

## **MODEL PRACTICAL EXAMINATION**

**-SRINITHYEE S K**

**185001166**

---

### **QUESTIONS:**

1. A set of nine numbers are stored in memory. Write an ALP using 8086 to find the largest number in the set and display the its position.
2. Write 8051 ALP to convert BCD to HEX.

### **SOLUTIONS**

**1.**

#### **ALGORITHM:**

1. Start
2. Declare the data segment
3. Initialise the data segment with an array holding 9 numbers and a variable to hold the length and the largest number
4. Close the data segment
5. Declare the code segment
6. Set the off set (preferably 100)
7. Load the data segment content into AX register
8. Use a pointer to parse the array
9. Store the length of the array in cl
10. The index of the largest element is stored in bx register
11. Loop through the array comparing each elements of the array
12. Now compare value of register AL from data(value) at next offset, if that data is greater than value of register AL then update value of register AL to that data else no change, and increase offset value for next comparison and decrease cl by 1 and continue this till count (value of register Cl) becomes 0.
13. Introduce an interrupt for safe exit (int 21h)
14. Close the code segment.
15. End.

## **PROGRAM:**

;Program to find the index of the largest of 9 numbers  
assume cs:code, ds:data

data segment

```
array db 0Bh, 0Dh, 03h, 0Ah, 0Bh, 23h, 47h, 7Fh, 18h
len db 08h
org 0010h
largest db ?
```

data ends

code segment

```
org 0100h
```

start:

```
mov ax, data
mov ds, ax
```

```
mov si, offset array ; Pointer to parse
mov cl, len           ; length of the array
```

```
mov bx, 0000h        ; index of largest element
```

loop:

```
mov al, [si]
cmp al, [bx]
jc here
mov bx, si
```

here:

```
inc si
dec cl
jnz loop
```

```
mov largest, bl
mov ah, 4ch
int 21h
```

code ends

end start

## SNAPSHOTS:

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
(C) Copyright 1982, 1983 by Microsoft Inc.

Run File [LARGEST.EXE]:
List File [NUL.MAP]:
Libraries [LIB]:
Warning: No STACK segment

There was 1 error detected.

D:\>DEBUG LARGEST.EXE
-U
076C:0100 B86A07      MOV     AX,076A
076C:0103 8ED8        MOV     DS,AX
076C:0105 BE0000      MOV     SI,0000
076C:0108 8A0E0900     MOV     CL,[0009]
076C:010C BB0000      MOV     BX,0000
076C:010F 8A04        MOV     AL,[SI]
076C:0111 3A07        CMP     AL,[BX]
076C:0113 7202        JB      0117
076C:0115 8BDE        MOV     BX,SI
076C:0117 46          INC     SI
076C:0118 FEC9        DEC     CL
076C:011A 75F3        JNZ     010F
076C:011C 8B1E1000     MOV     [0010],BL
-
```

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
076C:0115 8BDE        MOV     BX,SI
076C:0117 46          INC     SI
076C:0118 FEC9        DEC     CL
076C:011A 75F3        JNZ     010F
076C:011C 8B1E1000     MOV     [0010],BL
-D 076A
075A:0760                8A 4E E7 B5 00 3B      .N...:
075A:0770 C8 77 44 89 CB 80 BF B8-2C 00 74 36 D1 E3 8B 87 .wD.....t6...
075A:0780 AE 16 3B 46 FC 73 08 C7-87 AE 16 00 00 EB 0E 8A ...:F.s.....
075A:0790 5E E7 B7 00 D1 E3 8B 46-FC 29 87 AE 16 8A 5E E7 ^.....F.).....^
075A:07A0 B7 00 D1 E3 8B 87 AE 16-3B 06 B2 2C 72 04 40 A3 .....:.,r.@.
075A:07B0 B2 2C FE 46 E7 75 AF 8A-1E B6 2C B7 00 D1 E3 A1 ...:F.u.....
075A:07C0 B2 2C 89 87 AE 16 8A 46-06 D0 D8 73 0B 8A 1E B6 .....F...s...
075A:07D0 2C B7 00 C6 87 48 2F FF-8A 1E B6 2C B7 00 D1 E3 ....H/.....
075A:07E0 8B 87 FA 15 89 EC 5D CA-06 00                .....l...
-D 076A:0000
076A:0000 0B 0D 03 0A 0B 23 47 7F-18 08 00 00 00 00 00 00 .....#G.....
076A:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
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 .....
-
```

```

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
075A:07C0 B2 2C 89 87 AE 16 8A 46-06 D0 D8 73 0B 8A 1E B6 .....F...s...
075A:07D0 2C B7 00 C6 87 48 2F FF-8A 1E B6 2C B7 00 D1 E3 .....H/.....
075A:07E0 8B 87 FA 15 89 EC 5D CA-06 00 .....1...
-D 076A:0000
076A:0000 0B 0D 03 0A 0B 23 47 7F-18 08 00 00 00 00 00 00 .....#G.....
076A:0010 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
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 0B 0D 03 0A 0B 23 47 7F-18 08 00 00 00 00 00 00 .....#G.....
076A:0010 07 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
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 .....
-

```

2.

### ALGORITHM:

1. Start
2. Take the input value and segregate it into digits using anl and swap.
3. Multiply the ten's place digit with 0A.
4. Add one's place digit to the previous result.
5. Now store the result in R1
6. End

### PROGRAM:

```

START:
MOV R1, #16H ;INITIAL BCD VALUE

MOV A, R1
ANL A, #0FH ;TAKE THE LOWER NIBBLE
MOV R2, A ;PUT IT IN R2
MOV A, R1
ANL A, #0F0H ;TAKE THE HIGHER NIBBLE
SWAP A ;SWAP LOWER AND HIGHER NIBBLE

;HIGHER NIBBLE * 10 + LOWER NIBBLE

MOV B, #0AH ; MSB * 10
MUL AB

```

```

ADD A, R2      ; + LSB
MOV R2, A      ;RESULT TO R2

```

```

HALT:
SJMP HALT

```

## SNAPSHOTS:-

The screenshot displays the Proteus ISIS simulation environment for an 8051 microcontroller. The top section shows the System Clock (MHz) set to 12.0 and the Update Freq. dropdown set to 1. Below this, the Register File is visible, showing the status of various registers including R0 through R7, B, ACC, PSW, IP, IE, PCON, DPH, DPL, and SP. The Program Counter (PC) is highlighted in blue and shows the value 8051. The Data Memory section shows a table of memory addresses (00 to 70) and their corresponding values (00 to FF). The bottom section shows the Copyright information and a button to Remove All Breakpoints.

System Clock (MHz)	Update Freq.
12.0	1

R/O	W/O	TH0	TL0	R7	B
0x00	0x00	0x00	0x00	0x00	0x02

R6	ACC
0x00	0x20

R5	PSW
0x00	0x41

R4	IP
0x00	0x00

R3	IE
0x00	0x00

R2	PCON
0x00	0x00

R1	DPH
0x20	0x00

R0	DPL
0x32	0x00

SP
0x07

pins	bits	TH1	TL1
0xFF	0xFF	0x00	0x00

PC
0x0010

PSW
0 1 0 0 0 0 0 1

addr	0x00	0x32	value
0	00	32	20

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	32	20	00	00	00	00	00	00	00	00	00	00	00	00	00

Copyright ©2005-2016 James Rogers

Remove All Breakpoints

## RESULT

The Program Given has been executed and implemented successfully.