**Date : 15-07-23**
**Mentor : Gladden Rumao**
**Topic : DP Interview Problems**

## Q1. Number Of Paths :

**https://practice.geeksforgeeks.org/problems/number-of-paths0926/1**

```
class Solution{


    long numberOfPaths(int m, int n) {
        long[][] M = new long[m][n];

        for(int i=0 ; i<m ; i++){
            for(int j=0 ; j<n ; j++){
                if(i==0 || j==0) //first row or first column
                    M[i][j] = 1;
                else
                    M[i][j] = M[i-1][j] + M[i][j-1];
            }
        }

        return M[m-1][n-1];
    }


}
```

## Q2. Maximum Path Sum :

**https://practice.geeksforgeeks.org/problems/path-in-matrix3805/1**

```
class Solution{
    static int maximumPath(int N, int M[][])
    {
        for(int i=1 ; i<N ; i++){
            for(int j=0 ; j<N ; j++){
                if( j==0 ){ //first column
```

```
                    M[i][j] += Math.max( M[i-1][j] , M[i-1][j+1]);
                }
                else if(j==N-1){ //last column
                    M[i][j] += Math.max( M[i-1][j-1] , M[i-1][j]);
                }
                else{
                    M[i][j] += Math.max ( M[i-1][j-1] , Math.max(
M[i-1][j],M[i-1][j+1]) );
                }
            }
        }
        // find the maximum from last row
        int ans = 0;
        for(int j=0 ; j<N ; j++){
            ans = Math.max ( ans , M[N-1][j] );
        }
        return ans;

    }
}
```

### Q3. House Robber

https://leetcode.com/problems/house-robber/

```
class Solution {
    public int rob(int[] nums) {
        int n = nums.length;

        //only 1 house
        if(n==1)
            return nums[0];
        int ans[] = new int[n];
        ans[0] = nums[0];
        ans[1] = Math.max( nums[0] , nums[1] );

        for(int i=2 ; i<n ; i++){
            ans[i] = Math.max( nums[i]+ ans[i-2]  ,  ans[i-1] );
                              //robbed            , not robbed
        }
        return ans[n-1];
    }
}
```