



Computer Science & Engineering
CSE2006 – Microprocessor and Interfacing

LAB FAT

Submitted to **Prof. SANJAY R**

NAME: PUNIT MIDDHA

REG.NO: 19BCE2060

SLOT: L43+L44

DATE: 08/12/2021

Write the ALP to arrange a given series of hexadecimal bytes in ascending order. Assume the array of decimal numbers as 16, 27, 13, -23, -18, 10

AIM:

Name - Punit Middha

Reg No - 19BCE2060

Slot - L43+L44

Date - 08/12/2021

Faculty - Prof. Sanjay R.

LAB FAT

AIM-

We have to implement ALP program to arrange a given series of hexadecimal bytes in ascending order and we have to assume the array of decimal numbers {16, 27, 13, -23, -18, 10}. And most important we have to ~~as~~ implement the code in



Emu8086 emulator.

ALGORITHM:

ALGORITHM -

Step 1 : Start

Step 2 : Initialize arr[] = {16, 27, 13, -23, -18, 10}

Step 3 : Set temp = 0

Step 4 : Length = 6

Step 5 : Set i = 0. REPEAT Step 6 and step 7
Until i < length

Step 6 : PRINT arr[i]

Step 7 : i = i + 1

Step 8 : SET i = 0. REPEAT Step 9 to Step UNTIL
i < n

Step 9 : set j = i + 1. Repeat Step 10 until
j < length

Step 10 : if (arr[i] > arr[j]) then
temp = arr[i]
arr[i] = arr[j]
arr[j] = temp

Step 11 : j = j + 1

Step 12 : i = i + 1

Step 13 : Set i = 0; REPEAT Step 14 and Step 15
until i < length

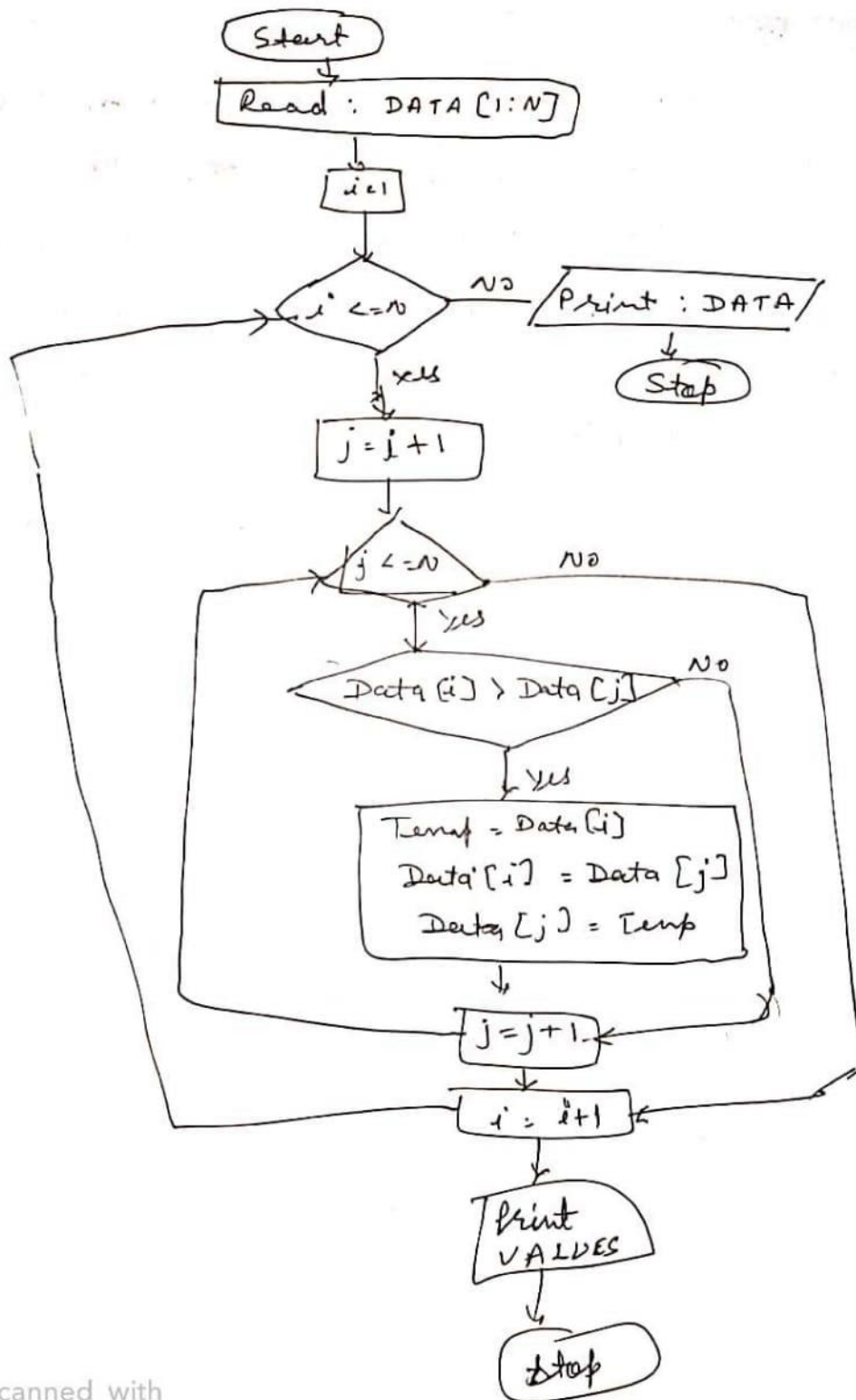
Step 14 : PRINT arr[i]

Step 15 : i = i + 1

Step 16 : Return Code Ends

Step 17 : End ~~Begin~~ Start





ALP PROGRAM:

ALP program -

; Name : PUNIT MIDDHA

; Reg no : 19 BCE 2060

DATA SEGMENT

MSG DB 0DH, 0AH, "Display Result", 0DH,
0AH, "\$"

A DW 10H, 1BH, 0DH, 17H, 12H, 0AH
SZ DW 06H

DATA ENDS

ASSUME CS:CODE, DS:DATA

CODE SEGMENT

START : MOV AX, DATA

MOV DS, AX

MOV SI, 0000H

MOV BX, 06H

DEC BX

X2 : MOV CX, BX

MOV SI, 00H

X1 : MOV AX, A[SI]

INC SI

INC SI

CMP AX, A[SI]

JNA X3

XCHG AX, A[SI]

MOV A[SI-2], AX

X3 : LOOP X1

DEC BX

JNZ X2



Scanned with
CamScanner

```
MOV AH, 09H
MOV DX, OFFSET MSG
INT 21H
```

```
LEA SI, A1
UP: MOV AX, [SI]
CALL PRINT
```

```
MOV DL, B2
MOV AH, 02H
INT 21H
```

```
INC SI
INC SI
DEC SZ
JNZ UP
MOV AH, 4CH
INT 21H
```

```
PRINT PROC
```

```
MOV CX, 0
MOV DX, 0
Label1:
CMP AX, 0
JE print1
MOV BX, 10
DIV BX
PUSH DX
INC CX
XOR DX, DX
JMP Label1
```



PRINT1:

```
CMP CX,0
JE exit
POP DX
Add DX,48
MOV AH,02H
INT 21H
DEC CX
JMP PRINT1
```

EXIT:

Ret

PRINT ENDP

CODE ENDS

END START




Scanned with
CamScanner

```
edit: C:\Users\Punit Middha\Desktop\LABFAT\MICRO\labfat.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
01: You may customize this and other start-up templates:
02: The location of this template is c:\emu8086\inc\8086_con_template.txt
03:
04: NAME: PUNIT MIDDHA
05: REGNO: 19BCE2060
06:
07: DATA SEGMENT
08: MSG DB 0DH,0AH,"DISPLAY Result",0DH,0AH,"$"
09: A DB 10h, 10h, 0Dh, 17h, 12h, 0Ah
10: SZ DB 06h
11: DATA ENDS
12:
13: ASSUME CS:CODE,DS:DATA
14:
15: CODE SEGMENT
16: START: MOV AX,DATA
17:         MOV DS,AX
18:         MOV SI,0000H
19:         MOV BX,06h
20:         DEC BX
21:
22:         X2: MOV CX,BX
23:              MOV SI,00H
24:
25:         X1: MOV AX,A[SI]
26:              INC SI
27:              INC SI
28:              CMP AX,A[SI]
29:              JNE X3
30:              XCHG AX,A[SI]
31:              MOV A[SI-2],AX
32:
33:         X3: LOOP X1
34:              DEC BX
35:              JNZ X2
36:
37: MOV AH, 09H
38: MOV DX, OFFSET MSG
39: INT 21H
40:
41: LEA SI, A
42: UP: MOV AX, (SI)
43: CALL PRINT
44:
45: MOV DL, 32
46: MOV AH, 02H
47: INT 21H
48:
49: INC SI
50: DEC SI
51: JNZ UP
52: MOV AH, 4CH
53: INT 21H
54:
55: INC SI
56: DEC SI
57: JNZ UP
58: MOV AH, 4CH
59: INT 21H
60:
line: 24 col: 63 drag a file here to open
```

```
edit: C:\Users\Punit Middha\Desktop\LABFAT\MICRO\labfat.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
53
54 INC SI
55 INC SI
56 DEC SZ
57 JNZ UP
58 MOV AH,4Ch
59 INT 21h
60
61
62
63 PRINT PROC
64     mov cx,0
65     mov dx,0
66     label1:
67         cmp ax,0
68         je print1
69         mov bx,10
70         div bx
71         push dx
72         inc cx
73         xor dx,dx
74         jmp label1
75     print1:
76         cmp cx,0
77         je exit
78         pop dx
79         add dx,48
80         mov ah,02h
81         int 21h
82         dec cx
83         jmp print1
84     exit:
85     ret
86 PRINT ENDP
87
88 CODE ENDS
89 END START
90
91
92 ;HEXA: 10h, 10h, 0Dh, 17h, 12h, 00h
93 ;DECIMAL: 16, 22, 13, -23, -18, 10
```

IMPLEMENTATION:

 emulator screen (80x25 chars)

