Yes , No

100%

String Matching

Advanced Strings

Hashing

KMP and ~~Z Algorithms~~

99.999% → 99.999999

- Priyansh Agarwal

99.9999999999

aabaacd X → n

|baac y → m

$O(m \cdot n)$

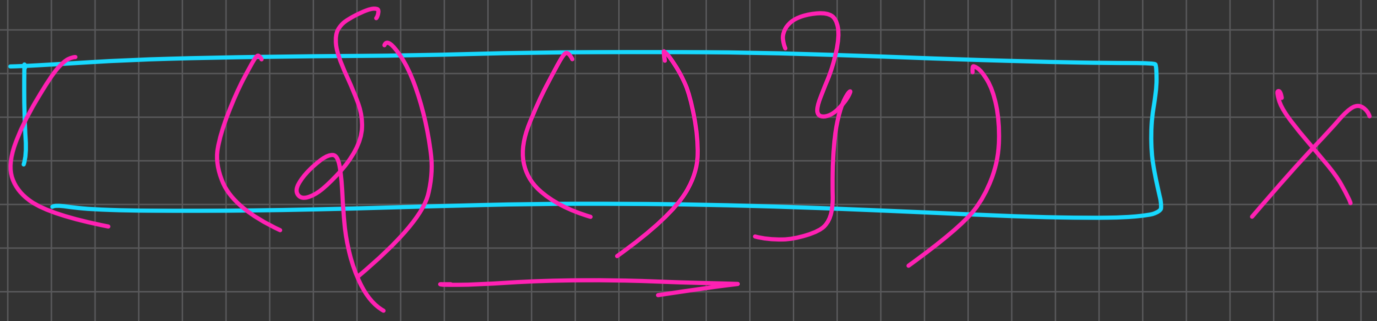
✓ ✓ ✓

aabaabaab X

$O(m \cdot n)$

aab y

S



Y

l to r

(l, r) $O(n)$

SubstringHash$(l, r) \to \log n$, $O(1)$

$O(n \log n)$ $O(n)$

Prefix Function

s

- The prefix function for a string is defined as an array **lps** of length **n** where **lps[i]** = length of longest proper prefix of the substring **[0..i]** which is also a suffix of this substring
- By definition, **lps[0] = 0**
- How to determine this array in **O(n)**?

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

lps → length 9

lps[i]

lps[5]

length = 1

a a b a a

a a b a a b a a

a a b a a

a a b

= a a b

≠ b a a

a a b a a b a a b

a → 0

"a"

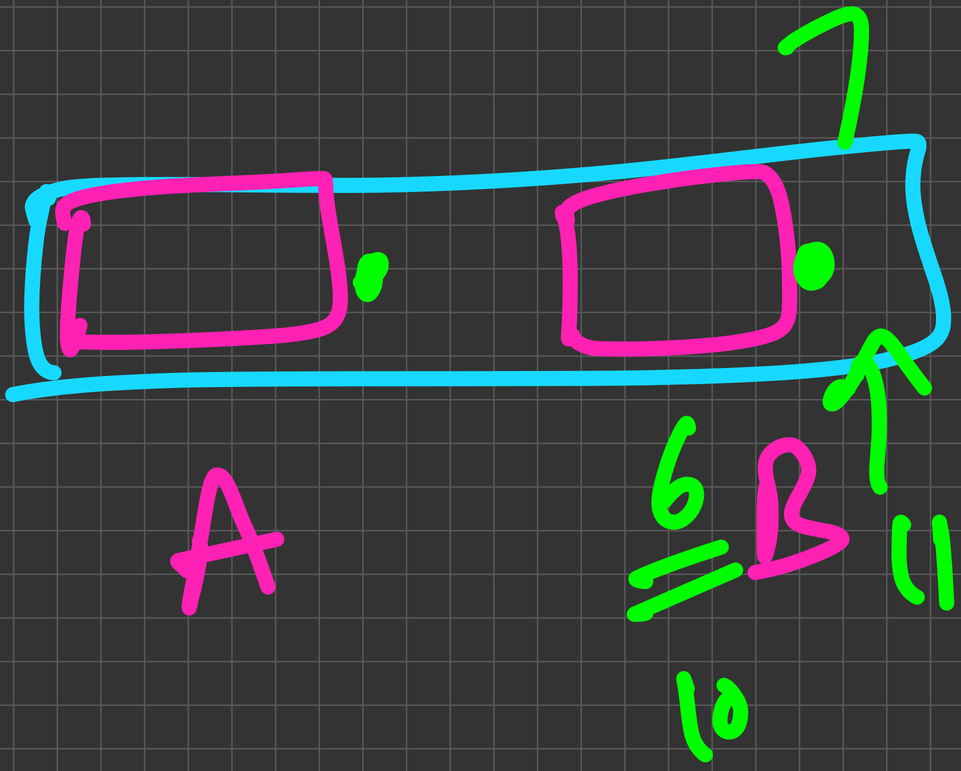
a a → 1

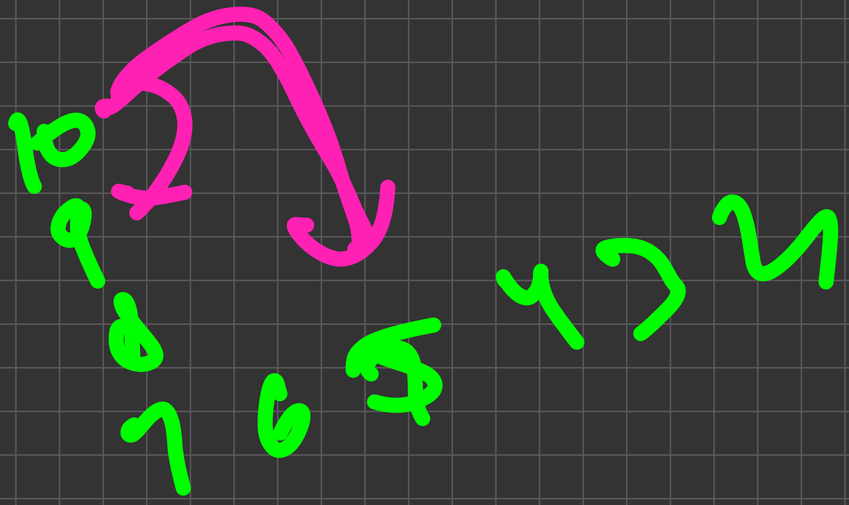
a a b → 0

a a b a → 1

a a b a a → 2

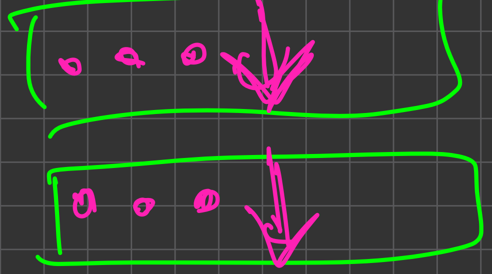
a a b a a b → 3





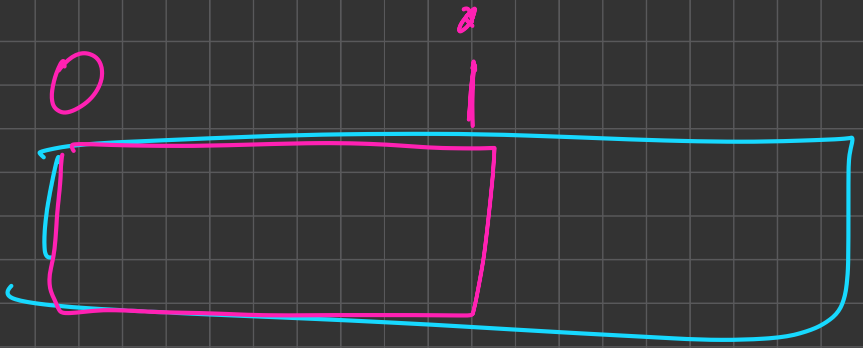
a a b a a c a a b a a d

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



$O(n^3)$

lps array



$O(i \times i) \rightarrow O(n^2)$

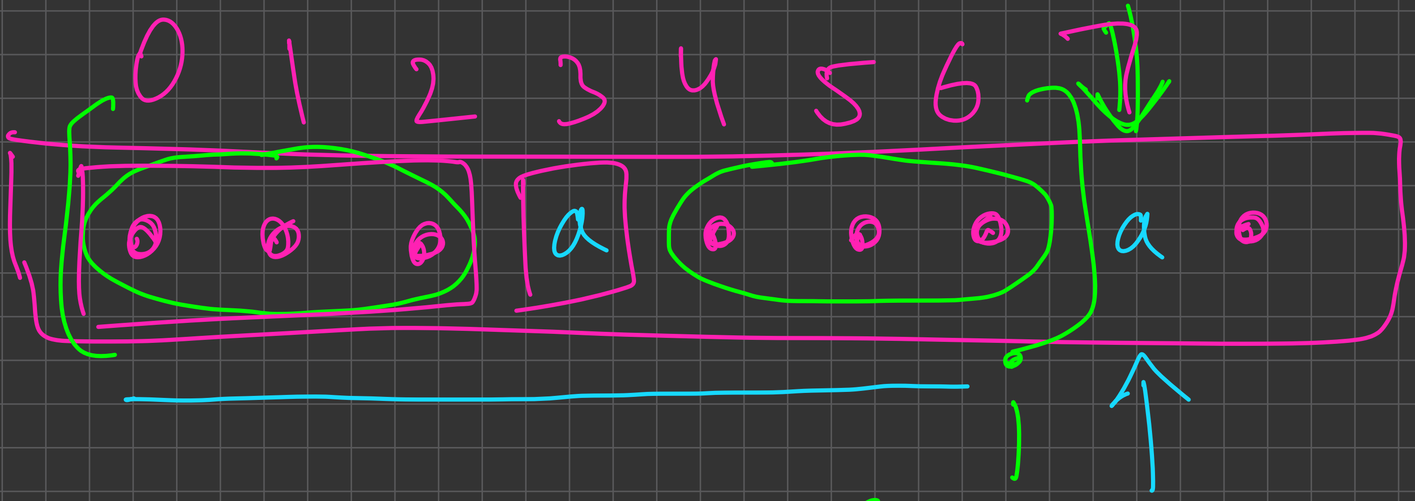
$O(n^3)$

length → 1

length → 2

length → 3

length i

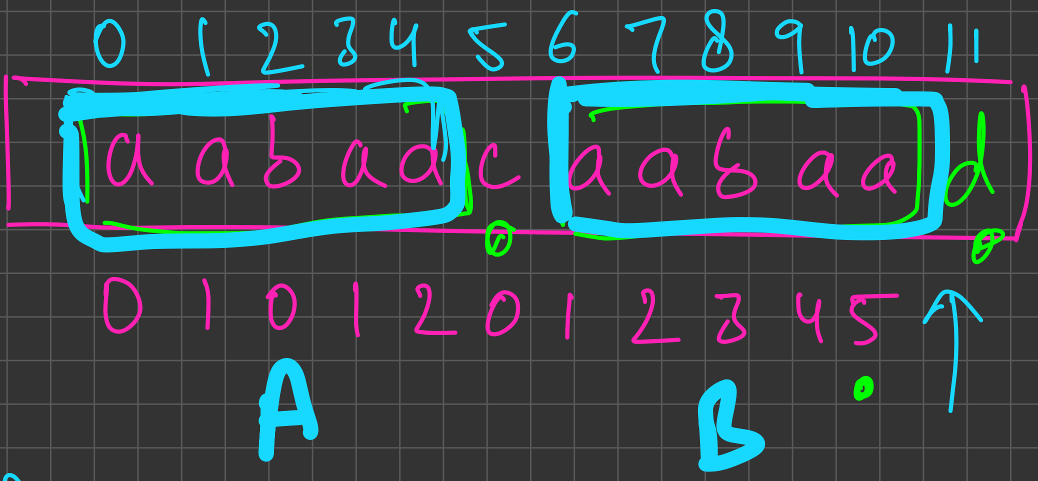


3

```
if ( s[ lps[i-1] ] == s[i] ){
//      lps[i] = lps[i-1] +1
}
```

$s[3] == s[7]$

$lps[7] = lps[6]+1$



0	1	2	3	4	5	6	7	8	9	10	11
a	a	b	a	a	c	a	a	b	a	a	d
0	1	0	1	2	0	1	2	3	4	5	↑

A ==

6

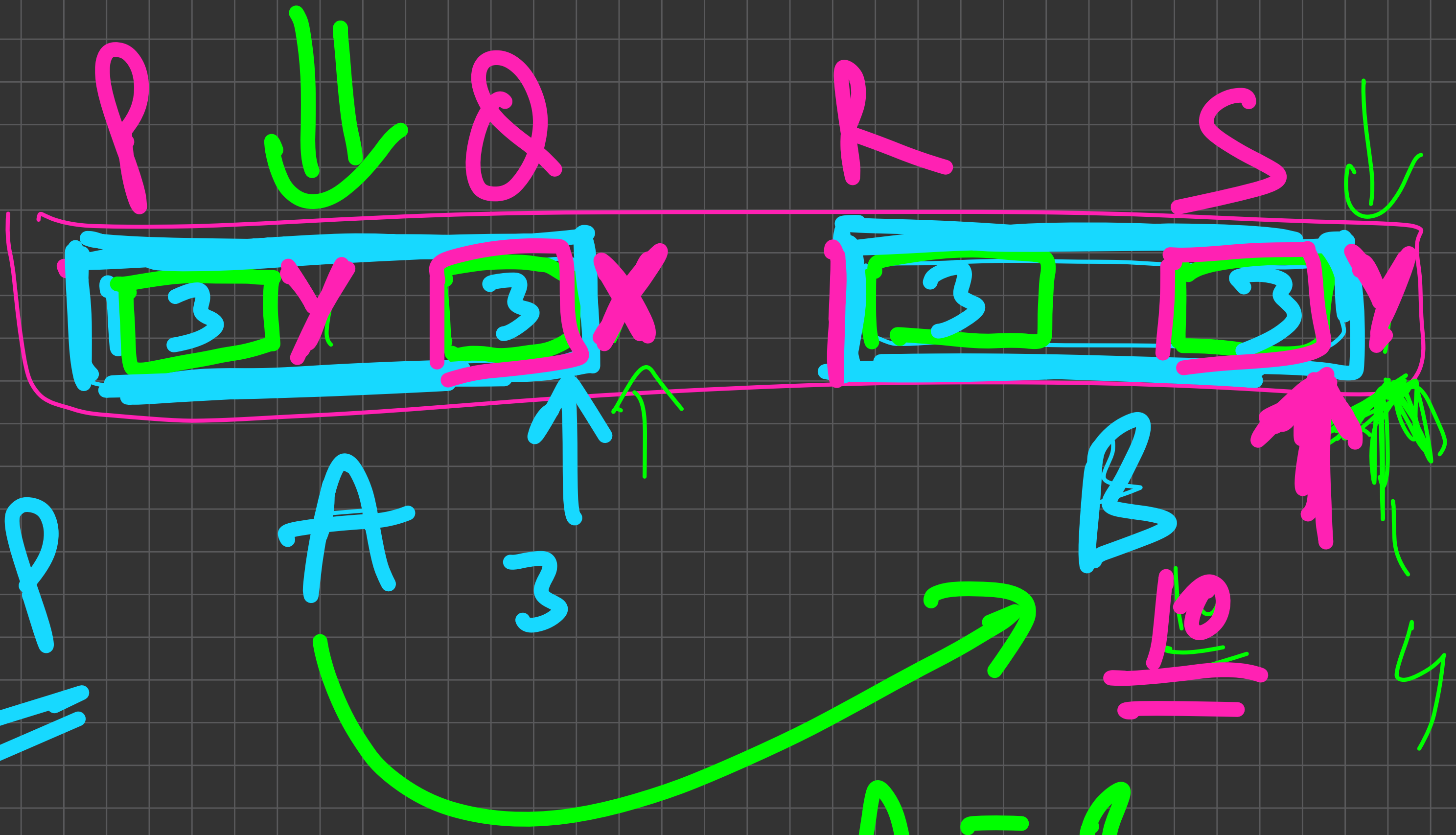
lps(11)

lps[10] → 5

if(s[lps[i-1]] == s[i])

s[5] == s[11] → No

9



s = P

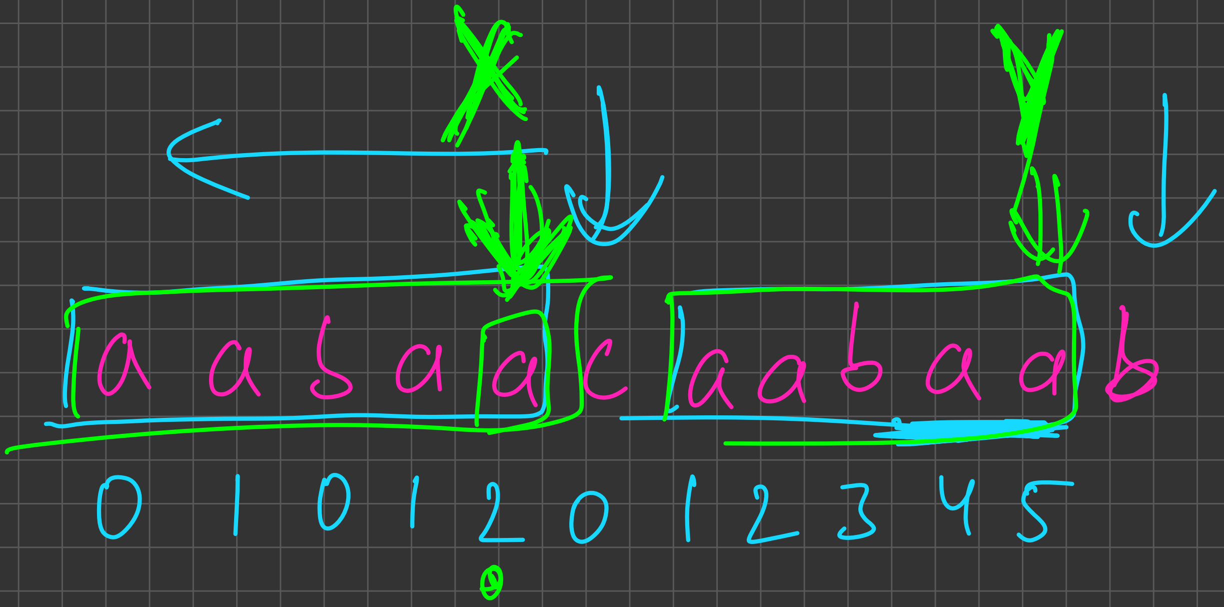
A = B

10 → 11 ?

9

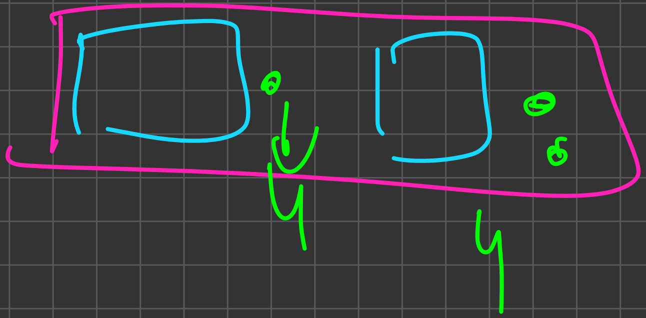
8

7



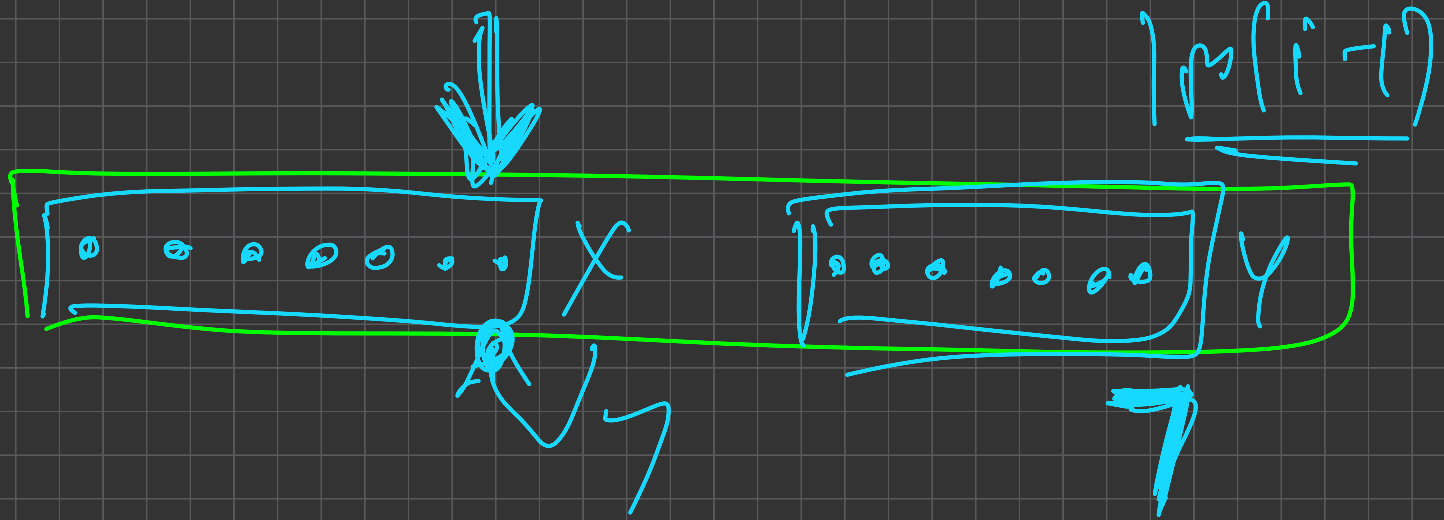
A = B

```
i <- lps[i]
```



```
int j = lps[i-1]
if (s[j] == s[i]) {
    lps[i] = lps[i-1] + 1
}
```

s[y] != s[i]



$$\boxed{\underline{lps[\, lps(i-1) - 1 \,]}}$$

$$\underline{lps[\, lps[\, lps(i-1) - 1 \,] - 1 \,]}$$

```
int j = lps[i-1]

while( j>0 && s[i] != s[j] ){
        j = lps(j-1)
}

j   if(s[i]==s[i])
                          lps[i] = j+1
```

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

a a b a a c a a b a a b

0 1 0 1 2 0 1 2 3 4 5

$j = 5$

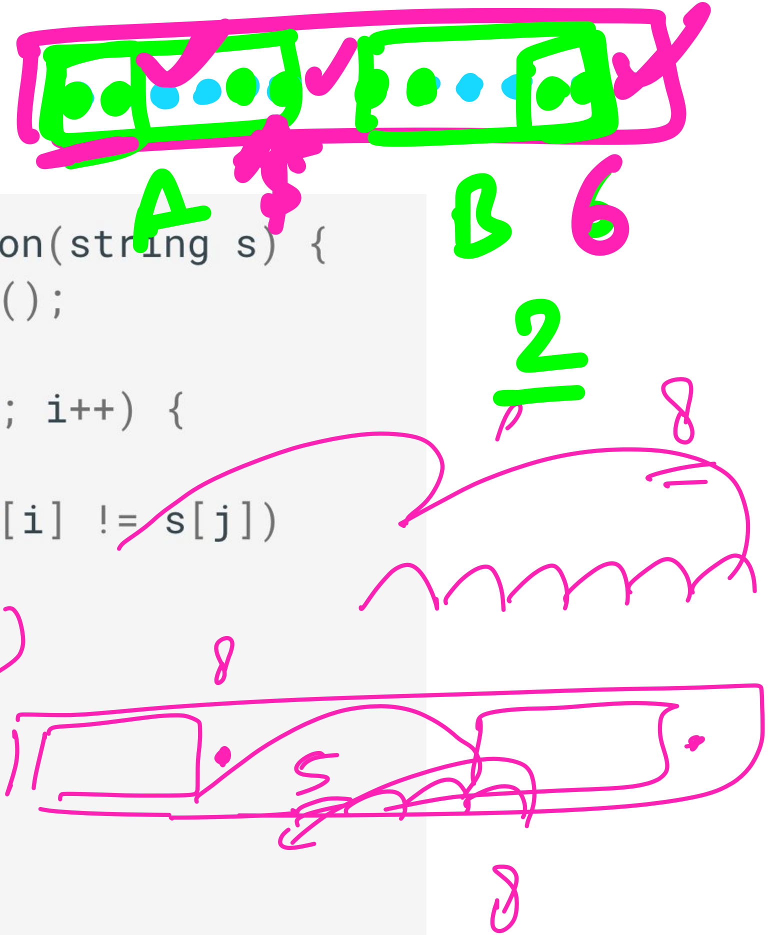
$j = lps(j-1)$

$j = lps(4) = 2$

Prefix Function

```
vector<int> prefix_function(string s) {
    int n = (int)s.length();
    vector<int> pi(n);
    for (int i = 1; i < n; i++) {
        int j = pi[i-1];
        while (j > 0 && s[i] != s[j])
            j = pi[j-1];
        if (s[i] == s[j])
            j++;
        pi[i] = j;
    }
    return pi;
}
```

|p(i)|=c



$b_1 \; , b_2 , \; b_3 \; - - - - - \; b_{\underline{\underline{d}}}$

$b_1 \mid b_2$

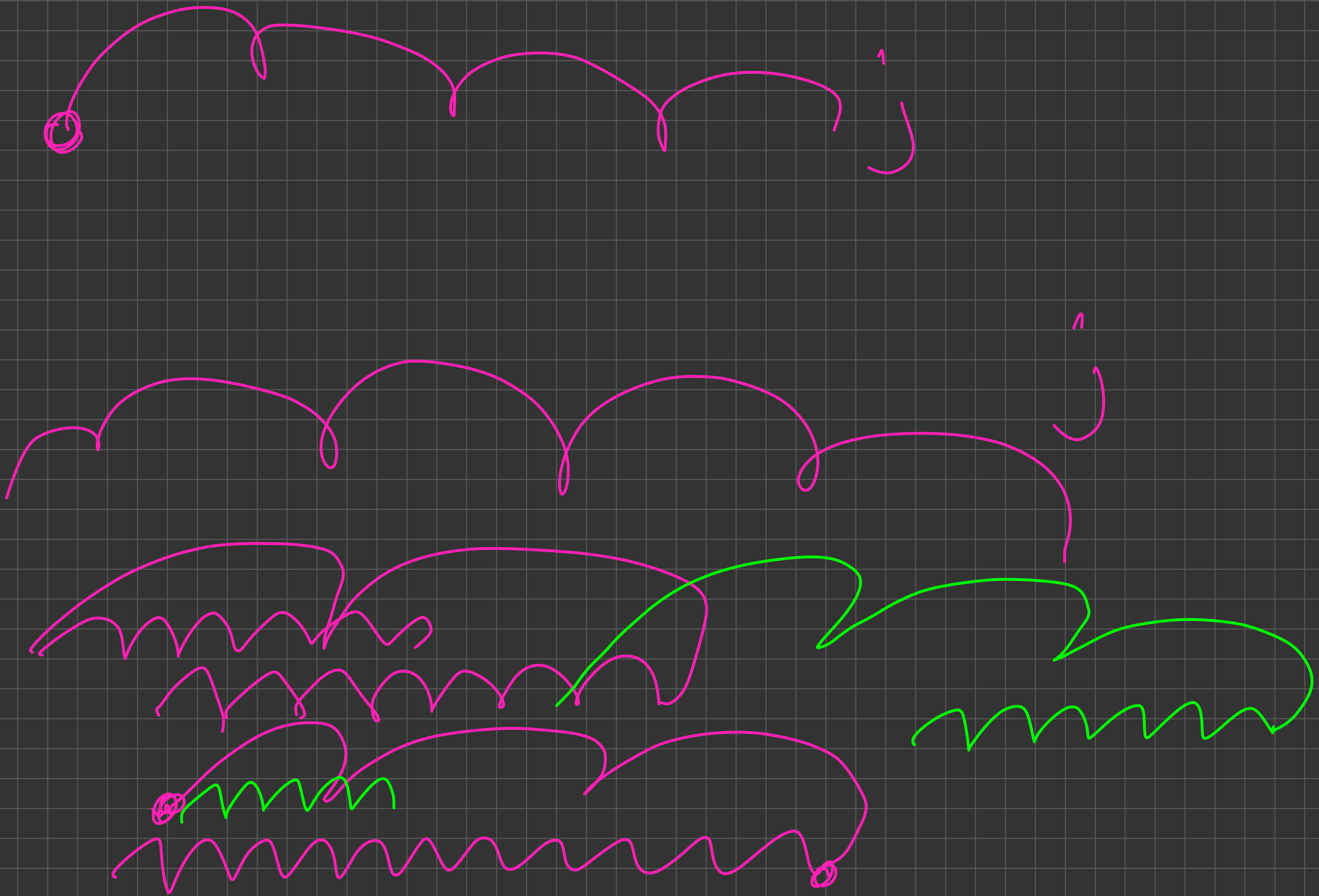
$b_2 \mid b_3$

$\vdots$

$\underline{\underline{(1, n)}}$

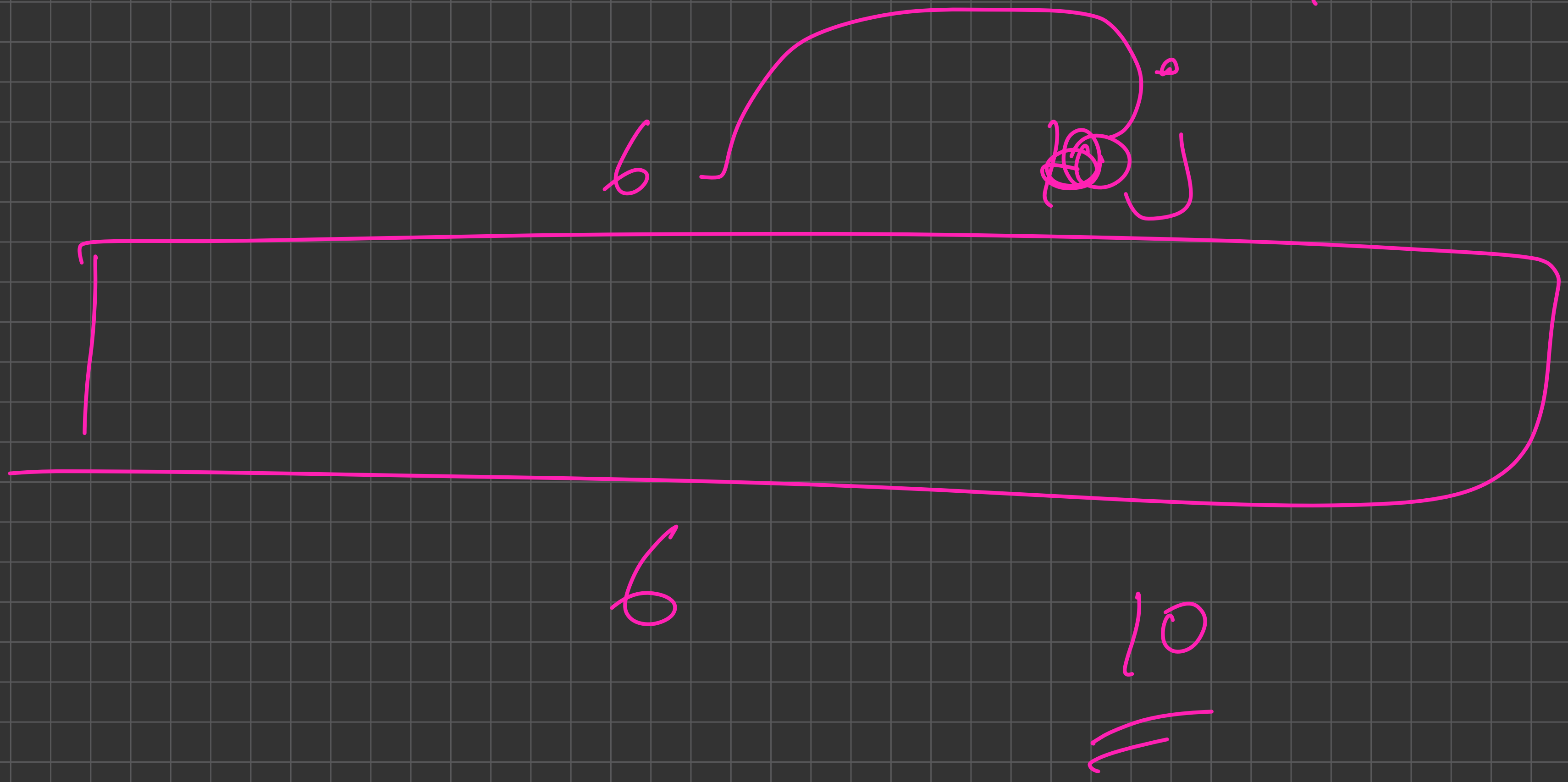
$\underline{\underline{k}}$

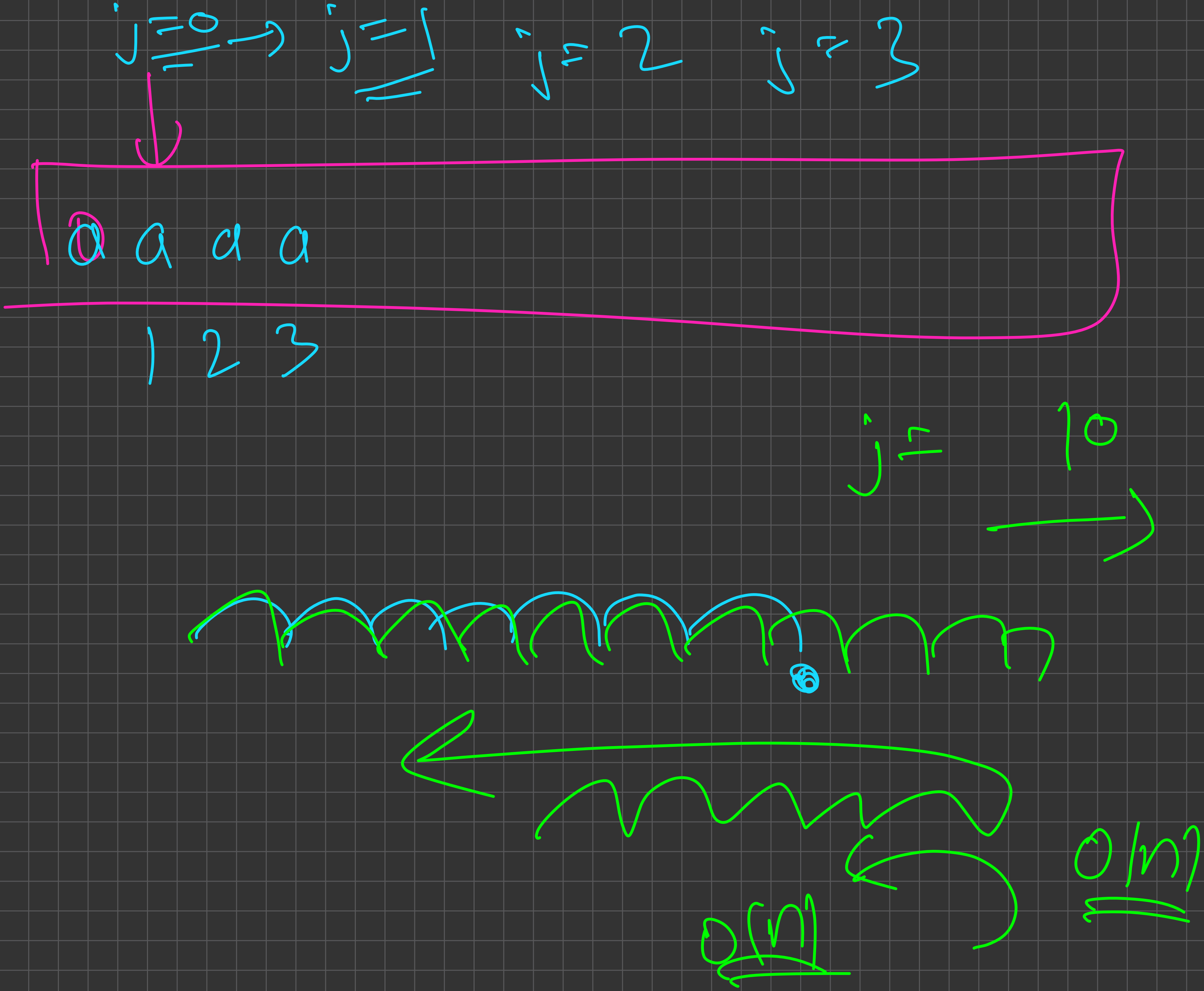
$$2 , \{ 2 , 4 , 6 , 8 , 10 \; \ldots \}$$

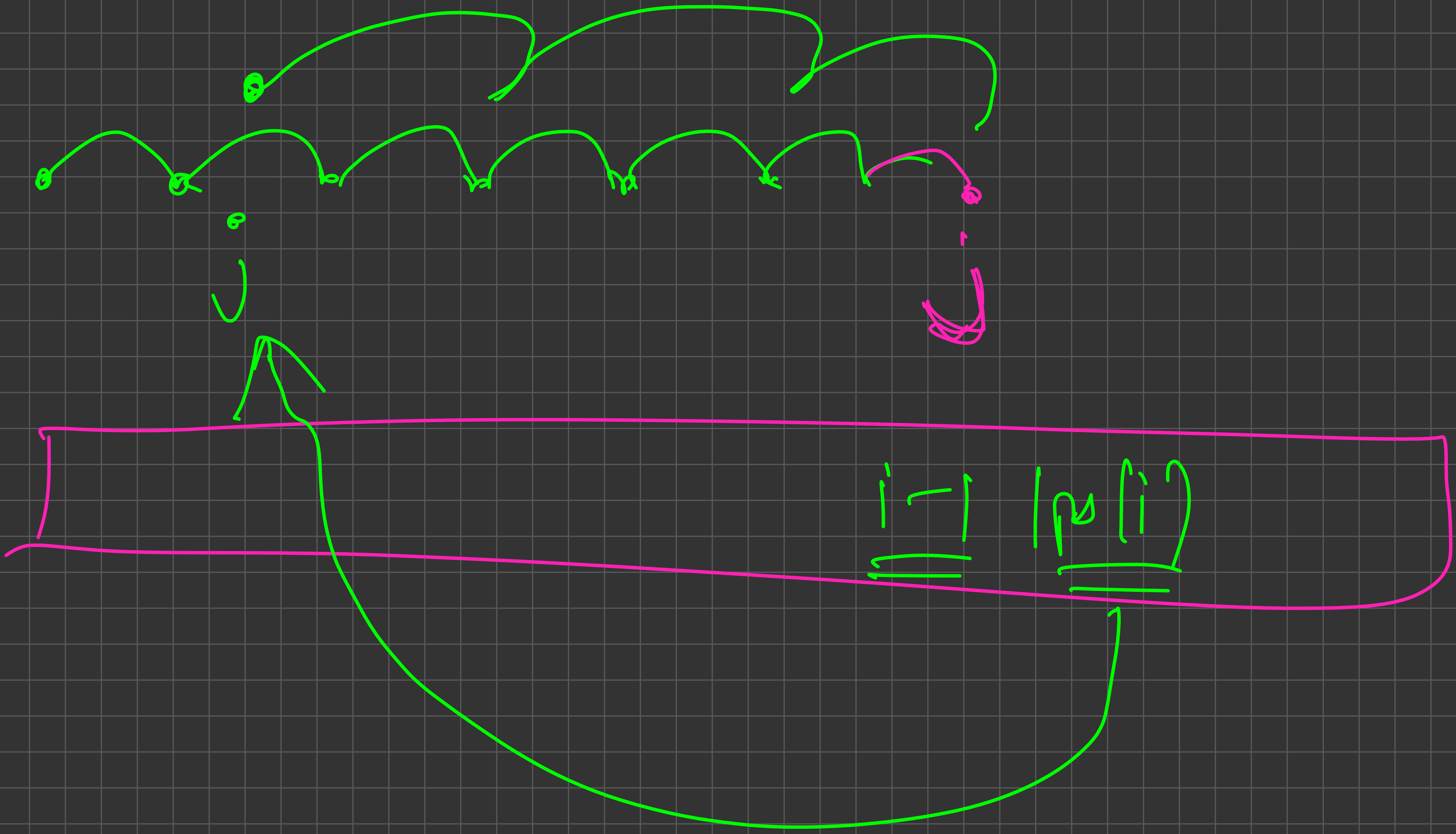


$lis[i] = j+1$

$j = lis[i-1] = 10$




j=0 → j=1
j=2
j=3
a a a a
1 2 3
j= 10
O(n)
O(n)


0
0
j
i
i-1
lps[i]

aab = X = pattern $O(m)$

aabaacaabaad = Y = find

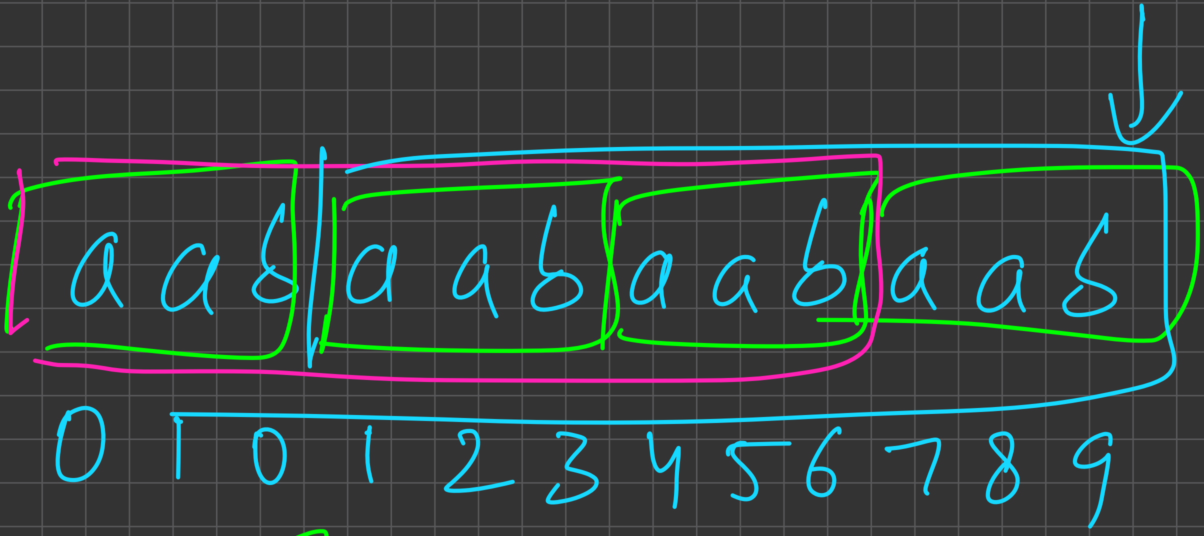
$O(n)$

X + "#" + Y

aab#aabaacaabaad

3 3

$O(m+n)$

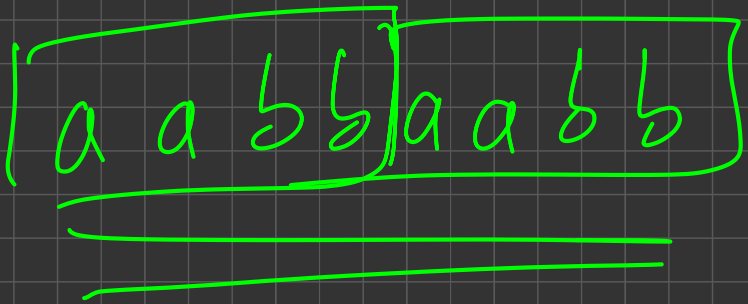


= 3

= 9

lps(n-1)

3



$n / (n - lps(n-1))$

$12/3$

a a b a a a a b a a

0 1 0 1 2 1 2 3 4 5

$lps(n-1) = \underline{\underline{5}}$, $n = 10$

$(n - lps(n-1)) = \underline{\underline{5}}$

$\underline{\underline{10 | 5}} \rightarrow \underline{\underline{5}}$

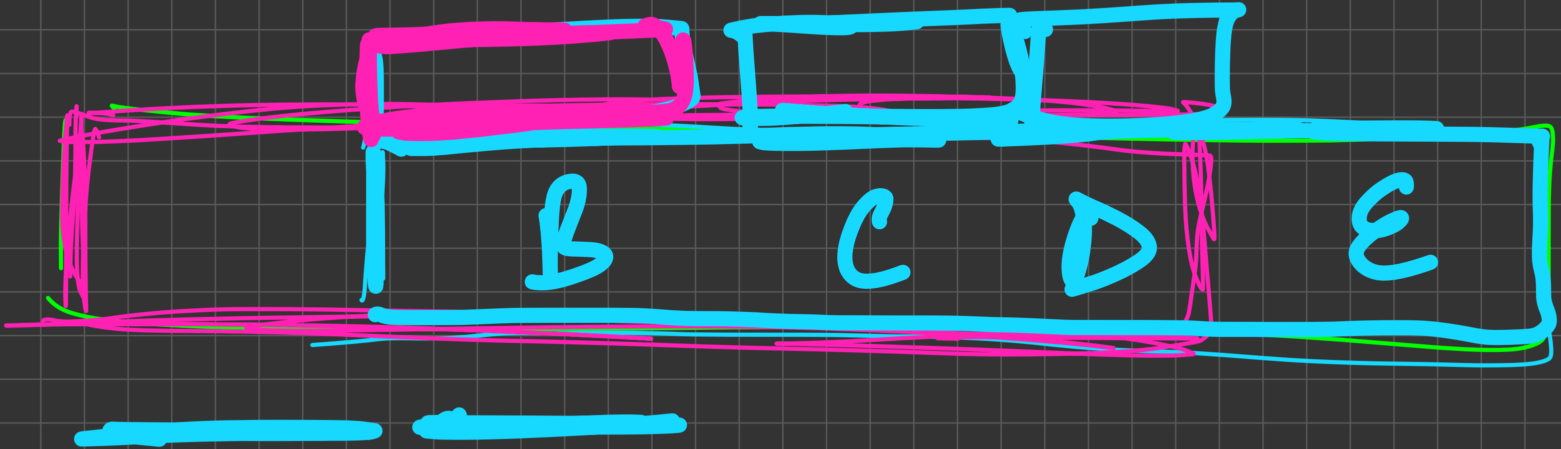
a	a	b	b	a	a	b	b	a	a	b	b
0	1	0	0	1	2	3	4	5	6	7	8

$n = 12$

$lps(12-1) = 8$

$n - lps(n-1) = \underline{\underline{4}}$

$\underline{\underline{12 | 4}}$ $\boxed{4}$



$A = B = C = D \triangleright E$

Applications of Prefix Function

- Pattern matching
- Detecting periodic string