

```

1
2 section .data
3     array db 11h,59h,33h,22h,44h
4
5     msg1 db 10,"ALP to find the largest number in an array",10
6     msg1_len equ $ - msg1
7
8     msg2 db 10,"The Array contains the elements : ",10
9     msg2_len equ $ - msg2
10
11     msg3 db 10,10, "The Largest number in the array is : ",10
12     msg3_len equ $ - msg3
13
14 section .bss
15     counter resb 1
16     result resb 4
17
18 %macro write 2
19     mov rax,1
20     mov rdi,1
21     mov rsi,%1
22     mov rdx,%2
23     syscall
24 %endmacro
25
26 section .text
27     global _start
28
29 _start:
30
31     write msg1 , msg1_len
32     write msg2 , msg2_len
33
34     mov byte[counter],05
35     mov rsi,array
36 next:  mov al,[rsi]
37         push rsi
38         call disp
39         pop rsi
40         inc rsi
41         dec byte[counter]
42         jnz next
43
44     write msg3 , msg3_len
45
46     mov byte[counter],05
47     mov rsi, array
48
49     Microprocessor Laboratory
50     Department of Computer Engineering, BSIOTR 7
51
52     mov al, 0           ; al is an 8 bit register , al stores max
53 repeat: cmp al,[rsi]    ;cmp opr1 , opr2 : opr1 - opr2

```

```

54         jg skip
55         mov al,[rsi]
56 skip:    inc rsi
57         dec byte[counter]
58         Jnz repeat
59
60         call disp
61
62         mov rax,60
63         mov rdi,1
64         syscall
65
66 disp:
67         mov bl,al                ;store number in bl
68         mov rdi, result          ;point rdi to result variable
69         mov cx,02                ;load count of rotation in cl
70 up1:
71         rol bl,04                ;rotate number left by four bits
72         mov al,bl                ;move lower byte in dl
73         and al,0fh               ; get only LSB
74         cmp al,09h               ;compare with 39h
75         jg add_37                ;if greater than 39h skip add 37
76         add al,30h
77         jmp skip1                ;else add 30
78 add_37: add al,37h
79 skip1:  mov [rdi],al             ;store ascii code in result variable
80         inc rdi                  ;point to next byte
81         dec cx                   ;decrement the count of digits to display
82         jnz up1                  ;if not zero jump to repeat
83
84         write result , 4
85
86         ret

```