0E25:0012 8A6401        MOV     AH,[SI+01]
0E25:0015 38C4          CMP     AH,AL
0E25:0017 7207          JB      0020
0E25:0019 86C4          XCHG    AL,AH
0E25:001B 8804          MOV     [SI],AL
0E25:001D 886401        MOV     [SI+01],AH
-_
```

Input and Output:



```
-d 0e24:0000
0E24:0000  01 02 03 04 05 04 00 00-00 00 00 00 00 00 00 00   ................
0E24:0010  B8 24 0E 8E D8 8A 0E 05-00 BE 00 00 8A 2E 05 00   .$..............
0E24:0020  8A 04 8A 64 01 38 C4 72-07 86 C4 88 04 88 64 01   ...d.8.r......d.
0E24:0030  46 FE CD 75 EB FE C9 75-E0 B4 4C CD 21 0A 75 18   F..u...u..L.!.u.
0E24:0040  D1 E3 8B 87 FC 13 3B 46-08 75 0D 8A 46 06 D0 D8   ......;F.u..F...
0E24:0050  73 03 E9 B8 02 E9 C0 02-FF 76 0A FF 76 08 B0 00   s........v..v...
0E24:0060  50 E8 A4 FA 89 46 FA 83-7E FA FF 75 03 E9 BB 00   P....F..~..u....
0E24:0070  8B 5E FA 8A 87 B7 2D 88-46 E7 B4 00 3B 06 AA 2C   .^....-.F...;..,
-g

Program terminated normally
-d 0e24:0000
0E24:0000  05 04 03 02 01 04 00 00-00 00 00 00 00 00 00 00   ................
0E24:0010  B8 24 0E 8E D8 8A 0E 05-00 BE 00 00 8A 2E 05 00   .$..............
0E24:0020  8A 04 8A 64 01 38 C4 72-07 86 C4 88 04 88 64 01   ...d.8.r......d.
0E24:0030  46 FE CD 75 EB FE C9 75-E0 B4 4C CD 21 0A 75 18   F..u...u..L.!.u.
0E24:0040  D1 E3 8B 87 FC 13 3B 46-08 75 0D 8A 46 06 D0 D8   ......;F.u..F...
0E24:0050  73 03 E9 B8 02 E9 C0 02-FF 76 0A FF 76 08 B0 00   s........v..v...
0E24:0060  50 E8 A4 FA 89 46 FA 83-7E FA FF 75 03 E9 BB 00   P....F..~..u....
0E24:0070  8B 5E FA 8A 87 B7 2D 88-46 E7 B4 00 3B 06 AA 2C   .^....-.F...;..,
-
```

Figure 2: **Input:** 01H, 02H, 03H, 04H, 05H;
Output: 05H, 04H, 03H, 02H, 01H

Result:

The 8086 programs were written to perform matrix operations, and the results observed.