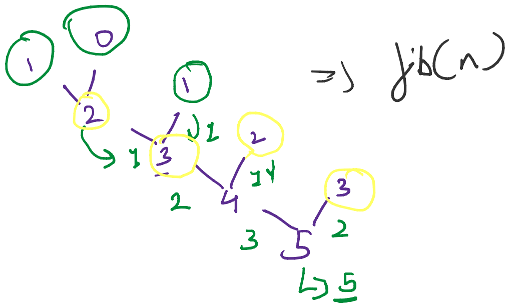
10th Feb

10 February 2022 18:54

Dynamic Prog.:-

fib:- 0, 1, 2, 3, 4, 5, 6, ...

[0, 1, 1, 2, 3, 5, 8 . . . . .]



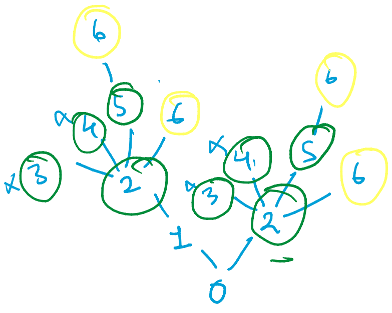
```
function (x, y)                dp = [ ][ ]

    if (<statement>) {
        functn (x+1, y)
    }
    else {
        function (x, y+2);
    }
```

Possible No. of Paths with variable jumps:-

$$\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \\ 2, & 1, & 4, & 0 & 0 & 1 \end{array}$$

[2, 1, 4, 0 0 1], Size = 6

Answer: 4



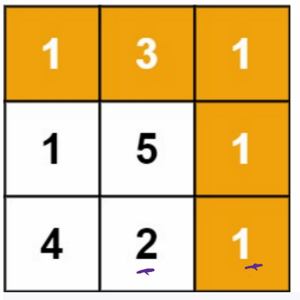
```
          solRec (arr, i) {
if (i == arr.length) return 1;
if (i > arr.length) return 0;
          jumps = arr[i];
          res = 0
    for (j = 1; j <= jumps; j++) {
    res += sol rec (arr, i+j);
    }
```

https://leetcode.com/problems/minimum-path-sum/

Min. Path Sum:-

```
          solRec (arr, i, j) {
if ( i == arr.length -1 &&
     j == arr.length -1 ) return arr[i][j];
```

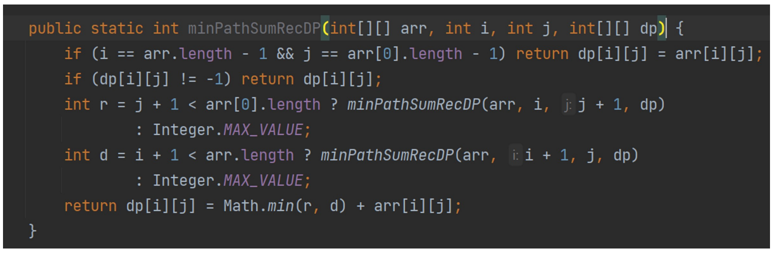
1	3	1



1	3	1
1	5	1
4	2	1

```
if( i==ars.length-1 &&
    j==ars.length-1 ) return arr[i][j];

int r = j+1 < ars[0].leng ? SolRec(arr,i,j+1):
            Int.Max_Val.
int d = i+1 < ars.lengu ? SolRec(arr,i+1,j ):
            Int.Max_Val.
return min(r,d) + ars[i][j];
```

o/p = 7



```java
public static int minPathSumRecDP(int[][] arr, int i, int j, int[][] dp) {
    if (i == arr.length - 1 && j == arr[0].length - 1) return dp[i][j] = arr[i][j];
    if (dp[i][j] != -1) return dp[i][j];
    int r = j + 1 < arr[0].length ? minPathSumRecDP(arr, i, j: j + 1, dp)
            : Integer.MAX_VALUE;
    int d = i + 1 < arr.length ? minPathSumRecDP(arr, i: i + 1, j, dp)
            : Integer.MAX_VALUE;
    return dp[i][j] = Math.min(r, d) + arr[i][j];
}
```

7	6	3
8	7	2
7	3	1

```
tab Sol (ars) {
 dp [ars.lngth ] [ars[0].lngth ];
dp [ars.lngth-1] [ars[0].lngth-1 ] = ars [ars.lngth-1] [ars[0].lngth-1 ];
```