Todays Content

- → Calculate Lps[]
- → Optimization Lps[]
- → Period of a string

LPS : longest Prefix, which is suffix, neglect complete string

LPS[] : LPS value of all substrings starting at index 0

Ex1:

	0	1	2	3	4	5	6	7
S =	c	a	c	y	c	a	c	a
Lps[ ] =	0	0	1	0	1	2	3	2

→ To Calculate lps[] TC: O(N) SC: O(N) ?

// Calculating lps[] =

Obs1: Given S of len N & assume lps[i] = 5

S_N : $S_0\ S_1\ S_2\ S_3\ S_4\ S_5 \cdot\cdot\ S_{i-6}\ S_{i-5}\ S_{i-4}\ S_{i-3}\ S_{i-2}\ S_{i-1}\ S_i \cdots S_{N-1}$

lps[] : _ _ _ _ _   _ _ _ _ _ 5

: Lps[i] = 5

→ $S_0\ S_1\ S_2\ S_3\ \cancel{S_4} == S_{i-4}\ S_{i-3}\ S_{i-2}\ S_{i-1}\ \cancel{S_i}$

$S_0\ S_1\ S_2\ S_3 == S_{i-4}\ S_{i-3}\ S_{i-2}\ S_{i-1}$

Lps[i-1] = 4

Is it possible $S_0 S_1 S_2 S_3 S_4 == S_{i-5}\ S_{i-4}\ S_{i-3}\ S_{i-2}\ S_{i-1}$

↳ lps[i-1] >= 4

//Generalize:

Say lps[i] = x

: lps[i-1] >= $x-1$

: lps[i-1] >= lps[i] - 1

: lps[i-1] + 1 >= lps[i]

: lps[i] <= lps[i-1] + 1

If lps[i-1] = a, lps[i] <= a+1

Obs: lps value, if increasing it can at max increase by 1.

Step: 2

Exn1:

	0	1	2	3	4	5	6	7
S =	a	b	a	y	a	b	a	ch
lps[] =	0	0	1	0	1	2	3	4?

unknown

If ch==y,

lps[7]=4

Exn2:

	0	1	2	3	4	5	6	7	8	9
S =	b	c	a	d	c	b	c	a	d	ch
Lps[]	0	0	0	0	0	1	2	3	4	5

ch==c

lps[9]=5

// Generalize :

S_N =	S_0	S_1	S_2	...	S_{n-1}	S_n	S_{n+1}	.....	S_{i-n}	S_{i-n+1}	S_{i-n+2}	.. S_{i-1}	S_i
Lps[] =												n	

$n+1$:

$S_i == S_n$

// Say Calculate lps[i] ?

```
n = lps[i-1]
if( S[i]== S[n] ){
    lps[i] = n+1
}
```

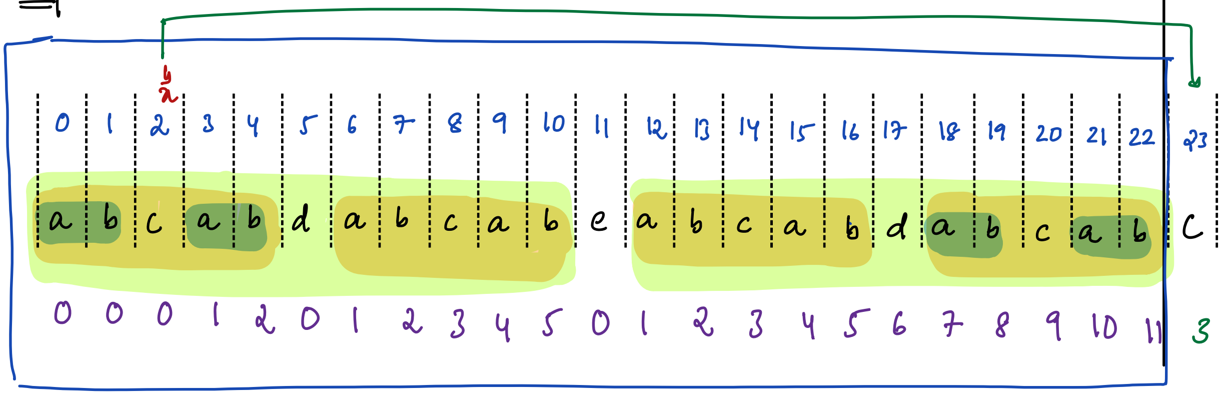
Step: 3

								n									
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
S =	c	a	c	y	c	a	c	a	b	c	a	c	y	c	a	c	y
lps[] =	0	0	1	0	1	2	3	2	0	1	2	3	4	5	6	7	4

i = 16, n = lps[i-1] = lps[15] = 7

n	S[i] == S[n]	
7	S[16] == S[7]	× n = lps[n-1] n = lps[6] = 3
3	S[16] == S[3]	lps[i] = n+1 = 4

Step4:

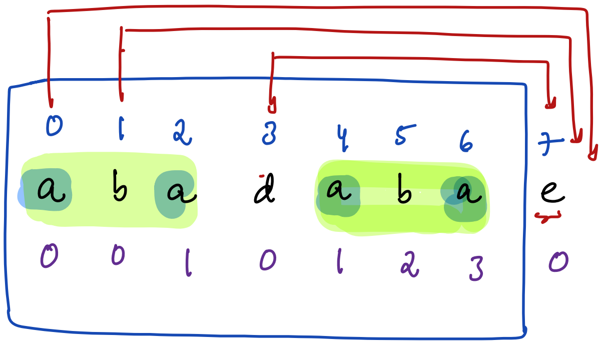


0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
a	b	c	a	b	d	a	b	c	a	b	e	a	b	c	a	b	d	a	b	c	a	b	c
0	0	0	1	2	0	1	2	3	4	5	0	1	2	3	4	5	6	7	8	9	10	11	3

i = 23, x = lps[i-1] = lps[22] = 11

x	s[i] == s[x]	
11	s[23] == s[11]	× x = lps[x-1] x = lps[10] = 5
5	s[23] == s[5]	× x = lps[x-1] x = lps[4] = 2
2	s[23] == s[2]	✓ lps[i] = x+1 = 3

Step:5



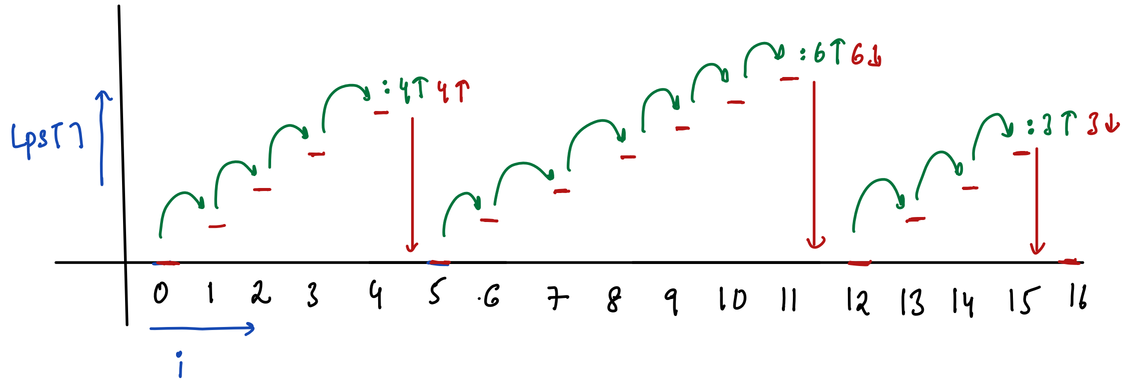
	0	1	2	3	4	5	6	7
S :	a	b	a	d	a	b	a	e
lps[] :	0	0	1	0	1	2	3	0

i = 7, x = lps[i-1] = 3

x	s[i] == s[x]	×
3	s[7] == s[3]	× x = lps[x-1] x = lps[2] = 1
1	s[7] == s[1]	× x = lps[x-1] x = lps[0] = 0
0	s[7] == s[0]	× if(x==0){ come out of loop} x = lps[x-1] ⇒ x = lps[-1]

```
int[] CreateLps(String s) {
    int n = s.size()
    int lps[n]
    lps[0] = 0;
    i=1; i<n; i++) {
        We need calulate lps[i] ?
        int x = lps[i-1]
        while( s[i] != s[x] ) {    s[i] != s[0]
            if (x == 0) { x = -1, break}
            x = lps[x-1]  // if x==0 avoid update
        }
        lps[i] = x+1
    }
    return lps[]
}
```

Obs: If lps value increases, it will increase by 1, Only 1 iteration



Obs1 : Given N, At max how many Inc iterate = N } Total iter = N+N = O(N)

↳ At max total dec iterations = N

Ex:

	0	1	2	3	4	5	6
	a	a	a	a	a	a	d

lps[] $0 \xrightarrow{+1} 1 \xrightarrow{+1} 2 \xrightarrow{+1} 3 \xrightarrow{+1} 4 \xrightarrow{+1} 5 \quad \underline{\underline{x}}$

$i = 6$, $x = lps[i-1] = lps[5] = 5$

x	$S[i] == S[x]$	
5	$S[6] == S[5]$	* $x = lps[x-1]$ $x = lps[4] = 4$
4	$S[6] == S[4]$	* $x = lps[x-1]$ $x = lps[3] = 3$
3	$S[6] == S[3]$	x $x = lps[x-1]$ $x = lps[2] = 2$
2	$S[6] == S[2]$	* $x = lps[x-1]$ $x = lps[1] = 1$
1	$S[6] == S[1]$	* $x = lps[x-1]$ $x = lps[0] = 0$
0	$S[6] == S[0]$	* $x = -1$ break

Period of String

Given S_N, period is defined as minimum n such that

$$\boxed{\forall_{i=0}^{N-n-1} S_i = S_{i+n}}$$

meaning

$S_0 = S_n$ $S_3 = S_{n+3}$

$S_1 = S_{n+1}$ $S_4 = S_{n+4}$

$\vdots$

$S_2 = S_{n+2}$ $S_{N-n-1} = S_{n-1}$

→ For a given String, if there no matching N itself is period ?

Ex1:

0	1	2	3	4	5	6	7	8
a	b	c	a	a	b	c	a	a

} ans=4

$n=1$? $s[0]=s[1]$ ✗

$n=2$: $s[0]=s[2]$ ✗

$n=3$: $s[0]=s[3]$ ✓

$s[1]=s[4]$ ✗

$n=4$:

$S_0 == S_4$ ✓

$S_1 == S_5$ ✓

$S_2 == S_6$ ✓

$S_3 == S_7$ ✓

$S_4 == S_8$ ✓

S_5 = nocomparison

0	1	2
a	b	c

: ans=3

n

$=1$: $s[0]=s[1]$ ✗

2: $s[0]=s[2]$ ✗

3: $s[0]=s[3]$ nocomparison

Ex2:

0	1	2	3	4
a	b	c	a	b

ans=3

n

1 : $s[0]==s[1]$ ✗

2 : $s[0]==s[2]$ ✗

3 : $s[0]==s[3]$ ✓

$s[1]==s[4]$ ✓

$s[2]==s[5]$ Stop

Idea: TC: $O(N^2)$ SC: $O(1)$

For every n check if it's period

```
n=1; n<=N; n++){
    if(check n is period){
        return n
    }
}
```

$S_0 = S_n$

$S_1 = S_{n+1}$

$S_2 = S_{n+2}$

$\vdots$

$S_{n-n-1} = S_{n-1}$

TC: $O(N)$

// Say S_N : $\boxed{\text{period} = n}$

$S_0 \quad S_1 \quad S_2 \quad S_3 \; .. \; S_n \quad S_{n+1} \quad S_{n+2} \; .... \; S_{N-n-1} \quad S_{N-n} \; ...... \; S_{N-1}$

$S_0 \quad S_1 \quad S_2 \quad S_3 \; .. \; S_n \quad S_{n+1} \quad S_{n+2} \; .. \; S_{n+3} \; ... \; S_{2n} \quad S_{2n+1} \quad S_{2n+2} \; ...... \; S_{N-1}$

$S_{[0, N-n-1]} = S_{[n, N-1]}$: length of string $[\overset{a}{n}, \overset{b}{N-1}] = \overset{b-a+1}{N-\cancel{1}-n+\cancel{1}}$

$\boxed{\text{Lps of String : } N-n}$

$\Rightarrow$ period$(S_N) = n$

Lps$(S_N) = N-n$

period(S_N) + Lps$(S_N) = N$

$\boxed{\text{period}(S_N) = N - \text{Lps}(S_N)}$

Note: Lps of Complete String = lps[N-1]

period$(S_N) = N - (\text{Lps}[N-1])$

→ Class & Object basics

→ LinkedList basics

} Wednesday → Thursday

Doubts:

→ KMP : pattern matching with lps[] , knult morris pratt

→ z Algo : → [] ✗

→ Rabin Karp : Heavy math

heavy code

Doubts:

A: a b a d e c

B: a b c

Testcases

{TA}

$$\underset{N}{\overset{s_1}{hs_1}}, \quad \boxed{\underset{M}{\overset{s_2}{hs_2}} = \text{filled}}$$

s=0, e=0, hs1, hs2 = filled

```
while( e<=N ){
    if( hs1 contain all keys hm2 ){
        ans = Max( ans, e-s )
        remove s1[s] from hm1 , s = s+1
    }
    else
        if( e == N ){ break }
        add s1[e] in hm1
    }
        e = e+1
}
return ans
```