// C++ program to find multiplicative modulo inverse using

// Extended Euclid algorithm.

#include<iostream>

using namespace std;

// C function for extended Euclidean Algorithm

int gcdExtended(int a, int b, int \*x, int \*y);

// Function to find modulo inverse of a

void modInverse(int a, int m)

{

    int x, y;

    int g = gcdExtended(a, m, &x, &y);

    if (g != 1)

        cout << "Inverse doesn't exist";

    else

    {

        // m is added to handle negative x

        int res = (x%m + m) % m;

        cout << "Modular multiplicative inverse is " << res;

    }

}

// C function for extended Euclidean Algorithm

int gcdExtended(int a, int b, int \*x, int \*y)

{

    // Base Case

    if (a == 0)

    {

        \*x = 0, \*y = 1;

        return b;

    }

    int x1, y1; // To store results of recursive call

    int gcd = gcdExtended(b%a, a, &x1, &y1);

    // Update x and y using results of recursive

    // call

    \*x = y1 - (b/a) \* x1;

    \*y = x1;

    return gcd;

}

// Driver Program

int main()

{

    int a = 3, m = 11;

    modInverse(a, m);

    return 0;

}