



Longest substring with at most K distinct characters

$s = a a a b b c c d$        $k = 2$





Longest substring with at most K distinct characters

$s = a a \underline{a b b c} = d \quad k = 2$

4  
↙





Longest substring with at most K distinct characters

s = aabcc d      k = 2





Longest substring with at most K distinct characters

s = a a a b b c c d      k = 2





```
fun ( i = 0 → n )  
{
```

```
    mpp.clear()
```

```
    fun ( j = i → n )
```

```
        mpp[s[j]]++;
```

```
        if ( mpp.size() <= 1 )  
            maxlen = max( maxlen, j - i + 1 );  
    }
```



```

fun ( i = 0 → n )
{
    mpp.clear()

    fun ( j = i → n )
    {
        mpp[s[j]]++;

        if ( mpp.size() <= 1 )
            maxlen = max(maxlen, j - i + 1);
        else
            break;
    }
    return maxlen;
}

```





Longest substring with at most K distinct characters

256

s = aaabbbccd      k = 2



func (s, k)

map<char, int>

for (i = 0 → n)

map.clear()

for (j = i → n)

map[s[j]]++;

if (map.size() <= k)

TC →  $O(n^2)$   
×  $\log(256)$

SC →  $O(256)$



Longest substring with at most K distinct characters

$s = \underline{1} \underline{a a a b b c} \underline{c} \underline{d} \quad k = 2$

marken = 0 1 2 3 4 5

C	1
b	2
a	3







Longest substring with at most K distinct characters

$s = a^x a b b c^x d \quad k = 2$

manlen = 0 1 2 3 4 5

c	1
b	2
a	2

3

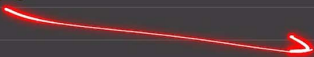




Longest substring with at most K distinct characters

$s = a a a b b c d \quad k = 2$

$\uparrow \quad \uparrow$



c	1
b	2

marker = 0 1 2 3 4 5



Longest substring with at most K distinct characters

$s = a a a \boxed{b b c c d}$   $k = 2$

marken = 0 1 2 3 4 5

C	2
b	2
d	1



$s = \overset{x \times x \times x}{a a a} b b c c d \quad k = 2$   
                                  1      n

```
func (s, k)
```

```
{
```

```
    maxlen = 0, l = 0, r = 0, mpp < char, int >
```

```
    while (r < s.size())
```

```
    {
```

```
        mpp[s[r]]++ ;
```





```
while ( mpp.size() > k )
{
    hash[s[l]]--;
    if (hash[s[l]] == 0) mpp.erase(mpp[s[l]]);
    l = l + 1;
}
if (
```





```
func (s, k)
```

```
{
```

```
    maxlen = 0, l = 0, r = 0, mpp < char, int >
```

```
    while (r < s.size())
```

```
    {
```

```
        mpp[s[r]]++;
```

```
        while (mpp.size() > k)
```

```
        {
```

```
            hash[s[l]]--;
```

```
            if (hash[s[l]] == 0) mpp.erase(mpp[s[l]]);
```

```
            l = l + 1;
```

```
        }
```





```
if (mpp.size() <= 1)
    maxlen = max(maxlen, n-l+1);
```

```
    n = n+1;
```

```
    ?
```

```
return maxlen;
```



$s = \overset{x \times x \times x}{a a a} b b c c d \quad k = 2$   
                                   1      2

$TC \rightarrow O(n) + O(n) + O$

$func(s, k)$   
 {

$maplen = 0, l = 0, r = 0, mpp \langle char, int \rangle$

$while (r < s.size())$   
 {

$mpp[s[r]]++;$

$\rightarrow while (mpp.size() > k)$   
 {

$hash[s[l]]--;$

$if (hash[s[l]] == 0) mpp.erase(mpp[s[l]]);$   
 $l = l + 1;$

}

$if (mpp.size() \leq k)$







Longest substring with at most K distinct characters

$s = \overset{x \times x \times x}{a a a b b c c d} \quad k = 2$   
                    1      n

TC  $\rightarrow O(N) + O(N)$   
          +  $O(\log 256)$   
SL  $\rightarrow O(256)$

func (s, k)

{  
  maxLen = 0, l = 0, r = 0, mpp < char, int >

while (r < s.size())  
{

  mpp[s[r]]++;

→ while (mpp.size() > k)  
  {

    hash[s[l]]--;

    if (hash[s[l]] == 0) mpp.erase(mpp[s[l]]);





}

aaa bbb c d

k = 2

manLen = 0 1 2 3 4 5

1, 1
5, 2
9, 3





}

<sup>x</sup>  
a a a b b c c d  
x L            n

k = 2

maxLen = 0 1 2 3 4 5

2, 1
5, 2
9, 3





}

xxx  
aaa bbb ccc  
xxx d r

6

k = 2

manin = 0 1 2 3 4 5

1.2  
5.2



```
func (s, k)
```

```
{  
    maxlen = 0, l = 0, r = 0, mpp < char, int >
```

```
    while (r < s.size())  
    {
```

```
        mpp[s[r]]++;
```

```
        while (mpp.size() > k)  
        {
```

```
            hash[s[l]]--;
```

```
            if (hash[s[l]] == 0) mpp.erase(mpp[s[l]]);  
            l = l + 1;
```

```
        }
```

```
        if (mpp.size() <= k)
```

```
            maxlen = max(maxlen, r - l + 1);
```

$+ O(\log 256)$   
 $SL \rightarrow O(256)$



func (s, k)

manLen = 0, l = 0, r = 0, mpp < char, int >

while (r < s.size())

    mpp[s[r]]++;

    if (mpp.size() > k)

        hash[s[l]]--;

        if (hash[s[l]] == 0) mpp.erase(mpp[s[l]]);

        l = l + 1;

    if (mpp.size() <= k)

        manLen = max(manLen, r - l + 1);

+ O(log 256)  
SL → O(256)





```
manLen = 0, l = 0, r = 0, mpp < char, int >
```

```
while (r < s.size())  
{
```

```
    mpp[s[r]]++;
```

```
    → if (mpp.size() > k)  
    {
```

```
        hash[s[l]]--;
```

```
        if (hash[s[l]] == 0) mpp.erase(mpp[s[l]]);  
        l = l + 1;
```

```
    }
```

```
    → if (mpp.size() <= k)  
        manLen = max(manLen, r - l + 1);
```

```
    r = r + 1;
```

```
}
```



### Longest substring with at most K distinct characters

$s = a a a b b c c d \quad k = 2$

T.C  $\rightarrow O(N) + O(\log 256)$   
S.C  $\rightarrow O(256)$

June (5, 10)

$maxlen = 0, l = 0, r = 0, mpp \langle char, int \rangle$

```
while (n < s.size())
```

```
mpp { s[n] } ++ ;
```

→ if ( mpp.size() > k )

hash [sfa] --;

if (hash[ss[1]] == 0) mpp.erase(mpp[ss[1]]);