1278. Palindrome Partitioning III

Hard 195 3 Add to List Share

You are given a string `s` containing lowercase letters and an integer `k`. You need to :

- First, change some characters of `s` to other lowercase English letters.
- Then divide `s` into `k` non-empty disjoint substrings such that each substring is palindrome.

Return the minimal number of characters that you need to change to divide the string.

Example 1:

```
Input: s = "abc", k = 2
Output: 1
Explanation: You can split the string into "ab" and "c", and change 1 character in "ab" to make it palindrome.
```

Example 2:

```
Input: s = "aabbc", k = 3
Output: 0
Explanation: You can split the string into "aa", "bb" and "c", all of them are palindrome.
```

Example 3:

```
Input: s = "leetcode", k = 8
Output: 0
```

minimum changes to convert (i,j) substring into palindindrome.

pdp:

	a	b	b	c	c	d	e	f	f	g	h	h
a	0	1	1	1	2	3	3	3	4	5	5	6
b		0	0	1	2	2	2	3	4	4	5	5
b			0	1	1	1	2	3	3	4	4	5
c				0	0	1	2	2	3	3	4	4
c					0	1	1	2	2	3	3	3
d						0	1	1	2	2	2	3
e							0	1	1	1	2	3
f								0	0	1	2	2
f									0	1	1	2
g										0	1	1
h											0	0
												0

```
for (int gap = 0; gap < n; gap++)
{
    for (int i = 0, j = gap; j < n; i++, j++)
    {
        if (gap == 0)
        {
            pdp[i][j] = 0;
            continue;
        }
        pdp[i][j] = pdp[i + 1][j - 1];
        if (s[i] != s[j])
            pdp[i][j] += 1;
    }
}
```

s(i) == s(j) → agar character equal hai to (i+1,j-1) mai jitne changes the utne hi chhaiye.

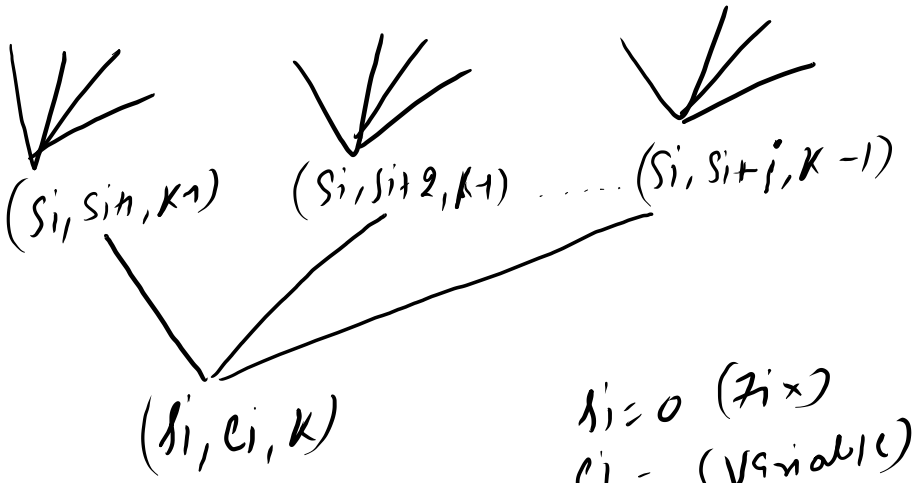
s(i) != s(j) → +1 agar equal nahi to (i+1,j-1) mein jitne changes the utne + 1(because unequal mein kisi ek ko change krdo.)

recursive call to divide int to k subset.

base case:

if (k == 0) return dp[k][ei] = 0;

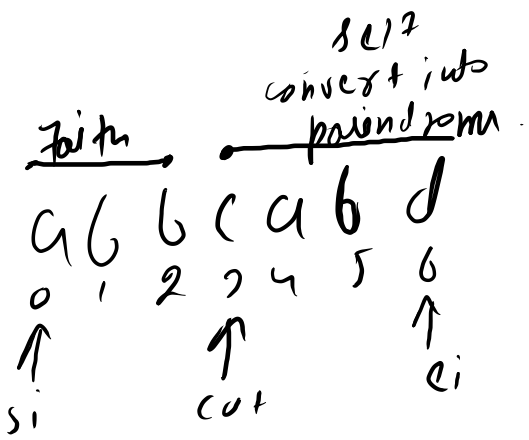
if (k == 1) return dp[k][ei] = pdp[0][ei];



Si = 0 (fix)

ei = (variable)

so -> 2D - dp.




```
for (int k = 0; k <= K; k++)
{
    for (int si = 0, ei = 0; ei < s.length(); ei++) // si is fix to 0,only ei vary.
    {
        if (k <= 1)
        {
            dp[k][ei] = ((k == 0) ? 0 : pdp[0][ei]);
            continue;
        }

        if (si == ei)
        {
            dp[k][ei] = 0;
            continue;
        }

        int min_ = (ei - si) + 1; // total character in between si to ei.
        for (int cut = si + 1; cut <= ei; cut++)
            min_ = min(min_, dp[k - 1][cut - 1] + pdp[cut][ei]);

        dp[k][ei] = min_;
    }
}
```