**★★★☆☆**

**題組：Problem Set Archive** **with Online Judge**

**題號：11584: Partitioning by Palindromes**

**解題者：符新祐**

**解題日期：2021年4月21日**

**題意：**

給N，為測資數，給N個不超過1000的字串，求此字串**最少必須**由幾個回文字串組成

字元視為回文字串: A, z

**題意範例：**

Input:

5

racecar

ghijkmnl

amadam

madama

zxzczxxzc

Output:

1

8

2

2

2

**解法：**

找&更新**分割數**

用DP:

if(isPalindrome(j - 1, i - 1))

                    DP[i] = min(DP[i], DP[j - 1] + 1);

 for 每個字元:

 往前找所有**回文字串**，找到**回文字串**代表可以分割成此**回文字串**前的分割數 + 1(此回文字串)

**解法範例：**

aaa**d**bccb **aaad**bccb 2 //假設已知

aaad**b**ccb : 3

 aaadb**c**cb : 4

 aaadbc**c**b : 4

 aaadbcc**b** aaad**bccb** 2 + 1 = 3

⸫ 總分割數 = “last b 分割數” = 3

**討論：**

 1.初始化設定: DP[0] = 0、DP[1] = 1

2.cstring vs string

scanf() 可以在讀的同時一個一個處理

 string.clear()

**程式：**

#include<iostream>

#include<string>

#include<algorithm>

using namespace std;

string inputStr;

bool isPalindrome(int, int);

int main(){

    int caseNum, caseCount = 0;

    int n, j, count = 0;

    int DP[1000];

    cin >> caseNum;

    while(caseCount < caseNum){

        cin >> inputStr;

        for(int i = 2; i <= inputStr.length(); i++){

             DP[i] = 10000;

        }

        DP[0] = 0;

        DP[1] = 1;

        //DP

        for(int i = 2; i <= inputStr.length(); i++){

            DP[i] = DP[i - 1] + 1;

            for(j = 1; j < i; j++){

                if(isPalindrome(j - 1, i - 1))

                    DP[i] = min(DP[i], DP[j - 1] + 1);

            }

        }

        cout << DP[inputStr.length()] << endl;

        //loop reset

        count = 0;

        caseCount++;

        inputStr.clear();

    }

    return 0;

}

bool isPalindrome(int a, int b){

    int n = 0;

    int edge = (b - a + 1) / 2;

    while(n < edge){

        if(inputStr[a] != inputStr[b])

            return false;

        a++;

        b--;

        n++;

    }

    return true;

}