1. 求回文串除了通过栈和头尾字母逐一对比这两个比较低效的方法外还有如下两种方法
2. 动态规划

Isroll(s)

{

Let len = s.length,dp =Array(len).fill(false).map(()=>{return Array(len).fill(false)});

Let ans = 0;

//dp[i][j]表示s的从i到j的子串是否为回文串

For(let i=0;i<len;i++)

For(let j=0;j<=i;j++)//判断j到i的子串是否为回文串

{

If(i===j)

{

Dp[j][i]=true;

Ans++;

Continue;

}

Dp[j][i]= s[i]===s[j] && (i-j<=1 || dp[j+1][j-1]);

//i-j<=1是为了防止边界情况的误判如 “aa” || “ab”,对于一个字符串如果 //s[i]=s[j]（头尾字符相同） 那么只需要判断去头去尾的子串是否为回文即可

If(dp[i][j])

Ans++;

}

Return ans;

}

1. 中心扩展法

Isroll(s)

{

Let ans = 0;

let expend = function(start,end)

{

Let count = 0;

While(start>=0 && end<s.length &&(s[start--]===s[end++])

Count++;

Return count;

}

For(let i=0;i<s.length;i++)

{

Ans+=expend(i,i);//计算子串长度为奇数的情况

Ans+=expend(i,i+1);//计算子串长度为偶数的情况

}

Return ans;

}

//上面的两个代码可以求出字符串s的所有回文子串的个数，稍微修改一下就可以作为判断str是否为回文串的函数