**Qn Link :** <https://www.desiqna.in/13822/google-dp-interview-question-jan-2023>

**Question :**

* You re given an array of size “N” and need to reach index “N” from 0.
* For each index , you can either move to index + 1 or index + 2.
* Find the number of ways to reach the index “N” such that the sum of the journey is odd and even.

**Sample Test case :**

# **[ 5 4 2 6 ]**

Number of ways to reach “N” with sum ODD: 3

(5 , 4 , 2 ,6)

(5 , 4 , 6)

(5 , 2 , 6)

Number of ways to reach “N” with sum EVEN: 0

Observation :

* Lets create an 2D dp array , because for each step we need to take count of number of ways to reach the destination by odd sum and even sum.
* dp[i][even] will hold the number of ways to reach the index “i” with sum even.
* dp[i][odd] will hold the number of ways to reach the index “i” with sum odd.

[2 , 3 , 5 , 8 , 10]

* dp[0][even] = 1 , dp[0][odd] = 0
* dp[1][even] = 0 , dp[1][odd] = 1
* dp[2][even] = 1 , dp[2][odd] = 0
* dp[3][even] = 0 , dp[3][odd] = 1
* dp[4][even] = 0 , dp[4][odd] = 1

Did we forgot anything ??

→ Yes , we only calculated , by using only one jump to reach the index “N”.

**Trick used by Sir :**

So let consider , we re at index “i” , if the value at index “i” is even , then what is the number of ways to reach the index “i” with the sum even and odd.

* Simple , dp[i- 1] [even] + dp[i - 2] [even] , because even number sum with even number results in even number.
* For odd , dp[i- 1] [odd] + dp[i - 2] [odd] , because sum of odd number with even number results in odd number.

Let the number at index “i” is odd , then adding odd number with even number results in odd and adding odd number with odd number results in even number.

* dp[i] [even] = dp[i- 1] [odd] + dp[i - 2] [odd]
* dp[i] [odd] = dp[i- 1] [even] + dp[i - 2] [even]

Thats it.

Recurrence relation :

If index “i” is even , then

* dp[i][even] = dp[i-1][even] + dp[i-2][even]
* dp[i][odd] = dp[i- 1] [odd] + dp[i - 2] [odd]

If index “i” id odd , then

* dp[i] [even] = dp[i- 1] [odd] + dp[i - 2] [odd]
* dp[i] [odd] = dp[i- 1] [even] + dp[i - 2] [even]

Code :

class Solution {

public int[] numberOfJourney(int [] nums , int n) {

int [][] dp = new int[n][2];

//Let's consider 0 for even and 1 for odd.

if(nums[0] % 2 == 0){

dp[0][0] = 1

}else{

dp[0][1] = 1;

}

//We need to be careful when calculating for index 1

//if the value at index 1 is even , then

if(nums[1] % 2 == 0){

//if the previous value is also even , then we can put 1

if(dp[0][0] == 1)

dp[1][0] = 1;

//If the previous value is odd , then it will be odd (odd + even) = odd

else

dp[1][1] = 1;

}

//if the number at current index is odd

else{

//if the previous index element is odd , then it will be even (odd + odd) = even

if(dp[0][1] == 1)

dp[1][0] = 1;

//If the previous index is even , then it will be odd (even + odd) = odd

else

dp[1][1] = 1;

}

for(int i = 2 ; i < n ; i++){

// we use "0" for even and "1" for odd

if(nums[i] % 2 == 0){

// dp[i][even] = dp[i-1][even] + dp[i-2][even]

// dp[i][odd] = dp[i- 1] [odd] + dp[i - 2] [odd]

dp[i][0] = dp[i - 1][0] + dp[i - 2][0];

dp[i][1] = dp[i - 1][1] + dp[i - 2][1];

}else{

// dp[i] [even] = dp[i- 1] [odd] + dp[i - 2] [odd]

// dp[i] [odd] = dp[i- 1] [even] + dp[i - 2] [even]

dp[i][0] = dp[i - 1][1] + dp[i - 2][1];

dp[i][1] = dp[i - 1][0] + dp[i - 2][0];

}

}

}

}