dp[l][r] express the least inserted characters to convert string **A** in interval [l,r] to a palindrome

int solve(string a, int l, int r)

{

if (dp[l][r] != -1) //the initial value of dp is -1 ，if dp[l][r]’s value is -1, it means algorithm doesn’t calculate dp[l][r]. Otherwise dp[l][r] has been calculated, just return value directly.

return dp[l][r];

if (l >= r) //if l == r, string a[l,r] has been a palindrome without extra character. Meanwhile, l > r is a illegal interval, so dp[l][r] = 0;

return dp[l][r] = 0;

dp[l][r] = min(solve(a, l + 1, r), solve(a, l, r - 1)) + 1; // dp[l][r] can transferred from [l+1,r] or [l,r-1]. In on situation [l+1,r], dp[l+1][r] express the number of extra characters to convert sting a[l+1][r] to a palindrome, so we can add a[l] in right side, just like a[l]a[l+1]…a[r]a[l], which can denoted as dp[l][r] = dp[l+1][r]+1. So do[l,r-1].

if (a[l] == a[r]) //There exists another situation is a[l] == a[r]，so we can transfer [l+1][r-1] to [l,r] directly.

dp[l][r] = min(dp[l][r], solve(a, l + 1, r - 1));

return dp[l][r];

}