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 as16, 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, +3, -23, -18, 10}. And most important we have to 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

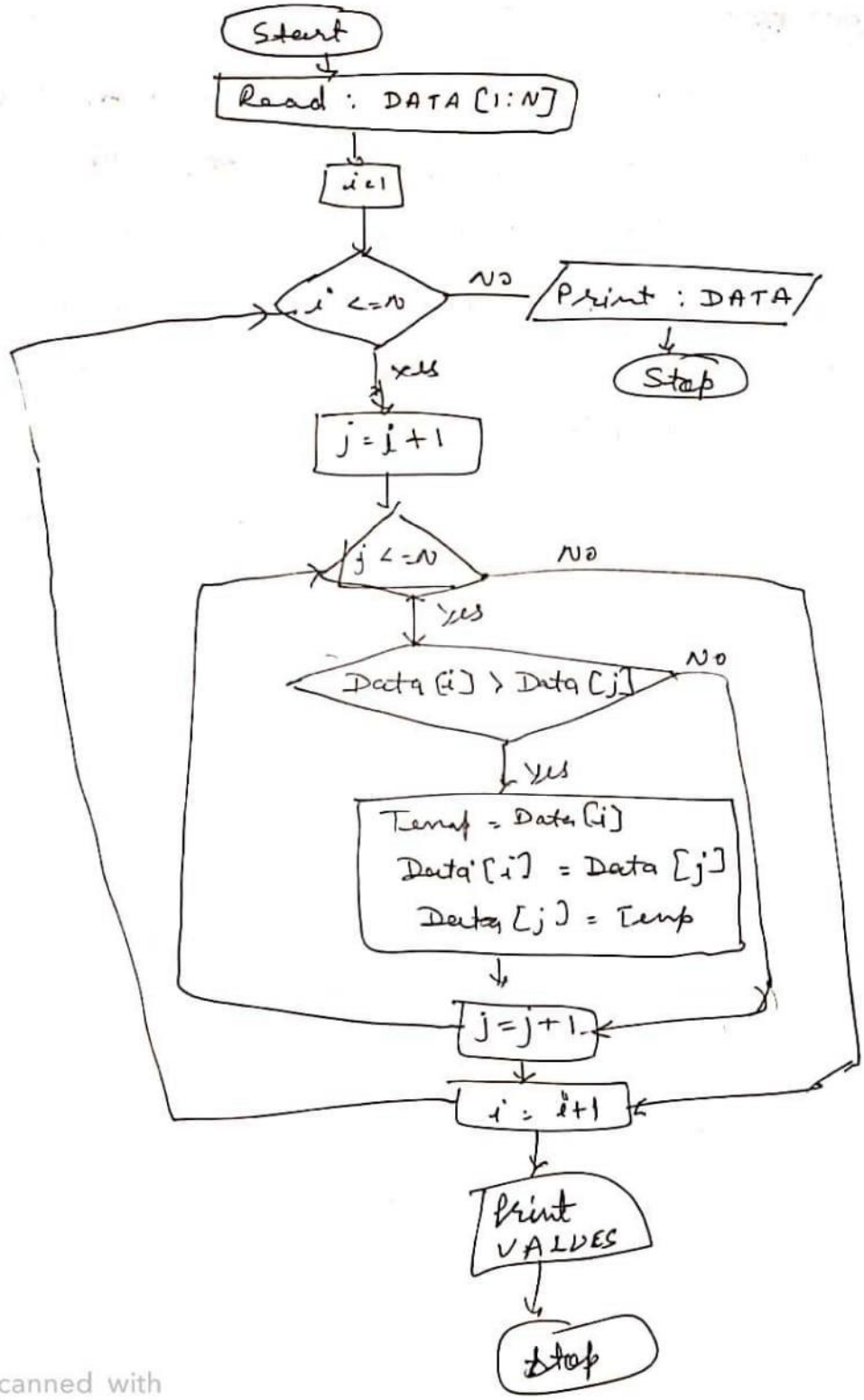
Step 13 : Set i = 0; REPEAT step 14 and step 15 until i < length

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
; Regno : 19BCE2060

DATA SEGMENT
    MSG DB 0DH, 0AH, "Display Result", 0DH,
            0AH, "$"
    A DW 10h, 1Bh, 0Dh, 17h, 12h, 0Ah
    SL 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
```

```
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 S2
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
```

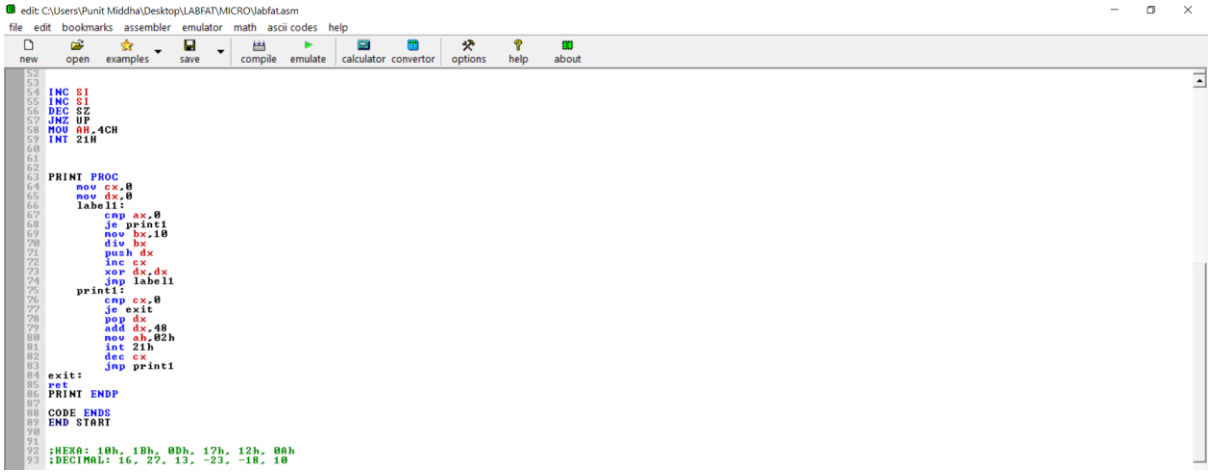

edit: C:\Users\Punit Middha\Desktop\LABFAT\MICRO\labfat.asm

file edit bookmarks assembler emulator math ascii codes help

new open examples save compile emulate calculator convertor options help about

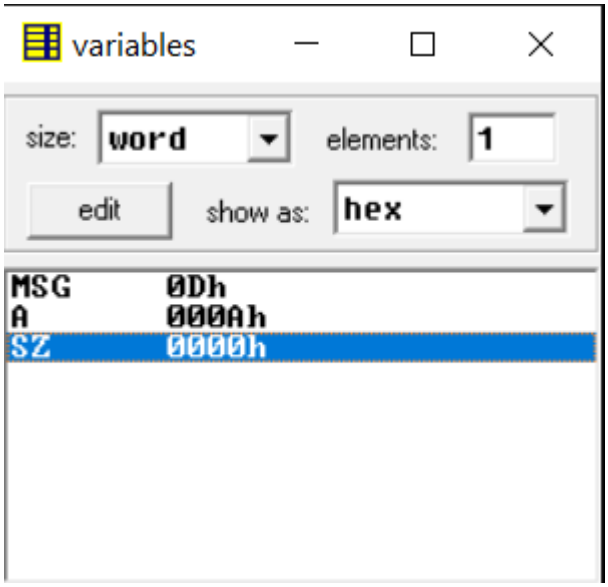
```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

; NAME: PUNIT MIDDHA
; REGNO: 19BCE2060

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


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


LEA SI, A
UP: MOV AX, [SI]
CALL PRINT

MOV DL, 32
    MOV AH, 02H
    INT 21H


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

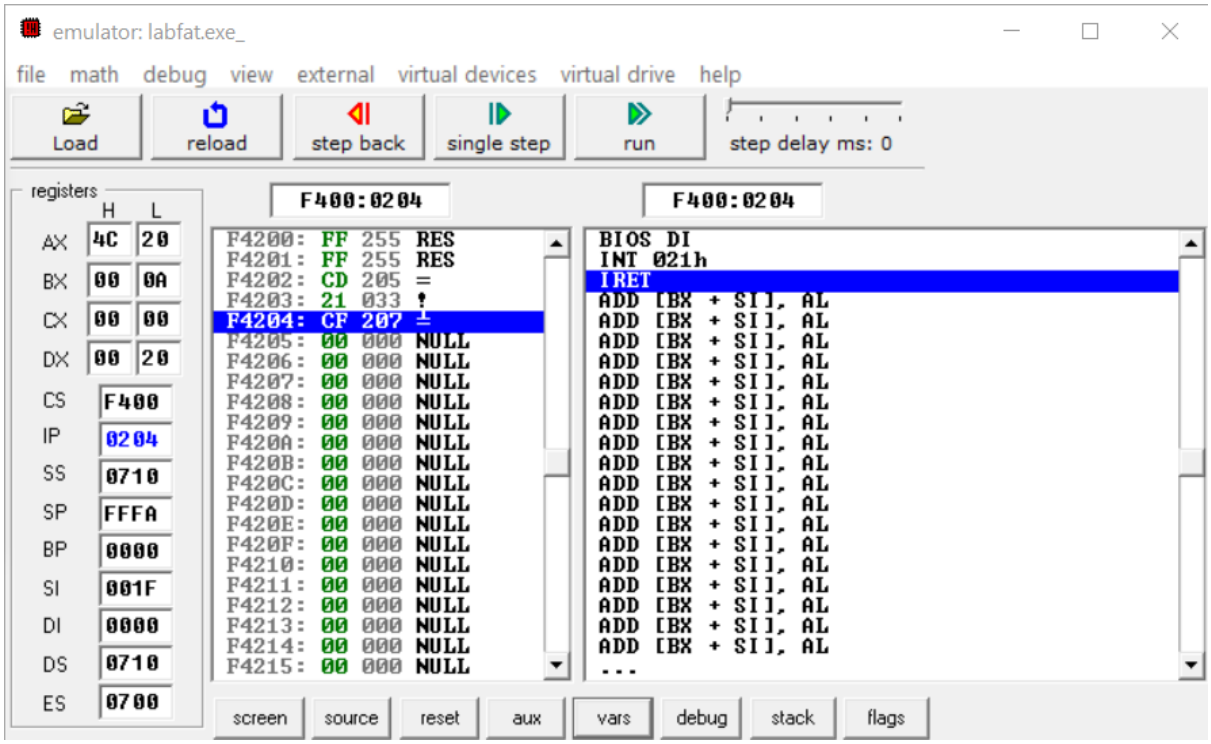
line: 24 col: 63 drag a file here to open




```asm
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


;HEXA: 10h, 1Bh, 0Dh, 17h, 12h, 0Ah
;DECIMAL: 16, 27, 13, -23, -18, 10
```

IMPLEMENTATION:





RESULTS:

RESULTS :-

As we can see the output screen for the first input { 16, 27, 13, 23, 18, 10 }, the program displaying the correct result as { 10, 13, 16, 18, 23, 27 }. So, for input 1 we can say the code is working fine.

HENCE, Using EMU8086 emulator we have successfully implemented ascending order code.