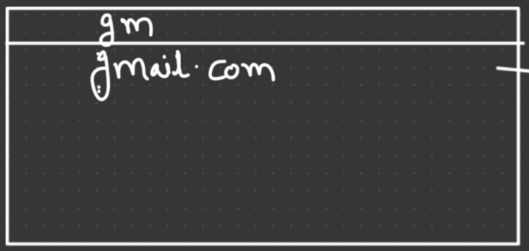
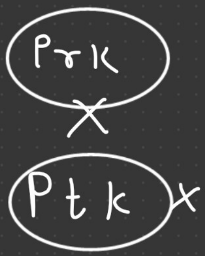
String Algorithms



$\longrightarrow$ Auto - completion.

	Prefix	Suffix
Prateek	P	K
	Pr	eK
	Pra	eeK
	Prat	teeK
	Prate	ateeK
	Pratee	rateeK
	PrateeK	PrateeK

Compare(prefix, Text list)
{
return list
}



Text = "My name is PrateeK" $\longrightarrow$ string Matching

Pattern = "PrateeK"

1) Brute force approach $\longrightarrow$
2) Robin Karp
3) KMP

Brute force Approach :-

Text = "$\overset{i}{\underline{A}}$BCAA$\overset{\downarrow}{\underline{B}}$$\underline{CD}$" = n

Pattern = "$\underset{j}{\underline{AAB}}$" = m

```
int BruteforceStringMatch(char T[], char P[], int n, int m)
{
    for(int i=0; i<=n-m; i++)
    {
        int j=0;
        while( j<m && T[i+j]==P[j])
        {
            j++;
        }
        if (j==m)
            return i;
    }
    return -1;
}
```

$O(n \times m)$

$\overset{i}{\underline{1}}\underline{23}121\overset{i}{\underline{4}}\underline{231}$

$n=10$

$\underset{j}{\boxed{2311}}$

$m=4$

$n-m$

$10-4=6$

2) Rabin Karp :- $O(n+m)$

Hashfunction (T(i, j) , p) $\longrightarrow O(1) \longrightarrow$ may be

(0,4) (1,5) $m = 4$ $\longrightarrow$ Not possible

T = 1 2 3 1 2 1 1 3 , $\boxed{121} \longrightarrow p$ $m=3$

i, j

$1\times10+2 = 12\times10+3 = \boxed{123}$

$2\times10+3 = 23\times10+1$

$= 231$

if (T % 10 == P % 10) $\longrightarrow O(1)$

{ may be

}

T = 123 % 10 = 3

P = 121 % 10 = 1

123 % 100

$23\times10+1$

$\boxed{231}$

$O(n+m)$

T = 231 % 10 = 1

P = 121 % 10 = 1 $\longrightarrow$ may be

231 % 100 = $31\times10+2$

$= \boxed{312}$

12 % 10 = $2\times10 = 20+1 = 21$

123 % 100 = $23\times10 = 230+1 = 231$

5641 % 1000 = $641\times10 = 6410+1 = 6411 \longrightarrow O(1)$

KMP Algorithm

$$\begin{tikzcd}[column sep=small] & \text{KMP Algorithm} \arrow[dl] \arrow[d] \arrow[dr] & \\ \text{Knuth} & \text{Morris} & \text{Pratt} \end{tikzcd}$$

Prefix Table

Ex:-

	j						i
	0	1	2	3	4	5	6
P	a	b	a	b	a	c	a
F	0	0	1	2	3	0	1

```
if(j==0)
{
  F[i]=0
    i++;
}
if( j>0)
   j = F[j-1]
```

$\rightarrow$ $\leftarrow$

aba

a	a	✓
ab	ba	

ababa

a	a	✓
ab	ba	X
aba	aba	✓
abab	baba	X

```
void prefixTable( int P[ ], int m)
{
    int i=1, j=0, F[0] =0;
                                  ⟶ O(m)
    while(i<m)
    {
        if (P[i]== P[j])          else if( j>0)
        {                         {
            F[i]=j+1;                  j = F[j-1];
             i++;
             j++;                 }
        }                         else { F[i] =0;
                                          i++;
                                       }
```

```
int KMP( char T[], char P[], int n, int m)
{
     int i=0, j=0;
      prefixTable( P, m);  ———>  O(m)

       while (i<n)      ————————>  O(n)
       {
            if (T[i]== P[i])
            {                                   O(n+m)
                  i++;  j++;                    ══════

                  if (j== m)
                     return i-j;
             }
            else if (j>0)
             {
                  j = F[j-1];
             }
            else
                  i++;
       }

       return -1;

}
```