**Question :**

You are given an array of size “N” ; you have to start your journey at index “1” and you need to end your journey at index “N”.

You can make jumps of size - 1 or 3 or 5 .

In the array positive as well as negative numbers can be available.

Please find the maximum sum of journey .

**Observation :**

* If the size of the array is 1 , then start and end is same.
* So , we can’t put dp[0] is zero.

Step 1 : Create an dp array of size N.

Step 2 : Dp[0] is array of zero not zero

Step 3 : calculate Dp values upto index 4 manually.

Recurrence Relation :

Dp[i] = max ( dp[i - 1] + nums[i] , dp[i - 3] + nums[i] , dp[i - 5] + nums[i])

class Solution {

private int max(int a , int b , int c){

if(a > b && a > c){

return a;

}else if ( b > c){

return b;

}

return c;

}

public int maxJouney(int [] nums) {

int n = nums.length;

int [] dp = new int[n];

dp[0] = nums[0];

dp[1] = nums[0] + nums[1];

dp[2] = nums[2] + dp[1];

// For 3 , we have two option either -1 or -3

dp[3] = math.max(dp[0] + nums[3] , dp[2] + nums[3]);

//For 4 , we have two options too , either (4 - 1) -> 3 or (4-3) -> 1

dp[4] = Math.max(dp[1] + nums[4] , dp[3] + nums[4]);

for(int i = 5 ; i < n ; i++){

dp[i] = max(dp[i - 1] , dp[i - 3] , dp[i - 5]) + nums[i];

}

return dp[n - 1];

}

}