Experiment No:-05 Date:-

Roll No:- A046

Aim :- To search a number, and its index from an array

Theory:-

Program1:- WAP To search a number and its index from an array

Algorithm :-

Step1:- Initialize the data segment.

Step 2:- Initialize the array element.

Step 3:- Initialize the number to search.

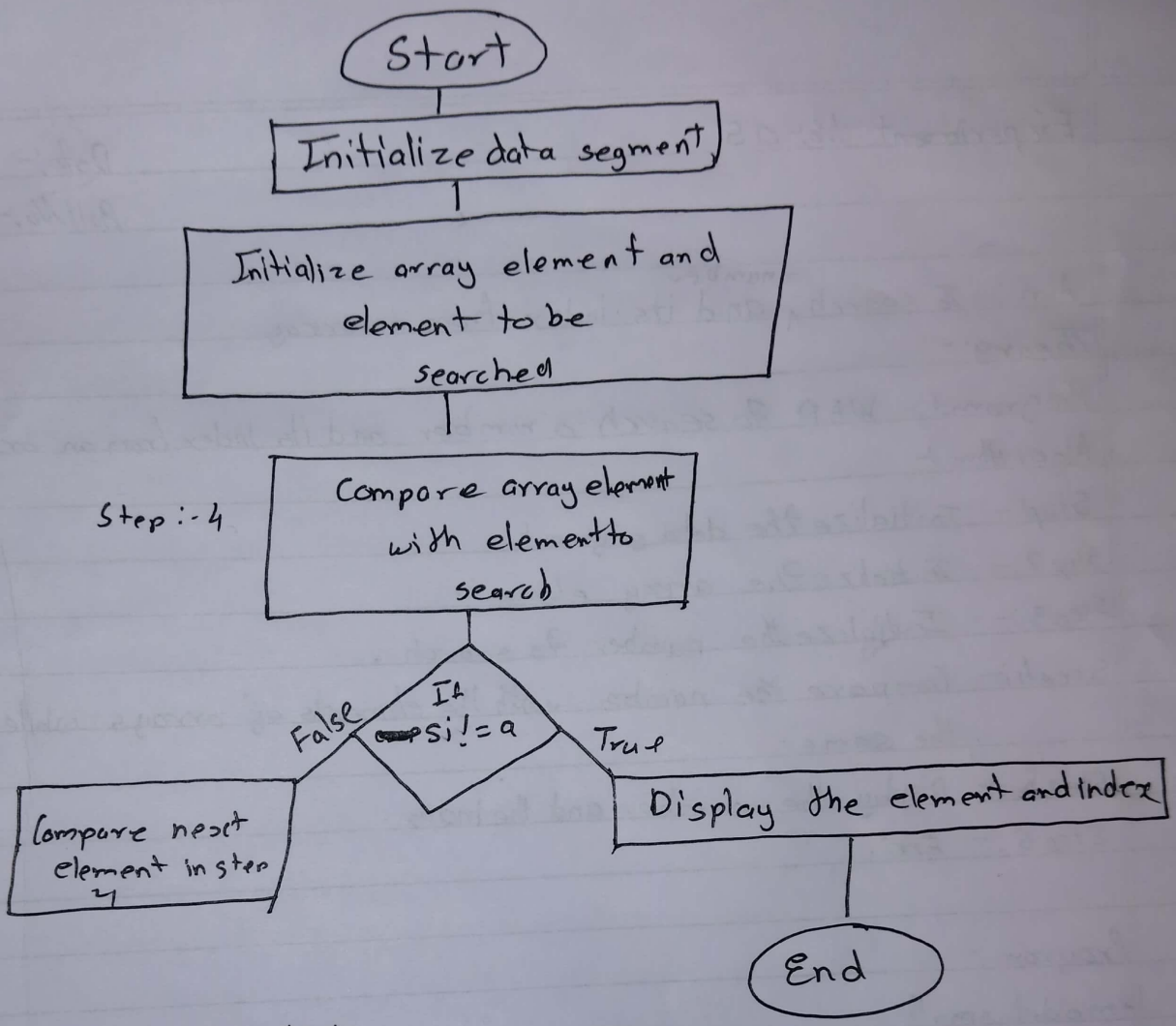
Step4:- Compare the number with the elements of arrays while its not the same.

Step 5 :- Display the number and the index.

Step 6 :- End.

Program:-

```
.model small
.data
arr db 03, 04, 02, 01, 06h
b db 02h
.code
mov ax, @data
mov ds, ax
lea si, arr
mov al, b
mov ch, 01h
back : mov bl, [si] mov ch, 00h
cmp al, bl mov dh, 05h
je I1 back: mov bl, [si]
inc si cmp al, bl
```



Steps to display output

tasm si1.asm

tlink si1.obj

si1

Output

03 01

```
je I1
inc si
dec dh
jmp back
I1: mov bh, ch
hello: mov ch, 02h
mov cl, 04h
I2: rol bh, cl
mov dl, bh
and dl, 0Fh
cmp dl, 09
jbe I11
add dl, 07h
i1: add dl, 30h
mov ah, 02h
int 21h
dec ch
jnz i2
mov dl, '
mov ah, 02h
int 21h
mov bh, bl
dec ah
jnz hello
mov ah, 4ch
int 21h
end
```

Conclusion :- Hence we searched number and its indexe from an array

Experiment No:-06 Date:-

Roll No:-A046

Aim:- To sort an array in ascending/descending order.

Theory:-

Program 1:- Wap to sort a given array in ascending order

Algorithm:-

Step 1:- Initialize the array and data segment

Step 2:- Compare array element ~~a and a+1~~ at index si a and ~~a+1~~ si+1 while counter is not equal to 0;

Step 3:- If a > a+1 swap the elements, repeat step 2 while counter != 0

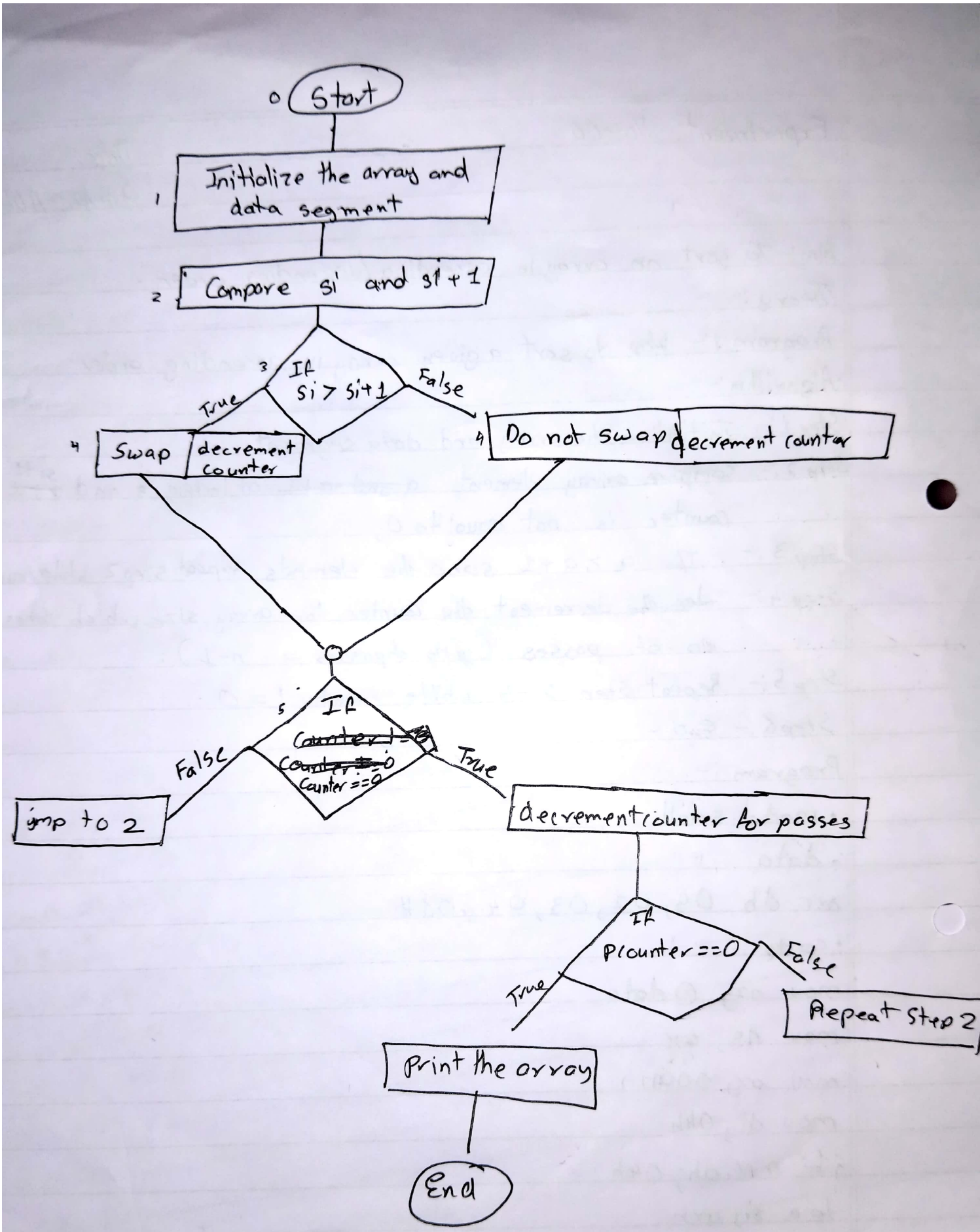
Step 4:- ~~dec the~~ decrement the counter for array size which notes the no. of passes (no. of passes = n-1):

Step 5:- Repeat step 2-4 while counter != 0.

Step 6:- End:-

Program:-

```
.model small
.data
arr db 05,02,03,04,01H
.code
mov ax, @data
mov ds, ax
mov ax, 0000h
mov dl, 04h
j1: mov ch, 04h
lea si, arr
back: mov al, [si]
mov bl, [si+1]
cmp al, bl
~~jbe j3~~ jae j3
```



```
xchg al, bl
i3: mov [si], al
mov [si+1], bl
inc si
dec dh
jnz back
dec dl
jnz i1
~~dec~~ mov dh, 05h
lea si, arr
print: mov al, [si]
inc si
mov dl, al
mov ch, 02h
mov cl, 04h
mov bh, al
i2: rol bh, cl
mov dl, bh
and dl, 0Fh
cmp dl, 09h
jbe i4
add dl, 07h
i4: add dl, 30h
mov ah, 02h
int 21h
dec ch
jnz i2
mov dl, ' '
mov ah, 02h
int 21h
```

Output

01 02 03 04 05

4 Steps to display output

Tasm as - asm

Tlink as. obj

as

```
dec dh
jnz print
mov ah, 4ch
int 21h
end
```

Program 2 :- Wap to sort a given array in descending order.

Algorithm:-

Step 1 :- Initialize array and data segment

Step 2 :- Compare array element at index a and a+1 while counter is != 0 & counter is set to n-1 elements

Step 3 :- If a < a+1 swap the elements

Step 4 :- Decrement the counter which notes the number of passes (no. of passes = n-1)

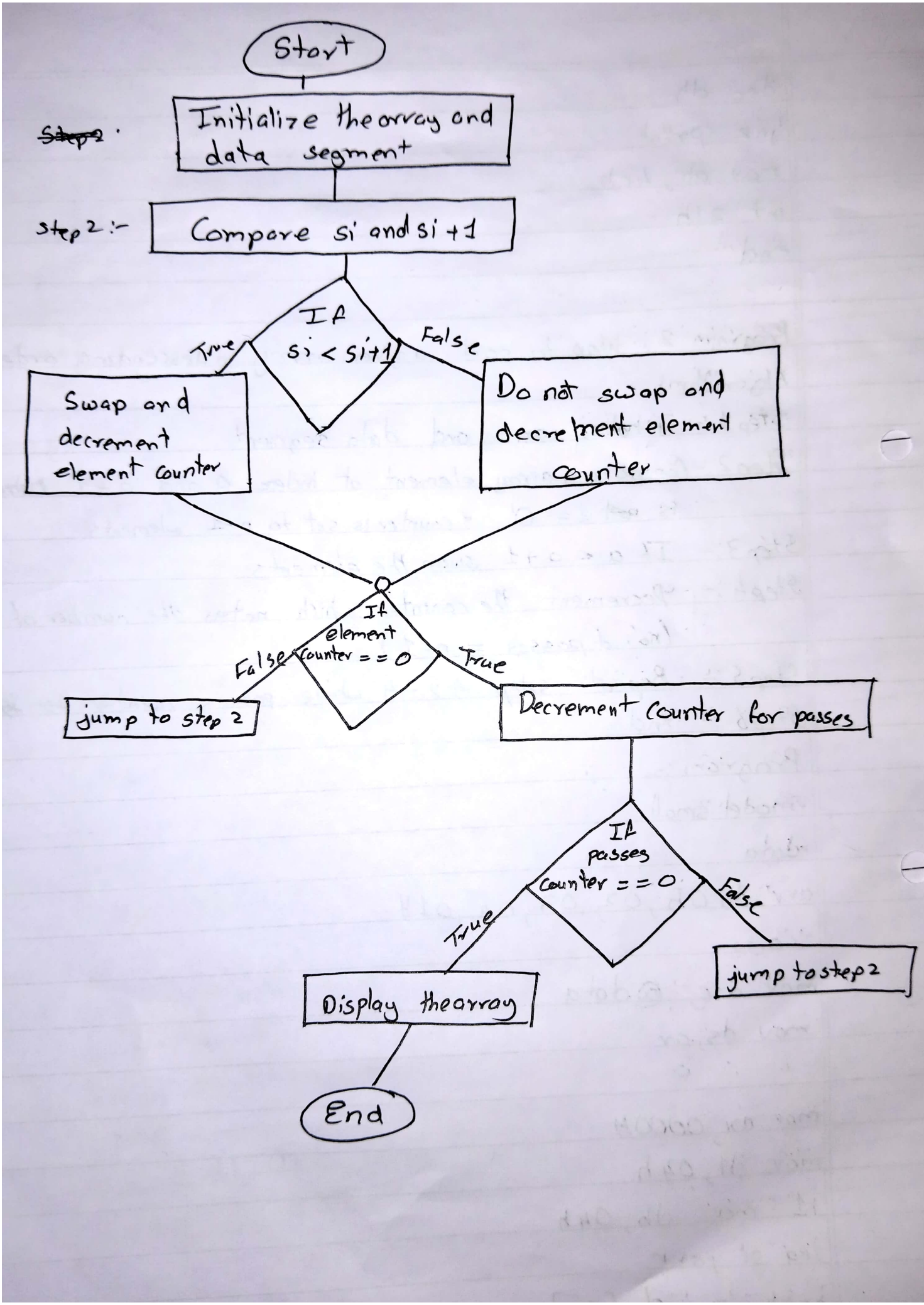
Step 5 :- Repeat step 2-4 while passes counter is != 0

Step 6 :- End

Program :-

```
.model small
.data
arr db 05, 02, 03, 04, 01H
.code
mov ax, @data
mov ds, ax

mov ax, 0000H
mov dl, 04h
j1 : mov dh, 04h
lea si, arr
back : mov al, [si]
mov bl, [si+1]
```



```
cmp al, bl

jbe i3
xchg al, bl

i3: mov [si], al
mov [si+1], bl

inc si
dec dx
jnz back
dec dl
jnz i1

mov db, 05H
lea si, arr
print: mov al, [si]
inc si
mov dl, al

mov ch, 02h
mov cl, 04h

mov bh, al
iz: rol bh, cl
mov dl, bh
and
and dl, 0Fh
cmp dl, 09H
```

Output

05 04 03 02 01

Steps to display output

tasm ds.asm

tlink ds.obj

ds

```
jbe i4
add dl, 07h
i4: add dl, 30h
mov ah, 02h
int 21h
dec ch
jnz i2
mov dl, ' '
mov ah, 02h
int 21h
dec dh
jnz print
mov ah, 4ch
int 21h
end
```

Conclusion :- Hence we sorted an array in ascending and descending order.