#include<bits/stdc++.h>

using namespace std;

bool pall(string str, int i, int j){

int s = i;

int e = j;

while(s<e){

if(str[s]!=str[e]){

return false;

}

s++;

e--;

}

return true;

}

int solve\_dp\_optimised(string s, int i, int j, vector<vector<int> > &dp){

if(i>=j) return dp[i][j] = 0;

if(pall(s,i,j)) return dp[i][j] = 0;

if(dp[i][j]==-1){

int ans = INT\_MAX;

for(int k=i;k<j;k++){

int temp\_ans = (dp[i][k]==-1?solve\_dp\_optimised(s,i,k,dp): dp[i][k]) + (dp[k+1][j]==-1?solve\_dp\_optimised(s,k+1,j,dp):dp[k+1][j]) + 1;

ans = min(ans,temp\_ans);

}

return dp[i][j] = ans;

}

return dp[i][j];

}

int solve\_dp(string s, int i, int j, vector<vector<int> > &dp){

if(i>=j) return dp[i][j] = 0;

if(pall(s,i,j)) return dp[i][j] = 0;

if(dp[i][j]==-1){

int ans = INT\_MAX;

for(int k=i;k<j;k++){

int temp\_ans = solve\_dp(s,i,k,dp) + solve\_dp(s,k+1,j,dp) + 1;

ans = min(ans,temp\_ans);

}

return dp[i][j] = ans;

}

return dp[i][j];

}

int solve\_rec(string s, int i, int j){

if(i>=j){

return 0;

}

if(pall(s,i,j)) return 0;

int ans = INT\_MAX;

for(int k=i;k<j;k++){

int temp\_ans = solve\_rec(s,i,k) + solve\_rec(s,k+1,j) + 1;

ans = min(ans,temp\_ans);

}

return ans;

}

int main(){

int n;

string s;

cin>>s;

n = s.size();

vector<vector<int> > dp(n+1,vector<int> (n+1,-1));

cout<<solve\_dp\_optimised(s,0,n-1,dp);

}