

# Palindrome Partitioning II

## Problem Statement

[Suggest Edit](#)

You are given a string '*str*' of length '*n*'.

Find the minimum number of partitions in the string so that no partition is empty and every partitioned substring is a palindrome.

Example :

Input: '*str*' = "aaccb"

Output: 2

Explanation: We can make a valid partition like aa | cc | b.

## Explanation and Understanding question:

s = a | a | c | c | b

Q. How many max partition  
can be there for  
the string to be palindrome

→ n-1 (max)

Yes there is possibility at least that  
every string has a maximum  $n-1$  partitions

We have find the minimum Partition

a a | c c | b  
~~~~~  
p

2 partition  
is minimum

How to Solve it?

Approach:

$\max = \underline{\underline{n-1}}$

Front Parthiban Algo

$1 + \text{cost}(j+1)$



Cost = 1 +

min =

len

Recursive:

$f(i)$

if  $(i \geq n)$  return 0,  
temp = ""

for  $(j = i; j < n; j++)$

temp +=  $s[j]$ ,

if (~~temp~~ isPalindrome(temp))  
Cost = 0 +  $f(j+1)$ ;

MinCost =  $\min(\text{minCost}, \text{Cost})$ .

# Space Optimization



int furr → curr, min

a a/b/c/d



```

int minPartitionT(string &s){
    int n=s.size();
    vector<int> dp(n+1,0);
    for(int i=n-1; i>=0; i--){
        int minCost = 1e9;
        for(int j=i; j<n; j++){
            if(isPalindrome(i,j,s)){
                int cost = 1 + dp[j+1];
                minCost = min(minCost,cost);
            }
        }
        dp[i] = minCost;
    }
    return dp[0];
}

```

