Page No.:
Date : / /

Lecture 21

* Longest prefix suffix :

Better Approach : O(N)

str = a b c d a b c e a b c d a b c d a b

a b c d a b c e a b c d a b c d a b

First start index from 1 for the string.

and we need two pointer :

1st : at 0th index

2nd : at 2nd index.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
a	b	c	d	a	b	c	e	a	b	c	d	a	b	c	d	a	b	

On this we move 2nd pointer only front side
↳ dont move back side.

First pointer Move both side.

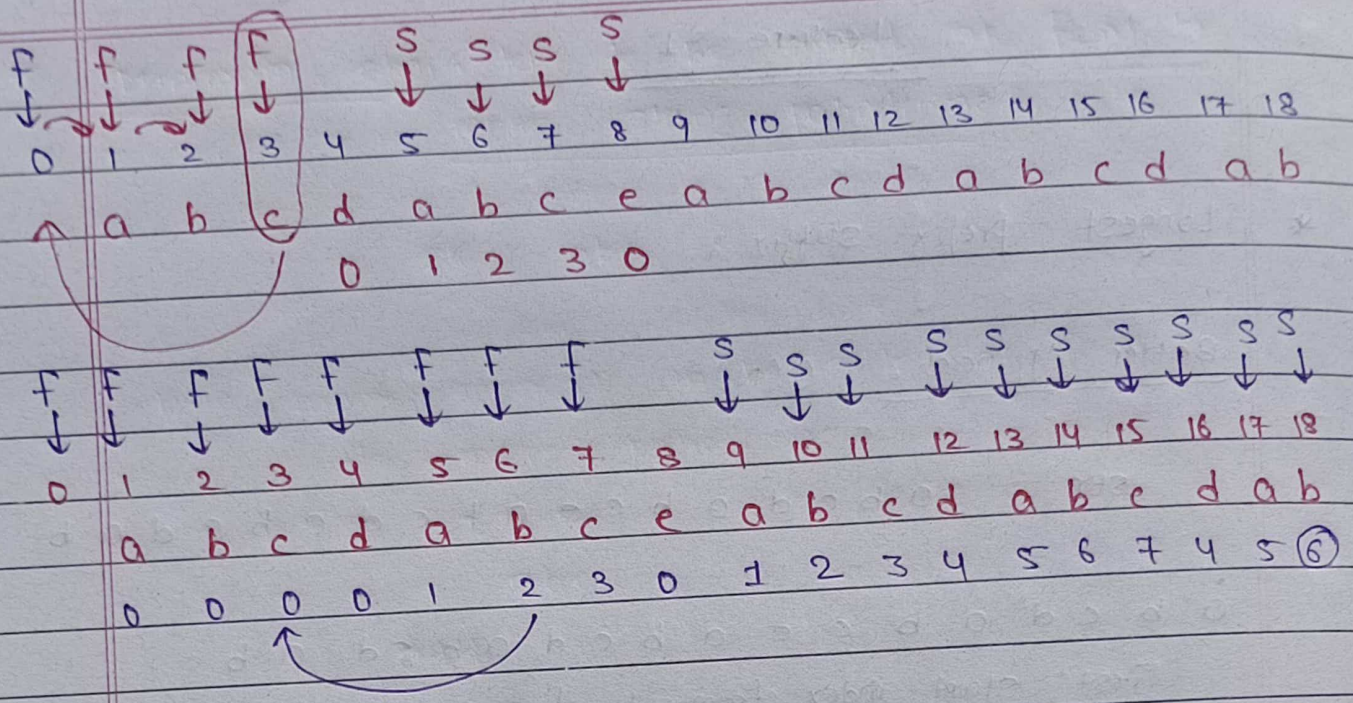
first ↓ second ↓

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
a	b	c	d	a	b	c	e	a	b	c	d	a	b	c	d	a	b	
Lps	0	0	0	0	1													

Now, Both First & second increase

First + 1 = second
↳ False (print 0).

Teacher's Signature.....




```
int LPS[s.size() + 1];
char str[s.size() + 1];

for (int i=0; i<s.size(); i++) {
        str[i+1] = s[i];
        LPS[i] = 0;
}

LPS[s.size()] = 0
int first = 0, second = 2;

While ( second <= s.size()) {
    if ( str[first + 1] = str[second]){
        LPS[second] = first + 1;
        first++;  second++;
    }
else{
    if ( first == 0)
        second++;
else
```

Teacher's Signature.....

```
            first = lps[first];
        }
    }
    return lps[s.size()];
}
```

* **Check if string is rotated by two places:**

Str1 : amazon

Str2 : anamaz

tell that How many time string 1 rotate to make Str2.

Str1 & Str2.

```
      a m a z on  &  a n a m a z
lps : 0 0 1 0 0 0  0  0 0 1 2 3 4
```

lps = 4

6 - 4 = 2

↳ return.

Teacher's Signature.....

* Minimum character to be added at front to make string palindrome:

str : abc

How many element to be added at front to make palindrome.

abc

add

cb

c b a b c

↳ 2 element.

Example 2:

str1 : tictk

str2 : reverse of str1 : ktcit

str1 $ str2

t i c t k $ k t c i t

Lps: 0 0 0 1 0 0 0 1 0 0 1

5 - 1 = 4

Teacher's Signature.....

Page No.:
Date: / /

Example:

Str1 = a a c e c a a a

Str2 = a a a c e c a a

↙

Reverse of Str1

Str1 & Str2

a	a	c	e	c	a	a	a	$	a	a	a	c	e	c	a	a
0	1	0	0	0	1	2	2	0	1	2	2	3	4	5	6	7

8 - 7 = (1)

Code:

```
int index [2*(S.size()+1)];
char str [2*(s.size()+1)];
int n = 2*(S.size()+1);

for(int i=0; i<size(); i++){
    str[i+1] = str[n-i-1] = s[i];
    index[i] = index[i+s.size()] = 0;

    index[n-1] = 0;
    Str[s.size()+1] = '$';
    int first=0, second=2;
  While( second <n){
    if(str[first+1] == str[second]){
        index[second] = first+1;
        first++;
        second++;
    }
```

Teacher's Signature.....

```
else {
    if ( first == 0)
        second++;
    else
        first = index [first];
    }
}
return  s.size() - index [str.size()];
}
```

Teacher's Signature.....