String - Algos

↳ Rabin karp
↳ $\boxed{\text{KMP}}$

Z - Algorithm

↳ Pattern Matching

abcde = a, ab, abc, abcd, abcde

Prefix: any substring starting from 1st character

Construct a Z - array

Z[i] = Length of longest substring starting from index 'i' which is also the prefix of main string.

S[5] != S[0]

	0	1	2	3	4	5	6	7	8	9	10
S =	a	b	c	a	a	b	c	a	a	b	c
Z[] =	~~11~~	0	0	1	7	0	0	1	3	0	0

Z[] ↓ integer

Z[0] = length(S)

i = 3

S[3 : 5] = aab ?
S[3 : 4] : aa
S[3 : 3] : $\boxed{a}$ ✓

pref(S) = a, ab, abc, abca, abcaa, abcaab, . . .

S[4 : 10] = abcaabc

Naive Way

	0	1	2	3	4	5	6	7	8	9	10
S =	a	b	c	a	a	b	c	a	a	b	c

$P_1 = 0$ $\quad$ $P_2 = 4$

S[P1] = S[P2] ?

~~count++;~~

count = ~~2 3 4 5 6~~ 7

z[N];

$\boxed{z[0] = N}$

```
for (i=1; i<N; i++) {
    p1 = 0,   p2 = i
    count = 0;
    while (p2 < N && s[p1] == s[p2]) {
        count++;
        p1++, p2++;
    }
    z[i] = count;
}
```

$\boxed{\{ z[i] = p2 - i + 1 \}}$

$\boxed{T.C: \quad O(n^2)}$

?

s = "f d e" → pattern

t = "a e f d e d f d e f"

occurences = 2.

step 1:

1) First choose a character which will not occur in our string $ (delimiter)

⇒ O(t+s)

fde

s' = s + '$' + t ⇒ O(s+1+t)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
s' =	f	d	e	$	a	e	f	d	e	d	f	d	e	f
z[] =	#	0	0	0	0	0	3	0	0	0	3	0	0	1

occurenly = No. of occurence of len(s) in z-array

z[i] > 3 ?

$z[i] > len(S)$

$\boxed{fde}$ \$

prefix

$n = (s+t)$

pattern text

Step1 T.C :

1) Construct z-array ⇒ $O(t+s)^2$ $O(t+s)$

2) Iterate z-array ⇒ $O(t+s)$

$\boxed{T.C: \; O((t+s)^2)}$

$\boxed{T.C: \; O(t+s)}$?

z-array computation

Interval : (l, r) $\boxed{0 \le l, r \le n-1}$

↓

z-box

Interval:

For a given l, r is the max possible index such that $s[l:r]$ is the prefix.

Ex

	0	1	2	3	4	5	6	7	8	9	10
s =	a	b	c	f	g	x	a	b	c	f	d

p1 = 0, p1, p1, p1, p1

r = 6, r, r, r, r $\boxed{r = 10}$

$l = 6$ $\boxed{3}$ $r = 9$

[6, 9] is an interval / z-box   s[6 : 8] 8-6+1

s[6 : 8] = abc

9

pl = 0, r = l

while(r < n && s[pl] == s[r]) {   o(n)

r++;

pl++;

}

r = r-1;

s[0..... x] = s[l..... r]

x - 0 = r - l

S[0... r-l] = S[l...r]

1) $\boxed{x = r - l}$

2) Length of $r - l + 1$

3) s[0... j ... r-l] = s[l ... i ... r] $l<i<r$

j - 0 = i - l

L   i   R

[a b c f] g x [a b c f] d

0 1 2 3 4 5 6 7 8 9 10

$\boxed{j = i - l}$

4) s[0.... j .... r-l] = s[l .... i . — . r]

length of S[i:r] = r - i + 1

Point:

If we are computing z[i], then all values of z[j] j<i have been computed

s[i.... R-L] = s[i.... R]   z[7]

# of characters = R-L+1

z[1] z[2] . . . . z[6]

aaacb   aacb

i=1   i=6

Case 1

R-i+1

9-6+1 = $\boxed{4}$

s[0:4] = s[5:9]

s[1:4] = s[6:9]

j = i-L

	0	1	2	3	4	5	6	7	8	9	10
S =	a	a	a	c	b	a	a	a	c	b	d
z =	X	2	1	0	0	5	2	1	0	0	

L, R, i

R = 9
i = 6
9-6+1 = 4

$\boxed{z[i] = z[i-L]}$

$\boxed{j = i-L}$

aac

s[0:4] = s[5:9]

L = 5, R = 9

→ we know 4 characters from i=6 are equal to 4 character from i=1

(aaacb = aaacb)

"aa" is longest substring which is prefix

z[1] = 2

i=1   i=6

aacb   aacb

(aac)

z[1] = 2

aa

r-i+1 = 4

z[j] = z[i-L] = 2

if(z[j] < r-i+1) {
    z[i] = z[j];
}

z[j] < (r-i+1)
z[i] = z[j]

j = i-L = 11-8 = 3

i=11

Case 2

j = i-L

L   R

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S =	a	b	c	a	b	c	a	b	a	b	c	a	b	c	d	b
z =	X	0	0	5	0	0	2	0	6	0	0	5	0	0		

z[11] = z[11-8]

S[0:5] = S[8:13]

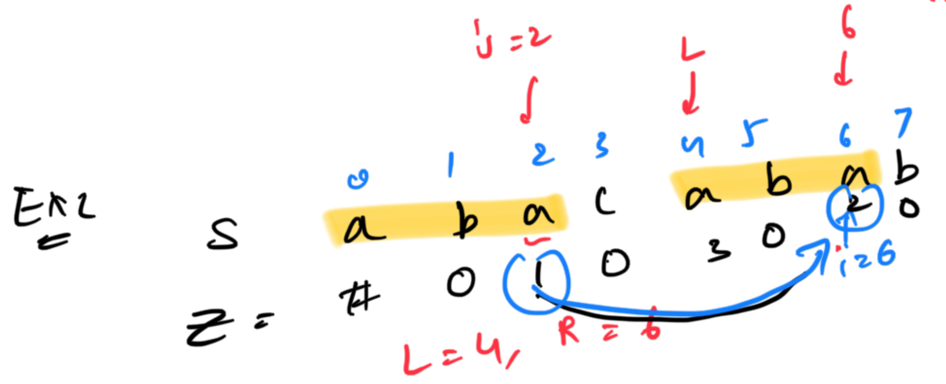
L = 8, R = 13

→ We know 3 characters from i = 11 are equal to 3 characters from j = 3

→ z[3] = 5 ⇒ abcab is the prefix

if(z[i] ≥ r − i + 1)

Ex 2



i = 6, j = 2

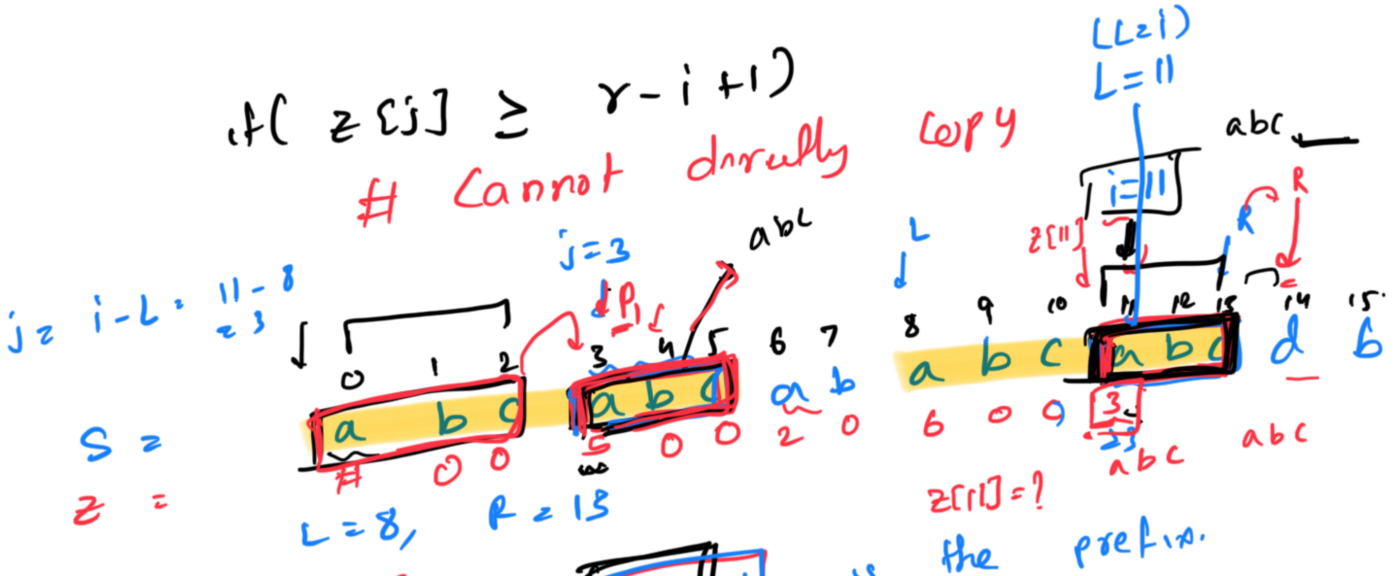
r − i + 1)
6 − 6 + 1 = $\boxed{1}$

z[j]
z[j] = z[2] = 1

r − i + 1 = 1

(z[j] = r − i + 1)

if(z[j] ≥ r − i + 1)
# Cannot directly copy

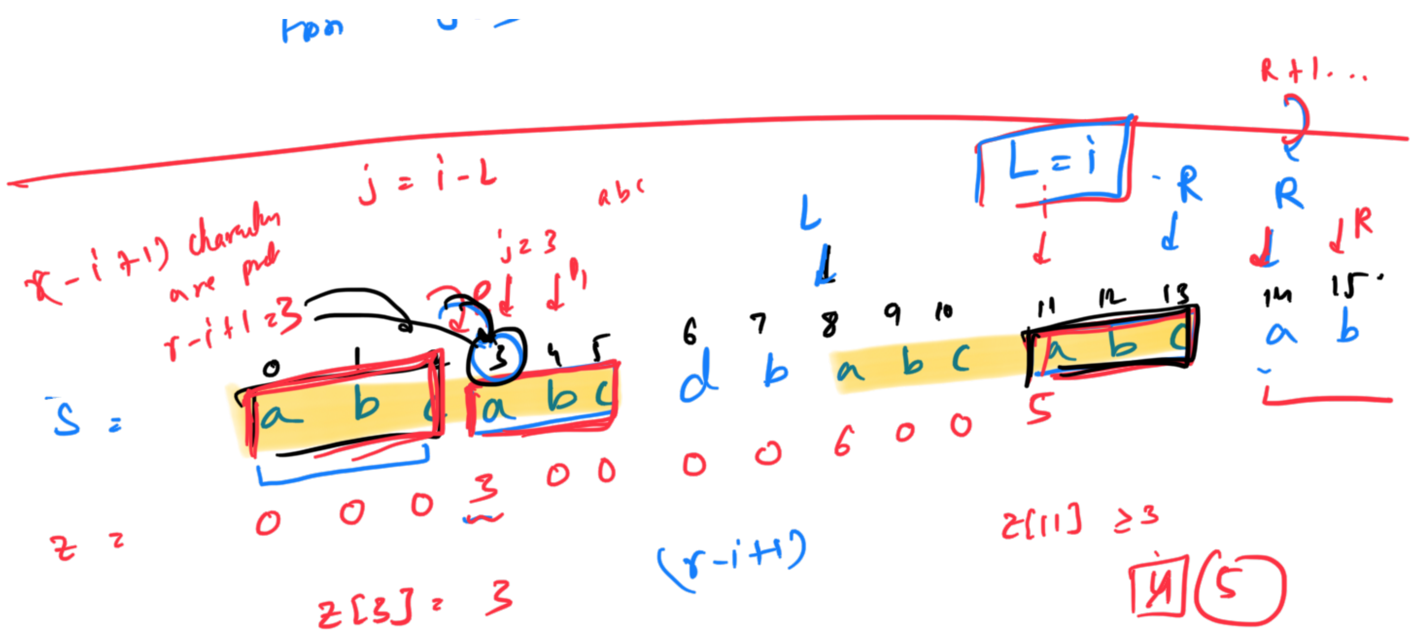


⇒ $\boxed{z[3] = 5}$ ⇒ $\boxed{abcab}$ is the prefix.

z[i] ≥ 3

→ 3 chars from i = 11 (abc) are equal to 3 chars
i = 3 (abc)

$\boxed{z[i] \geq r - i + 1}$



Sumarize :

1) If the current interval is [l, r] and we want to compute z[i] where

$i > R$,

then compute (z[i], L, R) using the usual brute force method

```
if( i > r) {
    L = i, R = i;
    // compute R for this given i
}
```

2) Case1 : $j = i - L$

```
if( z[j] < r - l + 1)
    z[i] = z[j];
```

Case 2 :

L = i;   R = R+1
P1 = r - i + 1;   R < N
while (s[R] == s[P1]) {
    R++;
    P1++;
}
R = R - 1

s[l : r] => r - l + 1

construct zarray (string s) {
    int N = s.length();
    L = 0, R = 0;
    for (i = 1; i < N; i++) {
        if (i > R) {
            L = i, R = i, P1 = 0
            while (R < N && s[R] == s[P1]) {
                R++; P1++;
            }
            R--;
            z[i] = R - L + 1;
        }
        else {
            j = i - L
            if (z[j] < R - L + 1) {
                z[i] = z[j];
            }
            else {
                L = i;
                P1 = r - i + 1, R = R + 1;
                while (R < N && s[P1] == s[R])
                    R++; P1++;

L = 0, R = 0
i = 1

L = 3
R = 5
i = 4

L ≤ i ≤ R

2a

2b

// z[j] ≥ R - L + 1)

L

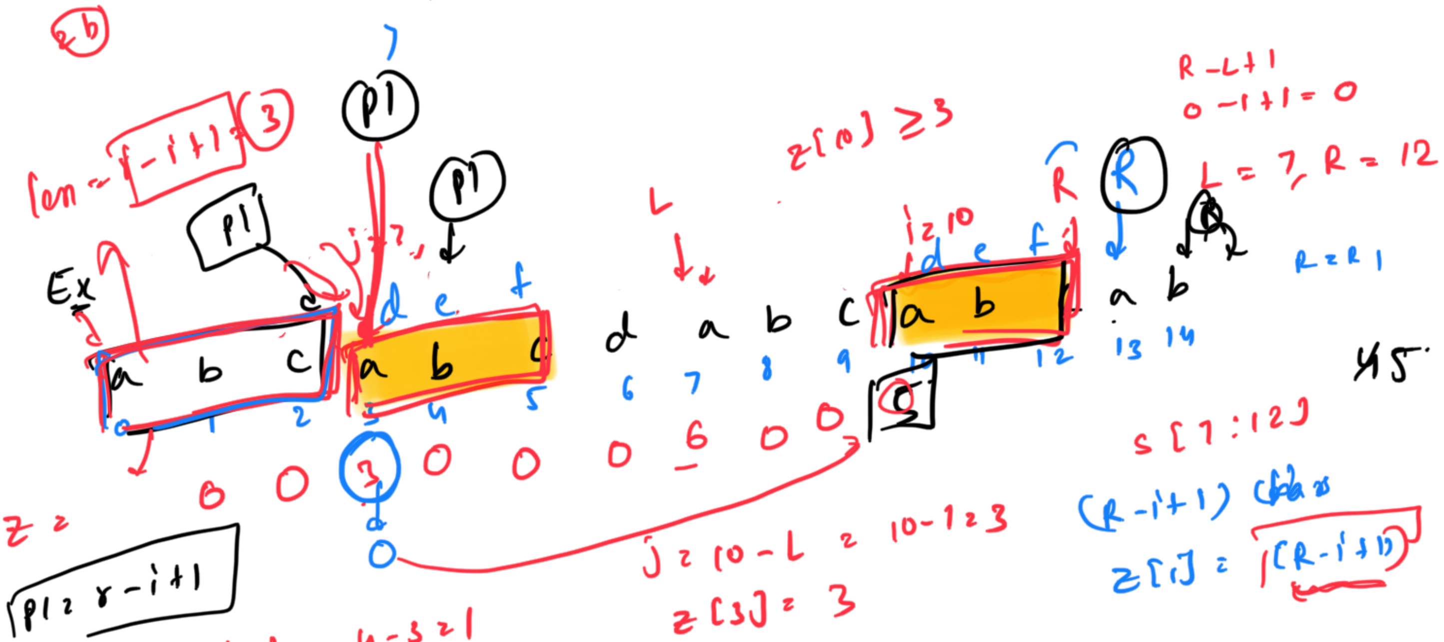
R = R-1

z[i] = R-L+1.

}

}

return z array.



j = i-L = 4-3 = 1

z[1] = 0

z[1] < r-l+1

R-i+1 = 12-10+1 = 3

pl = r-i+1

j = i-L = 2

z[2] = 0

T.C:

3 pointers

i [1 ⟶ N-1] O(n)

L [0 . . . . . . N-1] O(n)

R: [0 . . . . . . N-1] O(n)

2n iterations

At max ...

T.C: O(n)

S = fde

t = abfdegfdea

step1: $\boxed{S' = s + \text{'\$'} + t}$ O(t+s)

step2: compute z-array O(t+s)

step3: find no. of occurrency of len(s)

T.C: O(t+s)

S.C: O(t+s)

z-array

Question: Find period of string

S = abca abca abca abca $\boxed{k = 4}$ Period

abcaabca abcaabca => Period = 8

abcaabcaabcaabca = Period = 16

Ex2 ababababab $\boxed{k = 2}$

ab ab ab ab

abab abab = 2 K = 4

N = 17 k = 5

Ex 3

[a b a a b a b a a b a b a a b a b]

⇒ Period is the min value 'k' such that (Binary)

$S[i] = S[i \% k]$

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

$S[4] = S[4 \% 4]$

Lets consider a string "S" is repeating multiple time

$S' = S\ S\ S\ S\ S$ period = $|S|$

$S = abc$

$N = 12$, $K = 3$

$S' =$

			k			2k					
a	b	c	a	b	c	a	b	c	a	b	c
0	1	2	3	4	5	6	7	8	9	10	11
12	0	0	9	0	0	6	0	0	3	0	0

$Z =$ 12 0 0 9 0 0 6 0 0 3 0 0

i=1, i=2, i=3

$z[3] = N - K$

$z[K] = N - K$

$z[3] = 9$, $i = 3$, $3 + 9 = 12$

$z[i] + i$

If a string is periodic with period = k,

$z[K] = N - K$

$$\boxed{z[K] + K == N}$$

Find min index 'i' such that

$$\boxed{z[i] + i == N}$$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
						b	a	a	b	a	b	a	a	b

S = a b a a b a a b a a b a

z = 15 0 1 3 0 10 0 1 3 0 5 0 1 2 0

i=1 i=2 z[2]+2 N=15

z[3]+3

aba

z[5]+5 = 15?

10+5=15

// construct z-array → O(n)

```
for (i=1; i< N; i++) {
    if( i+ z[i] == N)
        return i;
```

→ O(n)

}

Question: Cyclic Permutations → {0,1}

Given 2 binary strings A, B,

Count cyclic Permutation of B such $A \wedge B = 0$ (A == B)

A = "1001"

B = 0110 → 0011 → 1001 → 1100

4 cycle permutat

Brute Force:

Permutation = n

T.C to check if this permutation = A? O(n)

T.C: $O(n \times n)$

: $O(n^2)$

A = 01102

Z - Algo

A = 1001 = S N

B = 0110 → 0011 → 1001 → 1100 → M

B + B = 0 1 1 0 0 1 1 0

2M

= 0 1 1 0 0 1 1

A + \$ + B+B

	0	1	2	3	4	5	6	7	8	9	10	11	12
S =	1	0	0	1	\$	0	1	1	0	0	1	1	0
Z =	#	0	0	1	0	0	1	4	0	0	1	2	0

A = 0110

N+2M

	0	1	2	3	4	5	6	7	8	9	10	11
S =	0	1	1	0	\$	0	1	1	0	0	1	1
Z =	#	0	0	1	0	4	0	0	1	3	0	0

$$\boxed{S' = A + \$ + B + B[0 \ldots n-2]}$$

T.C: $O(N+2M) = O(N+M)$

S.C: $O(N+M)$

2) In 99% of questions which maximi...

2) $\boxed{\text{Find longest Subarray of sum = 0 / K}}$

a = abcd

abcd, bcda .. 4!

abcd → dabc → cdab → bcda → abcd