**MODMUL - Modular Multiplication -** [**https://www.spoj.com/problems/MODMUL/**](https://www.spoj.com/problems/MODMUL/)

**🧠 Problem Understanding**

You're given two **non-negative 64-bit integers**, a and b. You need to:

* Multiply them: a \* b
* Take the result modulo 10000007: (a \* b) % 10000007
* Print the result with a case number.

**🧩 How to Think About It**

This is a classic **modular arithmetic** problem. The key idea is:

When numbers are very large, direct multiplication might overflow. But since we only care about the result modulo a small number (like 10000007), we can use modular properties to keep numbers manageable.

However, in C++, unsigned long long can handle up to 64-bit values, so direct multiplication is safe here.

**🛠️ Solution Approach**

1. Read input until EOF (end of file).
2. For each pair (a, b):
   * Compute (a \* b) % 10000007
   * Print the result in the format: Case #i: result

**💻 C++ Implementation**

#include <iostream>

using namespace std;

const long long MOD = 10000007;

int main() {

long long a, b;

int caseNum = 1;

while (cin >> a >> b) {

long long result = (a % MOD) \* (b % MOD) % MOD;

cout << "Case #" << caseNum++ << ": " << result << endl;

}

return 0;

}

**⏱️ Time & Space Complexity**

* **Time Complexity**: O(1) per test case (just a few arithmetic operations)
* **Space Complexity**: O(1) (no extra space used)

Another Approach :

#include <bits/stdc++.h>

using namespace std;

const long long MOD = 10000007;

// Function to do (a \* b) % mod safely without overflow

long long modMul(long long a, long long b, long long mod) {

    long long res = 0;

    a %= mod;

    while (b > 0) {

        if (b & 1) { // if b is odd

            res = (res + a) % mod;

        }

        a = (a \* 2) % mod;

        b >>= 1; // divide b by 2

    }

    return res;

}

int main() {

    ios::sync\_with\_stdio(false);

    cin.tie(NULL);

    long long a, b;

    int caseNo = 1;

    while (cin >> a >> b) {

        long long ans = modMul(a, b, MOD);

        cout << "Case #" << caseNo++ << ": " << ans << "\n";

    }

    return 0;

}