DSA SERIES

- Learn Coding

Topic to be Covered today

Dynamic Programming

70. Climbing Stairs

```
class Solution {
public:
    int solve(int n, vector<int>& dp) {
        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);
    }
};
```

279. Perfect Squares

```
class Solution {
public:
    // int solve(int n, vector<int> &dp ) {


    //     if (n == 0)
    //         return 0;


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


    //     int ans = INT_MAX;


    //     for (int i = 1; i*i <= n; i++) {
    //         ans = min(ans, 1 + solve(n - i * i , dp));
    //     }
```

```
    //     return dp[n] = ans;
    // }
    int numSquares(int n) {
        vector<int> dp(n + 1, INT_MAX);
        dp[0] = 0;


        for (int i = 1; i <= n; i++) {
            for (int j = 1; j * j <= i; j++) {
                dp[i] = min(dp[i], 1 + dp[i - j * j]);
            }
        }
        return dp[n];
    }
};
```

198. House Robber

```
class Solution {
public:
    int t[101];
    int solve(vector<int>& nums, int i, int n) {
        if (i >= nums.size())
            return 0;


        if (t[i] != -1)
            return t[i];
        int take = nums[i] + solve(nums, i + 2, n);
        int not_take = solve(nums, i + 1, n);


        return t[i] = max(take, not_take);
    }
    int rob(vector<int>& nums) {
        int n = nums.size();
```

```
        memset(t, -1, sizeof(t));
        return solve(nums, 0, n);
    }
};
```

Learn coding

THANK YOU