//Recursive simple

```
int solve(int n)
    {
        if(n == 0)
            return 1;

        if(n == 1)
            return 1;

        int left    = solve(n - 1);
        int right   = solve(n - 2);

        return left + right;
    }

    int climbStairs(int n)
    {
        return solve(n);
    }
```

//Recursive - 2

```
    int solve(int n)
    {
        if(n == 0)
            return 1;

        if(n == 1)
            return 1;

        return solve(n - 1) + solve(n - 2);
    }

    int climbStairs(int n)
    {
        return solve(n);
    }
```

//memoized

```
class Solution {
public:
    int solve(int n,vector<int> &dp)
    {
        if(n == 0)
            return 1;

        if(n == 1)
            return 1;

        if(dp[n] != -1)
            return dp[n];

        return dp[n] = solve(n - 1,dp) + solve(n - 2,dp);
    }

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

        return solve(n,dp);
    }
};
```

//tabular

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

      dp[0]= 1;
      dp[1]= 1;

      for(int i=2; i<=n; i++)
      {
          dp[i] = dp[i-1]+ dp[i-2];
      }
      return dp[n];
    }
```