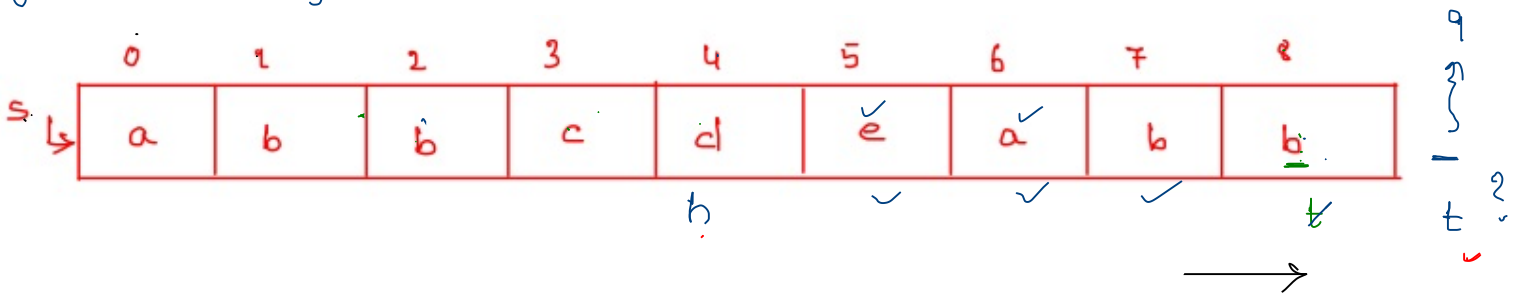
Sortings

longest substring : no duplicates




```
function fun(s,n)
{
        head=0
        map={}
        res=0
        for(tail=0;tail<n;tail++)
        {
                key=s[tail]
                while(key in map and map[key]>0)
                {
                        1. res=max(res,tail-head)
                        2. map[s[head]]=map[s[head]]-1
                        3. head++
                }
                if key in map
                {
                        map[key]++
                }
                else
                {
                        map[key]=1
                }
        }
        retrun max(res,tail-head)
}
```

5 ✓

completion ✓

map

key	value
d	1
e	1
a	1
b	1

key = d e a b

res = 5, 5

map[key]>0)

1, 2, ...

>=1

map[key]>=1

By-default, we consider as inc order, / :- Strictly no duplicates.

non-decreasing order.

↳ 2 4 4 5 6 9 9 . . .

1) Bubble sort :-

Ex :- n = 6

i/p

a ↳

0	1	2	3	4	5
9	7	6	4	2	1

↑ order

o/p

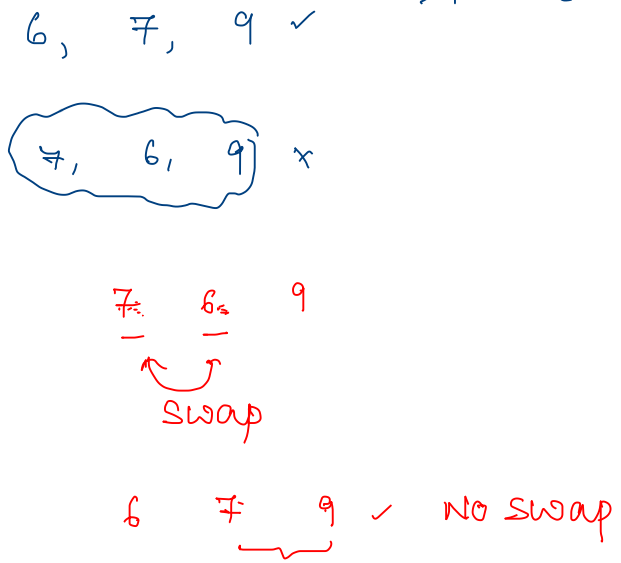
a ↳

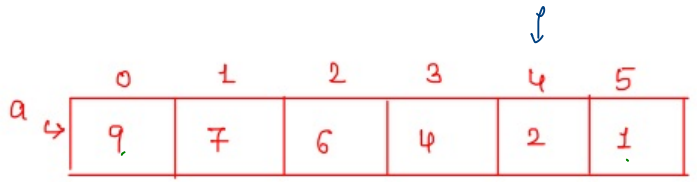
0	1	2	3	4	5
1	2	4	6	7	9

* *

idea : works by repeatedly swapping the adjacent elements if they are not in the proper order

↳ ↑ order





$n = 6$

✓ every pass will start from index-0 (beg)

→ everytime adjacent elements only compared.

Pass-1 :-

$i = 0$

i/p : 9 7 6 4 2 1

7 9 6 4 2 1

6 9 4 2 1

4 9 2 1

2 9 1

1 9

final o/p of P_1 : 7 6 4 2 1 9

$i = 0$, $j = 0$ to 4

$i = 1$, $j = 0$ to 3

$i = 2$, $j = 0$ to 2

Pass2 :- ($i = 1$)

Stop

7 6 4 2 1 9

6 7 4 2 1 9

4 7 2 1 9

2 7 1 9

1 7 9 ✓

o/p of P_2 :- 6 4 2 1 7 9

Pass3 :- ($i = 2$)

→ 6 4 2 1 7 9

H/W

4 2 1 6 7 9

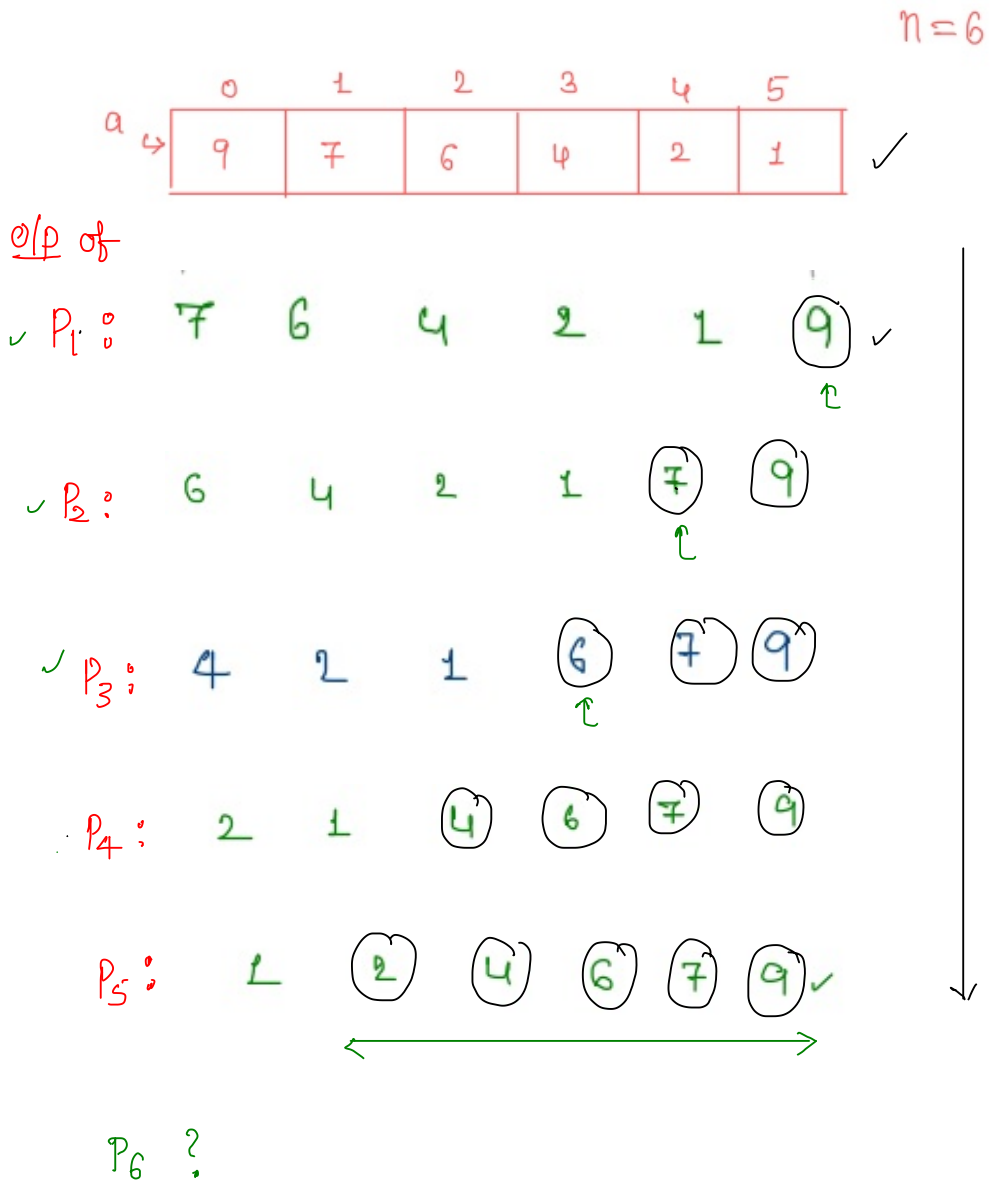
Pass4 :-

$i = 3$ 2 1 4 6 7 9

Pass5 :-

$i = 4$ 1 2 4 6 7 9 ✓

$i = 5$ x



out of 6 elements, if 5 elements in the correct position of final sorted list

which means 6th element automatically in its correct position

so we can say here 5 passes are enough when n=6

so we can say n-1 passes are enough

```
function bubbleSort(arr,n)
{
    for(i=0;i<=n-2;i++) // runs for n-1 times, becasue we need n-1 passes
    {
        for(j=0;j<=n-i-2;j++) // j loop is responsible for with in the pass what happening?
        {
            if(arr[j]>arr[j+1]) // out of order
            {
                temp=arr[j]
                arr[j]=arr[j+1]
                arr[j+1]=temp
            }
        }
    }

}
```

including when n=6

i=0, j=0 to 4 ✓ 6-0-2=4

i=1, j=0 to 3 6-1-2=3

i=2, j=0 to 2 6-2-2=2

.

.

≤ n-i-2

```
function bubbleSort(arr,n)
{
	for(i=0;i<=n-2;i++) // runs for n-1 times, becasue we need n-1 passes
	{
		for(j=0;j<n-i-1;j++) // j loop is responsible for with in the pass what happening?
		{
			if(arr[j]>arr[j+1]) // out of order
			{
				temp=arr[j]
				arr[j]=arr[j+1]
				arr[j+1]=temp
			}
		}
	}

}
```

```
function bubbleSort(arr,n)
{
    for(i=0;i<n-1;i++)
    {
        for(j=0;j<=n-i-2;j++)
        {
            if(arr[j]>arr[j+1])
            {
                temp=arr[j]
                arr[j]=arr[j+1]
                arr[j+1]=temp
            }
        }
    }

}
```

$n=6$ ✓

$j \le 4$

?

		# of comp's	# of swap's	
P_1	$i=0$	$j=0\ 1\ 2\ 3\ 4\ 5$	5	✓ ✓ ✓ ✓ ✓ 5
P_2	$i=1$		4	4
P_3	$i=2$		3	3
P_4	$i=3$		2	2
P_5	$i=4$		1	1

5 passes

$n=6$, total comp's : $1+2+3+4+5$

total swaps : $1+2+3+4+5$

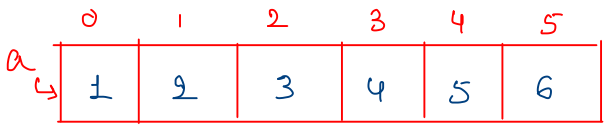
in general : $1+2+3+\cdots+n-1 = \frac{n(n-1)}{2}$ ✓ total # of comp's same as swaps

$1+2+3+\cdots+n-1+n = \frac{n(n+1)}{2}$

$n \rightarrow$ replace : $n-1$

$\therefore O(n^2)$ ✓

i/p




```
function bubbleSort(arr,n)
{
        for(i=0;i<n-1;i++)
        {
                for(j=0;j<=n-i-2;j++)
                {
                        if(arr[j]>arr[j+1])
                        {
                                temp=arr[j]
                                arr[j]=arr[j+1]
                                arr[j+1]=temp
                        }
                }
        }

}
```

total comp's = 15

total swaps = 0 ✓

in general -

$\frac{n(n-1)}{2}$

0

T.C: $O(n^2)$ ✓

```
function modifiedBubbleSort(arr,n)
{
        for(i=0;i<n-1;i++) // pass=1
        {
                falg=0
                for(j=0;j<=n-i-2;j++) // sorted
                {
                        if(arr[j]>arr[j+1])
                        {
                                temp=arr[j]
                                arr[j]=arr[j+1]
                                arr[j+1]=temp
                                flag=1
                        }
                }
                if(flag==0) // swap not happened
                {
                        break
                }
        }

}
```

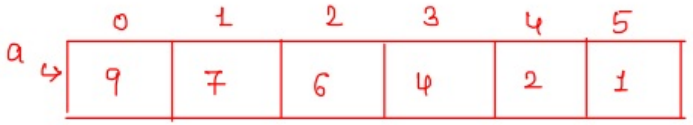
i/p : already ↑ order : TC : O(n)

i/p : o/w : $O(n^2)$

selection sort

↳ next class (Friday)

$n = 6$



```
function selectionSort(arr[], n)
{
        for( i=0;i<n-1;i++)
        {
                 min_index=i;
                for(j=i+1;j<n;j++)
                {
                        if(arr[j]<arr[min_index])
                        {
                                min_index=j;
                        }
                }
                if(min_index!=i)
                {
                        temp=arr[min_index];
                        arr[min_index]=arr[i];
                        arr[i]=temp;
                }
        }
}
```