**343. Integer Break: -**

**Medium Accepted: 288.6K Submissions: 493.7K Acceptance Rate: 58.5%**

Given an integer n, break it into the sum of k **positive integers**, where k >= 2, and maximize the product of those integers.

Return *the maximum product you can get*.

**Example 1:**

**Input:** n = 2

**Output:** 1

**Explanation:** 2 = 1 + 1, 1 × 1 = 1.

**Example 2:**

**Input:** n = 10

**Output:** 36

**Explanation:** 10 = 3 + 3 + 4, 3 × 3 × 4 = 36.

**Constraints:**

* 2 <= n <= 58

**Code: -**

class Solution {

public:

    unordered\_map<int,int> universe;

    int helper(int n, unordered\_map<int,int> &mp){

        // base case

        // dp found case

        if(universe[n] > 0)

            return universe[n];

        if(mp[n] > 0)

            return mp[n];

        // recursive case

        int leftmul, rightmul, ans = INT\_MIN;

        for(int i=2; i<=n/2; ++i){

            leftmul = helper(i, mp);

            rightmul = helper(n-i, mp);

            ans = max(ans, leftmul \* rightmul);

        }

        // return from current state

        return mp[n] = ans;

    }

    int integerBreak(int n) {

        if(n==2)    return 1;

        if(n==3)    return 2;

        unordered\_map<int,int> mp;

        mp[2] = 2;      mp[3] = 3;

        return universe[n] = helper(n, mp);

    }

};

**T.C: - O(N \* log N)**

**S.C: - O(N)**