

String = "abacbb"

Pattern = "bac"

Brute force = $O(MN)$

Many algorithms to solve it optimally $\approx O(N+M)$

- 1) Rabin-Karp =
 - 2) KMP =
 - 3) Z-algorithm =
 - 4) etc
- \searrow
 \swarrow
 $O(N+M)$

1) Prefix of a string.

$S \Rightarrow$ "abcda"
0 1 2 3 4

Prefix \Rightarrow a, ab, abc, abcd and abcda

\Rightarrow Substring starting from 0th index.

2) Suffix of a string.

$S =$ "abcda"
0 1 2 3 4

Suffix = a, da, cda, bcda, abcda.

Suffix is a substring which ends at $(n-1)^{th}$ index.

3) LPS of a string \Rightarrow longest prefix of a string which is also a suffix.

\downarrow
longest prefix
suffix

$S \Rightarrow$ abcab

LPS(S) \Rightarrow 2

<u>Prefix</u>	<u>Suffix</u>
a	b \times
ab	ab \checkmark
abc	cab \times
abca	bca \times
abcab	abcab \times

Ex $S = abcd$

$LPS(S) \neq 0$

Prefix	Suffix
a	d ✗
ab	cd ✗
abc	bcd ✗
abcd	abcd

Ex $S = aaaaa$

$LPS(S) = 4$

Prefix	Suffix
a	a ✓
aa	aa ✓
aaa	aaa ✓
aaaa	aaaa ✓
aaaaa	aaaaa

Note: To calculate the LPS we don't consider the full string.

Brute force to find lps of a string !

$S = a_1 a_2 a_3 a_4$
 $l \Rightarrow 2$
 p_1 p_2

$[1, n-1]$

mid

Prefix	Suffix
a_1	a_4
$a_1 a_2$	$a_3 a_4$
$a_1 a_2 a_3$	$a_2 a_3 a_4$

Pseudo Code

$lps \Rightarrow [1, n-1]$

int ans_lps $\Rightarrow 0$;

for (int lps $\Rightarrow 1$; lps $< n$; lps++) {

int p1 $\Rightarrow 0$; int p2 $\Rightarrow n - lps$;

while (p1 $< lps$) {

if (s[p1] == s[p2])

p1++, p2++;

else

break;

}

if (p1 == lps)

ans_lps = lps

}

$S =$
^{0 1 2 3 4}
 ' a b c a b '
 \uparrow \uparrow
 P_1 P_2

lps	prefix	suffix
1	a	b
2	ab	ab ✓
3	abc	a cab
4	abca	bcab

LPS Array : Given a string of length n , lps array would also contain n elements where $\text{lps}[i] \Rightarrow$ lps of the substring from index 0 to index i .
 $S[0 \dots i]$

$S \Rightarrow$ a a b a a b a

$\text{lps} \Rightarrow$

0	1	2	3	4	5	6
0	1	0	1	2	3	4

$\text{lps}[0] \Rightarrow \text{lps}(S[0 \dots 0])$

$\text{lps}[1] \Rightarrow \text{lps}(S[0 \dots 1]) = \text{lps}(aa)$

$\text{lps}[2] \Rightarrow \text{lps}(S[0 \dots 2]) = \text{lps}(aab)$

\Rightarrow

prefix	suffix
a	b
aa	ab

Brute force $\Rightarrow O(n^3)$



Optimized $\Rightarrow O(n)$

Q1

$S \Rightarrow aabacd$

$P \Rightarrow abac$

Find if the pattern exists in the string.

$S' \Rightarrow P + S$

0	1	2	3	4	5	6	7	8	9
a	b	a	c	a	a	b	a	c	d
0	1	2	3	4	5	6	7	8	9
0	0	1	0	1	1	2	3	4	0

if $lps[i] == \text{length of } P$

Tc: $O(N+M)$

there is a pattern
match

Sc: $O(N+M)$

Edge Case : $S = a b c d$.
 $P = a a$

$S' \Rightarrow P + S$
 $\Rightarrow \boxed{a} \boxed{a} a b c d$

0	1	2	3	4	5
0	1	2	0	0	0

$S' \Rightarrow P + \# + S$

↓
character

which is not

present in either part of
main string

$S' \Rightarrow a a \# a b c d$

0	1	0	1	0	0	0
---	---	---	---	---	---	---

$S \Rightarrow a a a a$

$P \Rightarrow a a$

$S' \Rightarrow P + S$

$a a a a a a$

0	1	2	3	4	5
---	---	---	---	---	---

$S' \Rightarrow P + \# + S$

$a a \# a a a a$

0	1	0	1	2	2	2
---	---	---	---	---	---	---

Q2 Given a string S and a pattern, find the number of times pattern occurred in S .

Ex : $S = abababaaab$
 $P = ab$

$S' \Rightarrow P + (\#) + S$

$S' \Rightarrow ab\#abababaaab$

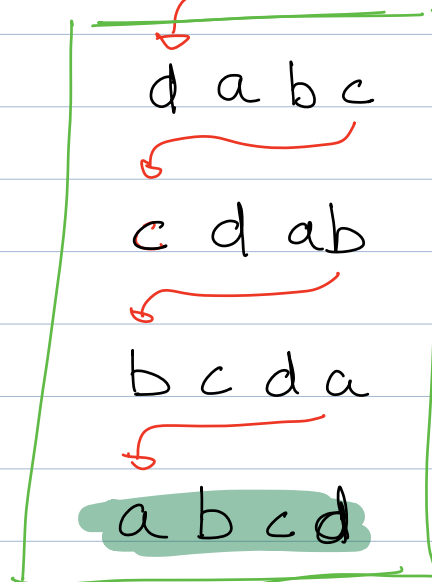
0	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	1	2	1	2	1	2	1	1	1	2

$Tc: O(n+m)$
 $Sc: O(n+m)$

Q₃ Given a string S. Find no of cyclic rotations of S which is equal to S.

Ex

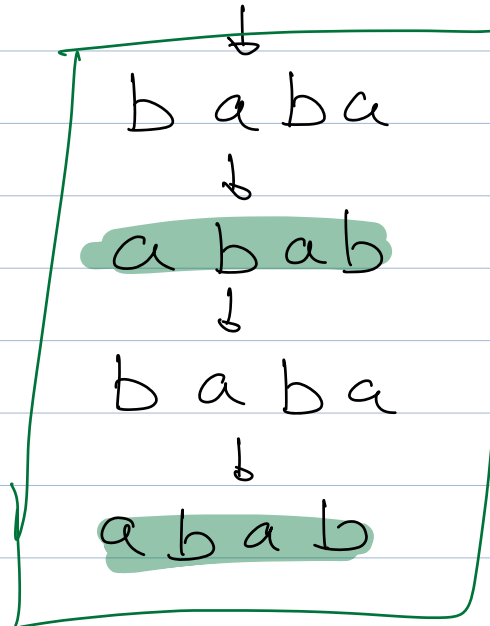
S \Rightarrow a b c d



ans = 1

\Rightarrow N cyclic rotations possible

S \Rightarrow a b a b



ans = 2

$$S' \Rightarrow S + S$$

0 1 2 3 4 5 6 7

abababab \Rightarrow It has all

cyclic rotations
of the original
string as
substring

\Rightarrow Substrings of length.

1) $S[0-3] \Rightarrow$ abab

2) $S[1-4] \Rightarrow$ baba

3) $S[2-5] \Rightarrow$ abab

4) $S[3-6] \Rightarrow$ ababa

5) $S[4-7] \Rightarrow$ abab

$$\# S' \Rightarrow S + S - \left\{ \begin{array}{l} \text{last} \\ \text{character} \end{array} \right\}$$

$S' \Rightarrow$ abababab

$$S'' \Rightarrow S + \# + S'$$

abab # abababab

0	1	2	3	4	5	6	7	8	9	10	11
0	0	1	2	0	1	2	3	4	3	4	3

Q Given a string s . Calculate the min no of characters to add in the beginning to s to make it a palindrome

Ex 1 $dcabacd$

Ex 2 $pedaabaadef$

Ex 3 $edabba de$

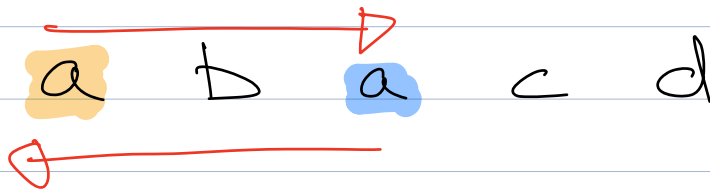
$L \Rightarrow$ length of the longest prefix palindrome.

Ans $\Rightarrow N - L$

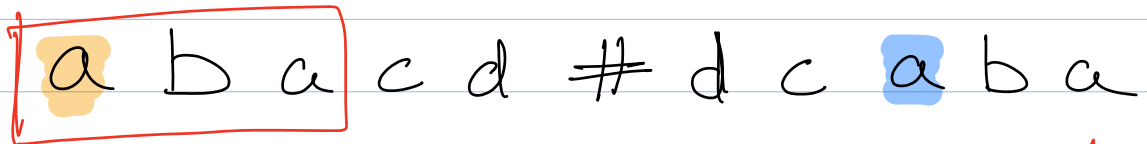
Prefix palindrome \Rightarrow Prefix which is equal to its reverse

Ex

a b a c d



a b a c d # d c a b a



↓
3

Ex

a b a b a

a b a b a # a b a b a

↓
5

$a_1 a_2 a_3 a_4 a_5 \neq a_5 a_4 a_3 a_2 a_1$



$$\begin{aligned} a_1 &= a_4 \\ a_2 &= a_3 \\ a_3 &= a_2 \\ a_4 &= a_1 \end{aligned}$$