My idea: Using 1 array with size n + 1 to store each calculated value. Result = array[n]

World idea:

**ntuition**

* The fibbonacci and tribonacci series which only depends on previous terms.
* This can be easily solved using constant space and can be converted in optimized codes instead of complex one.

**Approach**

* Store the previous three terms of the series and follow as in fibbonacci series.

**Complexity**

* Time complexity: O(n)
* Space complexity: O(1)

**Code**

class Solution {

public:

int tribonacci(int n) {

int dp[3] = {0, 1, 1};

for (int i = 3; i <= n; ++i)

dp[i%3] += dp[(i+1)%3] + dp[(i+2)%3];

return dp[n%3];

}

};