**Euclidean And Extended Euclidean Algorithm**

**Theory:**

Euclidean Algorithm:

- The Euclidean Algorithm is a method for finding the greatest common divisor (GCD) of two integers.

- It works by repeatedly applying the division algorithm, replacing the larger number with the remainder of the division until the remainder is zero.

- The GCD is the last non-zero remainder.

- Example: GCD(48, 18) = 6, as 48 = 2 \* 18 + 12, 18 = 1 \* 12 + 6, 12 = 2 \* 6, and the remainder is 0.

Extended Euclidean Algorithm:

- The Extended Euclidean Algorithm not only finds the GCD of two integers but also computes the coefficients of Bézout's identity.

- Bézout's identity states that for integers a and b, there exist integers x and y such that ax + by = GCD(a, b).

- The Extended Euclidean Algorithm finds these values x and y.

- Example: For a = 48 and b = 18, GCD(48, 18) = 6, and the Extended Euclidean Algorithm would give you x = 1 and y = -3 because 48 \* 1 + 18 \* (-3) = 6.

The Extended Euclidean Algorithm is particularly useful in modular arithmetic and cryptographic applications for solving linear congruences and finding modular multiplicative inverses.

**Code:**

#include<iostream>

#include<bits/stdc++.h>

using namespace std;

class menu

{

    public :

   long long find\_multiplicative\_inverse(long long a, long long b) {

    long long q, r, t1 = 0, t2 = 1, t, main\_a = a;

cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

    cout << "|\tQ\t|\tA\t|\tB\t|\tR\t|\tT1\t|\tT2\t|\tT\t|\n";

  cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

    while (b > 0) {

        q = a / b;

        r = a % b;

        t = t1 -  (t2 \* q );

        cout << "|\t" << q << "\t|\t" << a << "\t|\t" << b << "\t|\t" << r << "\t|\t" << t1 << "\t|\t" << t2 << "\t|\t" << t << "\t|\n";

      cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

        a = b;

        b = r;

        t1 = t2;

        t2 = t;

    }

    cout << "|\t" << q << "\t|\t" << a << "\t|\t" << b << "\t|\t" << r << "\t|\t" << t1 << "\t|\t" << t2 << "\t|\t" << t << "\t|\n";

    cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

    if (t1 < 0) {

        t1 += main\_a;

    }

    return t1;

}

    long long find\_large\_number\_gcd(long long a,long long b)

    {

        long long q,r;

            cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

            cout<<"|\t\tQ\t\t|\t\tA\t\t|\t\tB\t\t|\t\tR\t\t|\n";

          cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

            while(b>0)

            {

                    q=a/b;

                    r=a%b;

                    cout<<"|\t\t"<<q<<"\t\t|\t\t"<<a<<"\t\t|\t\t"<<b<<"\t\t|\t\t"<<r<<"\t\t|\n";

                 cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

                    a=b;

                    b=r;

            }

             cout<<"|\t\t"<<q<<"\t\t|\t\t"<<a<<"\t\t|\t\t"<<b<<"\t\t|\t\t"<<r<<"\t\t|\n";

    cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

            cout<<endl;

            return a;

    }

};

int main()

{

    main\_menu:

    cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

    cout<<"\n1.Find Multiplicative Inverse (Extended Euclidien Algo ) \n2.Find GCD Of large numbers(Euclideian Algo ) \n";

    cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

    cout<<"Enter Choice Code :\t";

    menu object;

    int ch;

    cin>>ch;

    cout<<"\n";

    long long a,b,ans;

    switch(ch)

    {

        case 1 :

            cout<<"\nEnter  A and B ( must be A>B)  :\t";

            cin>>a>>b;

            ans=object.find\_multiplicative\_inverse(a,b);

            cout<<"Multiplicative Inverse Of  "<<a<<"\tAnd "<<b<<"\t :\t"<<ans<<endl;

            goto main\_menu;

        case 2:

            cout<<"\nEnter  A and B  :\t";

            cin>>a>>b;

             ans=object.find\_large\_number\_gcd(a,b);

            cout<<"\nGCD Of   Of  "<<a<<"\tAnd "<<b<<"\t :\t"<<ans<<endl;

            goto main\_menu;

        default:

            cout<<"Invalid Input !";

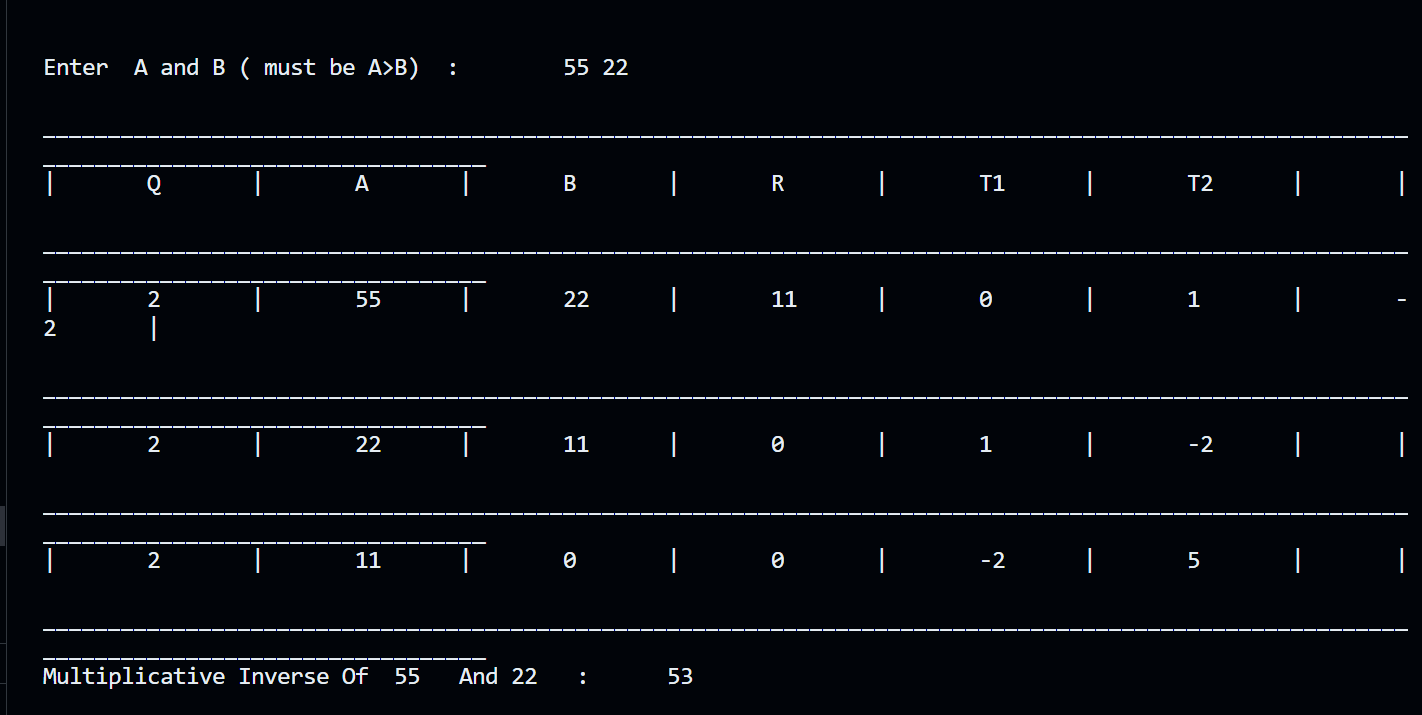
            break;

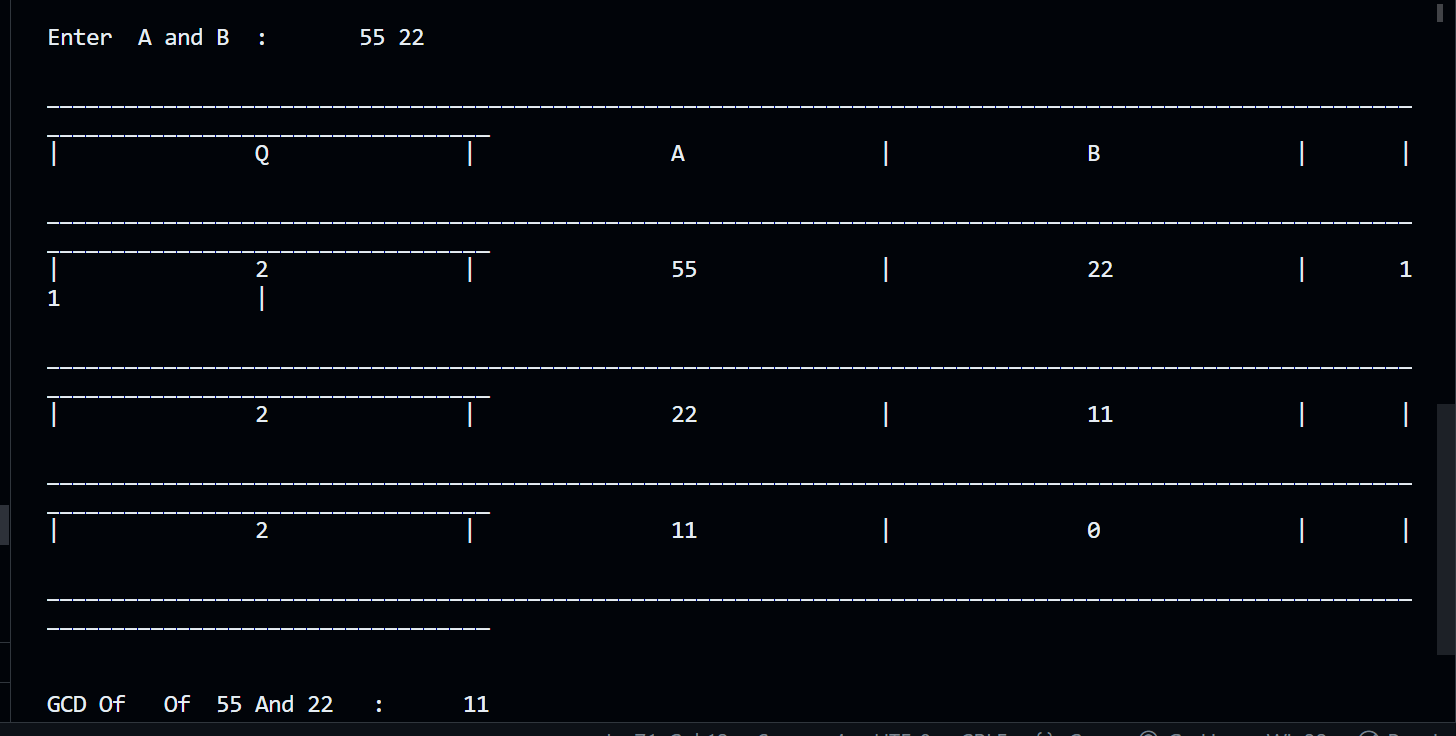
    }

    return 0;

}

**Screenshot:**

****

****