**Final Year B.Tech. (CSE) – II [ 2021-22 ]**

**Cryptograpy and Network Security Lab**

**PRN: 2019BTECS00015**

**Full name: Shraddha Sanjay Kharat**

**Batch: B1**

**Assignment No. 10**

**Title:** Chinese Remainder Theorem

**Aim:** To Demonstrate Chinese Remainder Theorem

**Theory:**

In mathematics, the Chinese remainder theorem states that if one knows the remainders of the Euclidean division of an integer n by several integers, then one can determine uniquely the remainder of the division of n by the product of these integers, under the condition that the divisors are pair wise co-prime**.**

* For example, if we know that the remainder of n divided by 3 is 2, the remainder of n divided by 5 is 3, and the remainder of n divided by 7 is 2, then without knowing the value of n, we can determine that the remainder of n divided by 105 (the product of 3, 5, and 7) is 23. Importantly, this tells us that if n is a natural number less than 105, then 23 is the only possible value of n.
* The Chinese remainder theorem is widely used for computing with large integers, as it allows replacing a computation for which one knows a bound on the size of the result by several similar computations on small integers.

**Code:**

#include<bits/stdc++.h>

using namespace std;

// returns x where (a \* x) % b == 1

int mul\_inv(int a, int b)

{

    int b0 = b, t, q;

    int x0 = 0, x1 = 1;

    if (b == 1) return 1;

    while (a > 1) {

        q = a / b;

        t = b, b = a % b, a = t;

        t = x0, x0 = x1 - q \* x0, x1 = t;

    }

    if (x1 < 0) x1 += b0;

    return x1;

}

int chinese\_remainder(int \*n, int \*a, int len)

{

    int p, i, prod = 1, sum = 0;

    for (i = 0; i <len; i++)

        prod \*= n[i];

    cout<<"The Product of Divisors is: "<<prod<<endl;

    for (i = 0; i <len; i++) {

        p = prod / n[i];

        sum += a[i] \* mul\_inv(p, n[i]) \* p;

    }

    return sum % prod;

}

int main(void)

{

    int n[] = { 3, 5, 7 };

    int r[] = { 2, 3, 2 };

    cout<<"The Divisors are: ";

    for(int i = 0;i < 3;i++)

        cout<<n[i]<<" ";

    cout<<"and their respective remainder are: ";

    for(int i = 0;i < 3;i++)

        cout<<r[i]<<" ";

    cout<<endl;

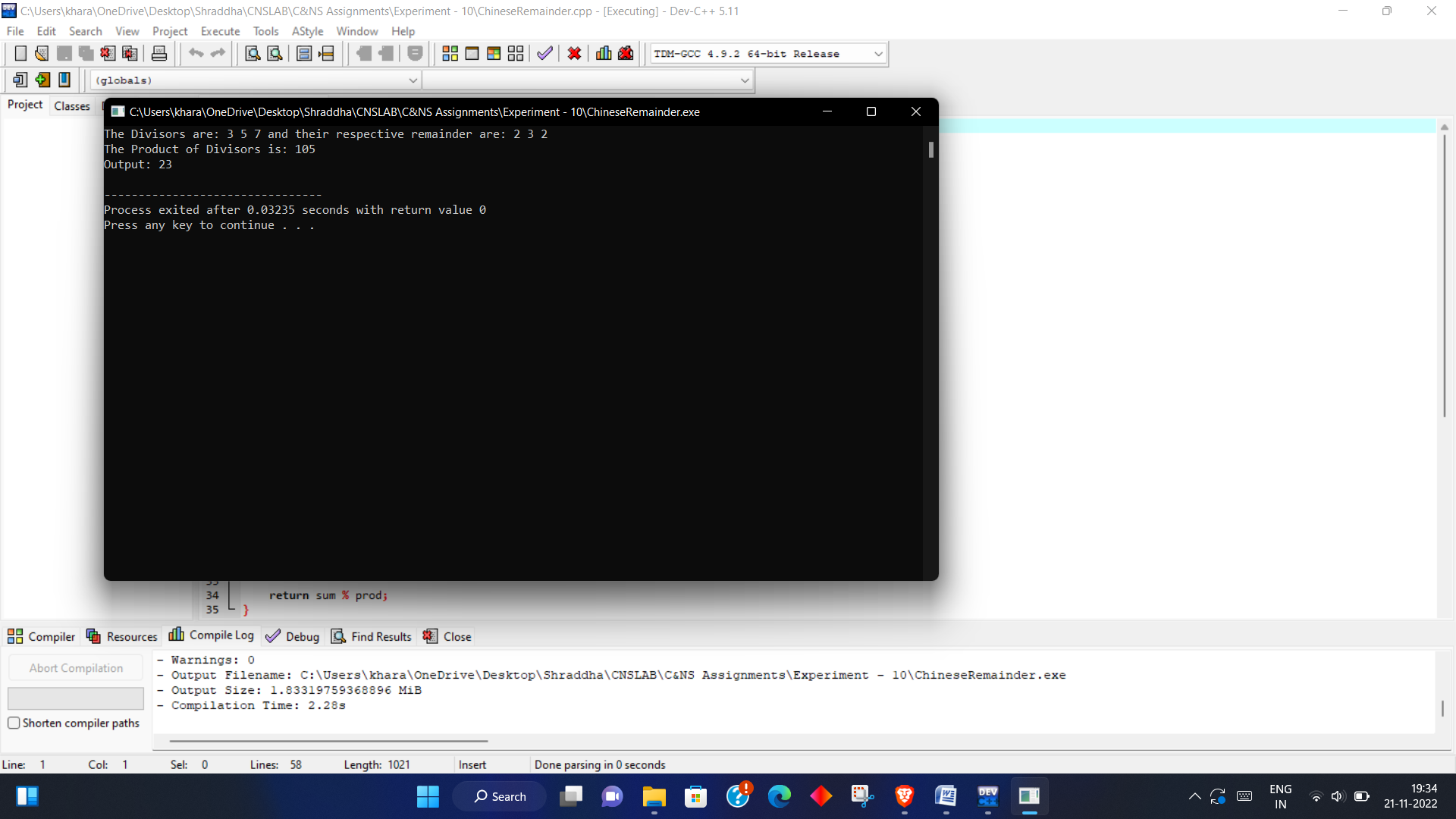
    int ans = chinese\_remainder(n, r, sizeof(n)/sizeof(n[0]));

    cout<<"Output: "<<ans<<endl;

    return 0;

}

**Output:**

****

**Conclusion:**

The Chinese remainder theorem is widely used for computing with large integers, as it allows replacing a computation for which one knows a bound on the size of the result by several similar computations on small integers.