

Name: Manthan . M . Sonawane
 Panel: E Roll No: 17

MAIOT LAB Assignment - 1)

Problem Statement: Write an ALP to sort 8-bit numbers in ascending or descending order

Theory: Explain new instructions used

1. ROL - used to rotate bits of byte/word towards left.
2. AND - used for adding each bit in a byte/word with the corresponding bit in another byte/word
3. CMP - used to compare 2 provided byte/word
4. JBE - used to ~~jump~~ jump before instruction satisfies
5. JNZ - used to jump if not zero flag $zf=0$
6. JAE - used to jump if above instructions satisfies

Algorithm used

① For sorting the integer numbers in ascending/descending order :

1. Declare numbers to be sorted
2. Specify counter to perform sorting
3. select pointer
4. Decide instruction to be used for decisions of greatest smaller number
5. Arrange the sorted in ascending or descending order.
6. Use 2 digit display for unpacking of numbers
7. Print all numbers and terminate the code

- ② Display sorted numbers
- ① set d to 5
2. store array in rsi
3. store result in rdi
4. set dl to 2
5. store [rsi] in al
6. Rotate al 4 times to the left
7. Store al in bl
8. And al and Fh
9. compare al with 9H
10. IF less than 9H go to step 12
11. ADD al and 7H
12. ADD 30H to al
13. dec al
14. IF not 0, go to step 6
15. Store AH in rdi
16. increment rdi
17. decrement al
18. IF not 0, step 4

Platform → open source Linux

Conclusion → This, the program is implemented in an assembly language to sort 8 bit number

FAQ's

- 1) Explain the following instructions with an example.
→ CMP x CHQ - this instructions is used to compare two operands & exchange

BSWP - this instruction takes the contents of any 32 bit register and swap the first byte with fourth, and the second with the third

PUSHA - it is used to put all the register into the stack

POPA - it is used to get words from the stack to all register

Q.2. Write down the algorithm if we have to accept numbers from users

-
- i) Declare variable temp
 - ii) Input from user the number and store it in temp
 - iii)

```
mov rax, 0
mov rdi, 0
mov rsi, temp
mov rcx, 6
```
 - iv) end

%macro operate 4 ; macro declaration

mov rax,%1

mov rdi,%2

mov rsi,%3

mov rdx,%4

syscall

%endmacro

section .data

msg db "Sorted array is: ", 10

msglen equ \$-msg

arr db 05h,0Ah,75h,0D3h,12h

section .bss

result resb 15 ;used to display the sorted number

section .text

global _start

_start:

mov bl,5

;outer loop runs for n times

loop_outer: mov cl,4

;inner loop runs n-1 times

```
mov rsi,arr
```

```
up: mov al,byte[rsi]
```

```
cmp al,byte[rsi+1]
```

```
jbe only_inc  
;no swapping
```

```
xchg al,byte[rsi+1]  
;swap
```

```
mov byte[rsi],al
```

```
only_inc: inc rsi
```

```
dec cl  
;decrementing inner loop
```

```
jnz up
```

```
dec bl  
;decrementing outer loop
```

```
jnz loop_outer
```

```
operate 1,1,msg, msglen
```

```
mov rdi,arr ;unpacking
```

```
mov rsi,result
```

```
mov dl,5  
;for one number there are two digits
```

```
disp_loop1:
```

```
mov cl,2
```

```
mov al,[rdi]
```

```
againx:
```

```
rol al,4  
;rotate by 4
```

```
mov bl,al
```

```
and al,0FH
```

```
cmp al,09H
```

```
jbe downx ;For ascending order
```

```
add al,07H
```

```
downx:
```

```
add al,30H
```

```
mov byte[rsi],al
```

```
mov al,bl
```

```
inc rsi
```

```
dec cl
```

```
jnz againx
```

```
mov byte[rsi],0AH  
;inserting enter
```

```
inc rsi  
;result
```

```
inc rdi
```

```
dec dl
```

jnz disp_loop1

operate 1,1,result,15

operate 60,0,0,0

OUTPUT

