**EXPERIMENT REPORT OF ASSEMBLY LANGUAGE**

Assignment 2 Experiment1

NAME : ABID ALI

STUDENT ID :2019380141

DATE : 4/15/2021

SUBMITTED TO :PROFESSOR Yin LU

**Problem Description:**

Sorting of 10 Defined Byte data. There are 10 Defined Byte data in the variable called DATAS, and the count of 10 in the variable DATANUM. Please sort the 10 data in DATAS in the order of from smaill to large.

The template is given below:

;========================================================

;Description: Program of Assignment 2 Experiment1

;Author:[name][student ID]

;Date:[Date]

;========================================================

.MODEL SMALL

.STACK 32

.DATA

DATANUM DB 10

DATAS DB 21H,13H,4H,5H,7H, 6H,8H,20H,9H,11H

.CODE

MAIN PROC FAR

; INITIALIZE DATA SEGMENT

MOV AX, @DATA

MOV DS, AX

;here is the program body

; RETURN TO DOS

MOV AX, 4C00H

INT 21H

MAIN ENDP

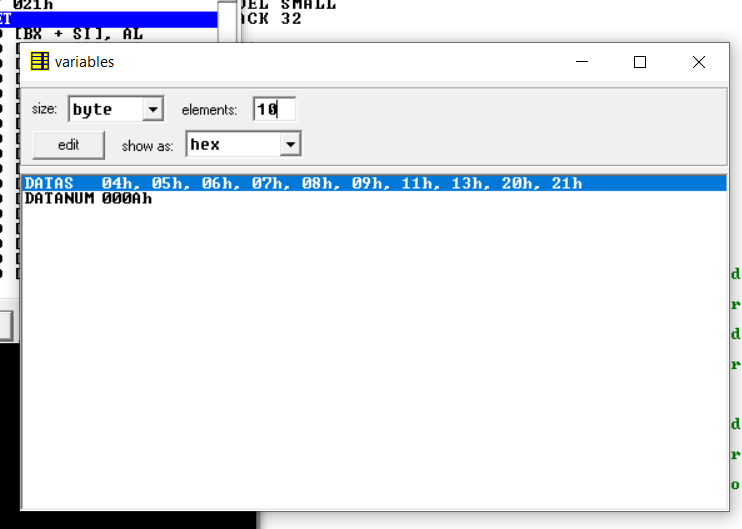
END MAIN

**Goal:**

Sorting 10 hexadecimal number from small to big.

Given 10 unsorted hexadecimal number : 21H,13H,4H,5H,7H, 6H,8H,20H,9H,11H

10 sorted hexadecimal number small to big : 4H,5H,6H,7H,8H,9H,11H,13H,20H,21H



**Code:**

;========================================================

;Description: Program of Assignment 2 Experiment1

;Author:[ABID ALI][2019380141]

;Date:[2021/05/14]

;SORTING ARRAY BY USING BUBBLE SORT ALGORITHM

;========================================================

INCLUDE 'EMU8086.INC'

.MODEL SMALL

.STACK 32

.DATA

DATAS DB 21H,13H,4H,5H,7H, 6H,8H,20H,9H,11H

DATANUM DW 10

.CODE

MAIN PROC

MOV AX, @DATA ; initialize DS

MOV DS, AX

MOV BX,DATANUM ; set BX=10

LEA SI, DATAS ; set SI=offset address of DATAS

CALL PRINT\_DATAS ; call the procedure PRINT\_DATAS

LEA SI, DATAS ; set SI=offset address of the DATAS

CALL BUBBLE\_SORT ; call the procedure BUBBLE\_SORT

PRINTN

LEA SI, DATAS ; set SI=offset address of DATAS

CALL PRINT\_DATAS ; call the procedure PRINT\_DATAS

MOV AH, 4CH ; return control to DOS

INT 21H

MAIN ENDP

;------------------------- Procedure Definitions ------------------------;

;----------------------------- PRINT\_DATAS ------------------------------;

PRINT\_DATAS PROC

; this procedure will print the elements of a given array

; input : SI=offset address of the array

; : DATANUM=BX=size of the array

; output : none

PUSH AX ; push AX onto the STACK

PUSH CX ; push CX onto the STACK

PUSH DX ; push DX onto the STACK

MOV CX, BX ; set CX=BX

@PRINT\_DATAS: ; loop label

XOR AH, AH ; clear AH

MOV AL, [SI] ; set AL=[SI]

CALL OUTDEC ; call the procedure OUTDEC

MOV AH, 2 ; set output function

MOV DL, 20H ; set DL=20H

INT 21H ; print a character

INC SI ; set SI=SI+1

LOOP @PRINT\_DATAS ; jump to label @PRINT\_DATAS while CX!=0

POP DX ; pop a value from STACK into DX

POP CX ; pop a value from STACK into CX

POP AX ; pop a value from STACK into AX

RET ; return control to the calling procedure

PRINT\_DATAS ENDP

;---------------------------- BUBBLE\_SORT -------------------------------;

BUBBLE\_SORT PROC

; this procedure will sort the array in ascending order

; input : SI=offset address of the array

; : BX=array size

; output : none

PUSH AX ; push AX onto the STACK

PUSH BX ; push BX onto the STACK

PUSH CX ; push CX onto the STACK

PUSH DX ; push DX onto the STACK

PUSH DI ; push DI onto the STACK

MOV AX, SI ; set AX=SI

MOV CX, BX ; set CX=BX

DEC CX ; set CX=CX-1

@OUTER\_LOOP: ; loop label

MOV BX, CX ; set BX=CX

MOV SI, AX ; set SI=AX

MOV DI, AX ; set DI=AX

INC DI ; set DI=DI+1

@INNER\_LOOP: ; loop label

MOV DL, [SI] ; set DL=[SI]

CMP DL, [DI] ; compare DL with [DI]

JNG @SKIP\_EXCHANGE ; jump to label @SKIP\_EXCHANGE if DL<[DI]

XCHG DL, [DI] ; set DL=[DI], [DI]=DL

MOV [SI], DL ; set [SI]=DL

@SKIP\_EXCHANGE: ; jump label

INC SI ; set SI=SI+1

INC DI ; set DI=DI+1

DEC BX ; set BX=BX-1

JNZ @INNER\_LOOP ; jump to label @INNER\_LOOP if BX!=0

LOOP @OUTER\_LOOP ; jump to label @OUTER\_LOOP while CX!=0

POP DI ; pop a value from STACK into DI

POP DX ; pop a value from STACK into DX

POP CX ; pop a value from STACK into CX

POP BX ; pop a value from STACK into BX

POP AX ; pop a value from STACK into AX

RET ; return control to the calling procedure

BUBBLE\_SORT ENDP

;-------------------------------- OUTDEC --------------------------------;

OUTDEC PROC

; this procedure will display a decimal number

; input : AX

; output : none

PUSH BX ; push BX onto the STACK

PUSH CX ; push CX onto the STACK

PUSH DX ; push DX onto the STACK

XOR CX, CX ; clear CX

MOV BX, 10 ; set BX=10

@OUTPUT: ; loop label

XOR DX, DX ; clear DX

DIV BX ; divide AX by BX

PUSH DX ; push DX onto the STACK

INC CX ; increment CX

OR AX, AX ; take OR of Ax with AX

JNE @OUTPUT ; jump to label @OUTPUT if ZF=0

MOV AH, 2 ; set output function

@DISPLAY: ; loop label

POP DX ; pop a value from STACK to DX

OR DL, 30H ; convert decimal to ascii code

INT 21H ; print a character

LOOP @DISPLAY ; jump to label @DISPLAY if CX!=0

POP DX ; pop a value from STACK into DX

POP CX ; pop a value from STACK into CX

POP BX ; pop a value from STACK into BX

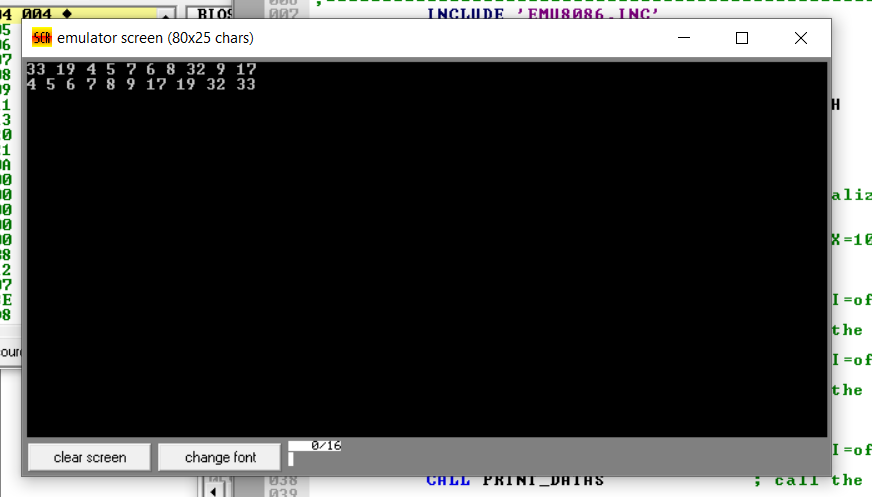
RET ; return control to the calling procedure

OUTDEC ENDP

END MAIN

**Debugging:**

This is a new IDE for me ,I have never used it before .So,at the beginning ,I couldn’t understand the use of so many features.Eventually,after watching video and getting suggestion from my friend.I used lot of sub function to call another sub function.By doing that,the program looks complicated but it is working eventually.I tried to show the result in Hexadecimal in prompt screen but it shows the result in Decimal.I tried to add another sub function to convert decimal to hexadecimal .I make the program length and complicated.So,I removed it and kept it as like that.Now,the Prompt screen shows the sorting in Decimal but we can see the sorting in hexadecimal in the **vars**.



**Attachment:**

1) Experiment-1(part-2).m4v

2) Exp-1\_part2.asm

**Acknowledgement:**

I complete this assignment by myself by using online videos and taking help from online.There were lot of similar program like this in the internet.I tried to understand the algorithm andlogic from those program and implemented in this program.