**Code:**

%macro write 2

mov eax,4

mov ebx,1

mov ecx,%1

mov edx,%2

int 0x80

%endmacro

%macro read 2

mov eax,3

mov ebx,2

mov ecx,%1

mov edx,%2

int 0x80

%endmacro

section .data

p1 db "Enter number of elements: "

p1len equ $-p1

p2 db "Enter contents of array: ",10

p2len equ $-p2

p3 db "The sorted array is: ",10

p3len equ $-p3

p4 db "Pass "

p4len equ $-p4

p5 db " : "

p5len equ $-p5

newline db 10

spa db " "

section .bss

n resb 4

arr resb 10

i resb 4

j resb 4

trash resb 1

section .text

global \_start

\_start:

write p1,p1len

read n,1

read trash,1

call input

write newline,1

mov eax,'0'

mov [j],eax

call bubblesort

write p3,p3len

call display

mov eax,1

xor ebx,ebx

int 0x80

input:

write p2,p2len

mov [i],dword '0'

L1:

mov esi,[i]

cmp esi,[n]

jge after1

sub esi,'0'

add esi,arr

read esi,1

read trash,1

inc dword[i]

jmp L1

after1:

ret

display:

write p4,p4len

write j,1

write p5,p5len

mov [i],dword '0'

L2:

mov esi,[i]

cmp esi,[n]

jge after2

sub esi,'0'

add esi,arr

write esi,1

write spa,1

inc dword[i]

jmp L2

after2:

write newline,1

ret

bubblesort:

mov al,0

mov bl,[n]

sub bl,'0'

sub bl,1

L3:

cmp al,bl

jge after1

pushad

call display

popad

mov ecx,0

mov dl,bl

sub dl,al

L4:

cmp cl,dl

jge after3

mov esi,arr

add esi,ecx

mov ah,[esi]

mov bh,[esi+1]

cmp ah,bh

jle after4

mov [esi+1],ah

mov [esi],bh

after4:

inc cl

jmp L4

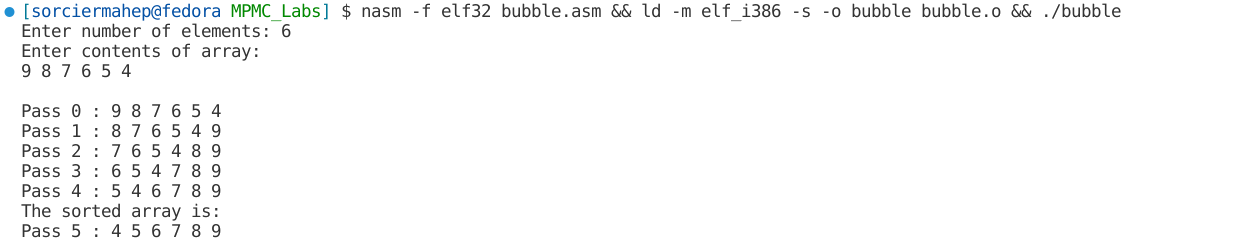
after3:

inc al

inc byte[j]

jmp L3

**Output:**

****

**Code:**

%macro write 2

mov eax,4

mov ebx,1

mov ecx,%1

mov edx,%2

int 0x80

%endmacro

%macro read 2

mov eax,3

mov ebx,2

mov ecx,%1

mov edx,%2

int 0x80

%endmacro

section .data

p1 db "Enter number of elements: "

p1len equ $-p1

p2 db "Enter contents of array: ",10

p2len equ $-p2

p3 db "The sorted array is: ",10

p3len equ $-p3

p4 db "Pass "

p4len equ $-p4

p5 db " : "

p5len equ $-p5

newline db 10

spa db " "

section .bss

n resb 4

arr resb 10

i resb 4

j resb 4

trash resb 1

section .text

global \_start

\_start:

write p1,p1len

read n,1

read trash,1

mov eax,'0'

mov [j],eax

call input

call insertionsort

write p3,p3len

call display

mov eax,1

xor ebx,ebx

int 0x80

input:

write p2,p2len

mov [i],dword '0'

L1:

mov esi,[i]

cmp esi,[n]

jge after1

sub esi,'0'

add esi,arr

read esi,1

read trash,1

inc dword[i]

jmp L1

after1:

ret

display:

write p4,p4len

write j,1

write p5,p5len

mov [i],dword '0'

L2:

mov esi,[i]

cmp esi,[n]

jge after2

sub esi,'0'

add esi,arr

write esi,1

write spa,1

inc dword[i]

jmp L2

after2:

write newline,1

ret

insertionsort:

mov eax,1

mov bl,[n]

sub bl,'0'

L3:

cmp al,bl

jge after1

pushad

call display

popad

mov ecx,0

mov cl,al

sub cl,1

mov dl,[arr+eax]

L4:

cmp cl,0

jl after3

cmp dl,[arr+ecx]

jge after3

mov dh,[arr+ecx]

mov [arr+ecx+1],dh

dec ecx

jmp L4

after3:

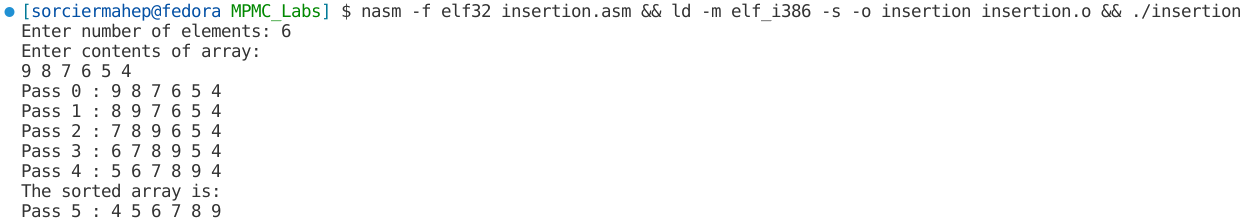
mov [arr+ecx+1],dl

inc al

inc byte[j]

jmp L3

**Output:**

****

**Code:**

%macro write 2

mov eax,4

mov ebx,1

mov ecx,%1

mov edx,%2

int 0x80

%endmacro

%macro read 2

mov eax,3

mov ebx,2

mov ecx,%1

mov edx,%2

int 0x80

%endmacro

section .data

p1 db "Enter number of elements: "

p1len equ $-p1

p2 db "Enter contents of array: ",10

p2len equ $-p2

p3 db "The sorted array is: ",10

p3len equ $-p3

p4 db "Pass "

p4len equ $-p4

p5 db " : "

p5len equ $-p5

newline db 10

spa db " "

section .bss

n resb 4

arr resb 10

i resb 4

j resb 4

trash resb 1

section .text

global \_start

\_start:

write p1,p1len

read n,1

read trash,1

sub byte[n],'0'

mov eax,'0'

mov [j],eax

call input

call selectionsort

write p3,p3len

call display

mov eax,1

xor ebx,ebx

int 0x80

input:

write p2,p2len

mov [i],dword 0

L1:

mov esi,[i]

cmp esi,[n]

jge after1

add esi,arr

read esi,1

read trash,1

inc dword[i]

jmp L1

after1:

ret

display:

write p4,p4len

write j,1

write p5,p5len

mov [i],dword 0

L2:

mov esi,[i]

cmp esi,[n]

jge after2

add esi,arr

write esi,1

write spa,1

inc dword[i]

jmp L2

after2:

write newline,1

ret

selectionsort:

mov eax,0

mov bl,[n]

sub bl,1

L3:

cmp al,bl

jge after1

pushad

call display

popad

mov ecx,0

mov cl,al

add cl,1

mov edi,arr

add edi,eax

L4:

cmp cl,[n]

jge after3

mov esi,arr

add esi,ecx

mov bh,[esi]

mov dh,[edi]

cmp bh,dh

jge after4

mov edi,arr

add edi,ecx

after4:

inc cl

jmp L4

after3:

mov bh,[arr+eax]

mov dh,[edi]

mov [arr+eax],dh

mov [edi],bh

inc al

inc byte[j]

jmp L3

**Output:**

