5.Longest Palindromic Substring

Given a string **s**, find the longest palindromic substring in **s**. You may assume that the maximum length of **s** is 1000.

Example:

```
Input: "babad"

Output: "bab"

Note: "aba" is also a valid answer.
```

Example:

```
Input: "cbbd"

Output: "bb"
```

My Thought

Absolutely it's a ***DP*** problem. We use a mark array **hw[i][j]** to indicate if index between **i** and **j** in string **str** is a **Palindromic Substring**, than we get update rule like this:

$$\begin{aligned} hw[i][i] &:= true, \\ hw[i][i+1] &:= true \text{ if } str[i] = str[i+1] \\ hw[i][j] &:= true \text{ if } hw[i+1] = hw[j-1] \text{ and } str[i] = str[j] \end{aligned}$$

with these update rules, we can write code with **O(N^2)**

Code(C++ 73ms)

```
class Solution {
public:
    bool hw[1001][1001]={false};
    string longestPalindrome(string s) {
        int len = s.size();
        hw[len-1][len-1]=true;
        int maxlen = 1, strbegin=0;
        for(int i=0;i<len-1;++i){
        hw[i][i]=true;
        if(s[i]==s[i+1]){
            hw[i][i+1]=true;
            maxlen = 2;
            strbegin = i;
```

```
        }
    }
    for(int l=3;l<=len;++l){
        for(int j=0;j<len-l+1;j++){
            if(s[j]==s[j+l-1]&&hw[j+1][j+l-2]){
                hw[j][j+l-1]=true;
                maxlen=l;
                strbegin=j;
            }
        }
    }
    return s.substr(strbegin,maxlen);
    }
};
```