**Garvin Chanderia**

**2203132**

**LY-ISA1**

**Experiment No:2**

**Title:**  **Extended Euclidean Algorithm**

**Problem Statement:** Implement Euclidean and Extended Euclidean algorithm to find out GCD and solve the inverse mod problem.

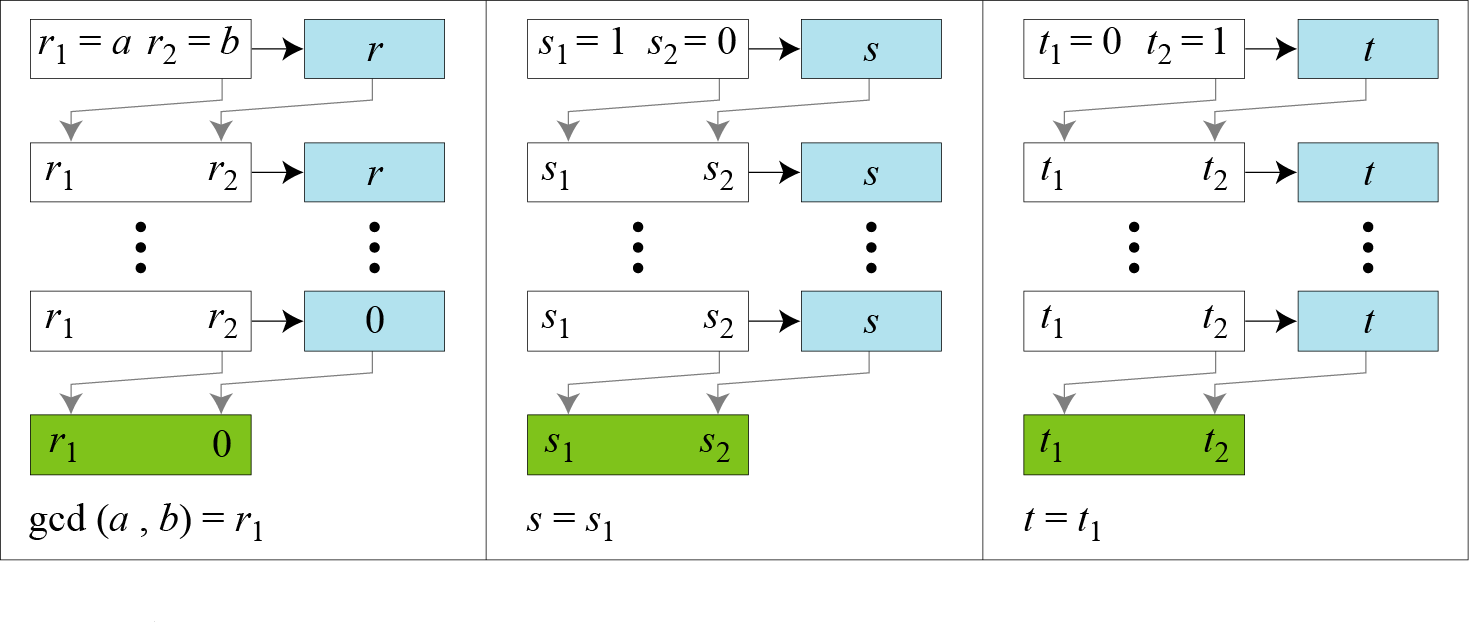
**Aim:**  To study Euclidian & Extended Euclidian algorithm

**Theory:**

The extended Euclidean algorithm is an extension to the Euclidean algorithm. Besides finding the greatest common divisor of integers *a* and *b*, as the Euclidean algorithm does, it also finds integers *x* and *y* (one of which is typically negative).

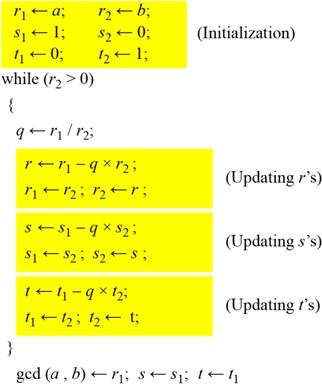
*ax* + *by* = gcd (*a*, *b*) or s*a* + t*b* = gcd (*a*, *b*)

The extended Euclidean algorithm is particularly useful when *a* and *b* are coprime, since *x* is the multiplicative inverse of *a* modulo *b*, and *y* is the multiplicative inverse of *b* modulo *a*.

 **Figure:** Extended Euclid’s Algorithm Process

Algorithm:

Extend the algorithm to compute the integer coefficients x and y such that gcd (a, b) = ax + by

Extended-Euclid (a, b)

Example:

GCD (161, 28)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *q* | *r1* | *r2* | *r* | *s1* | *s2* | *s* | *t1* | *t2* | *t* |
| 5 | 161 | 28 | 21 | 1 | 0 | 1 | 0 | 1 | -5 |
| 1 | 28 | 21 | 7 | 0 | 1 | -1 | 1 | -5 | 6 |
| 3 | 21 | 7 | 0 | 1 | -1 | 4 | -5 | 6 | -23 |
|  | 7 | 0 |  | -1 | 4 |  | 6 | -23 |  |

Here the GCD value we are getting as 7. Value of s is -1 and value of t is 6. If we put these values into equation,

*ax* + *by* = gcd (*a*, *b*) 161 \* (-1) + 28 \* (6) = 7

This satisfies the equation for Extended Euclidean Algorithm

**Algorithm/Pseudocode:**

# Python3 program to demonstrate Basic Euclidean Algorithm

# Function to return gcd of a and b

def gcd(a, b):

if a == 0:

return b

return gcd(b % a, a)

# Driver code

if \_\_name\_\_ == "\_\_main\_\_":

a = 10

b = 15

print("gcd(", a, ",", b, ") = ", gcd(a, b))

a = 35

b = 10

print("gcd(", a, ",", b, ") = ", gcd(a, b))

a = 31

b = 2

print("gcd(", a, ",", b, ") = ", gcd(a, b))

Output:

GCD(10, 15) = 5

GCD(35, 10) = 5

GCD(31, 2) = 1

**CODE:**

#include <iostream>

using namespace std;

int main() {

int t1 =0;

int t2 = 1;

int t;

int a,b,q,rem;

cin>>a>>b;

if(a>b) {

while(b!=0) {

q = a/b;

rem = a%b;

t = t1 - (q\*t2);

a = b;

b = rem;

t1 =t2;

t2 = t;

}

}

cout<<"Remainder and quotient:"<<rem<<" & "<<q<<endl;

cout<<"Value of a and b: "<<a<<" mod "<<b<<endl;

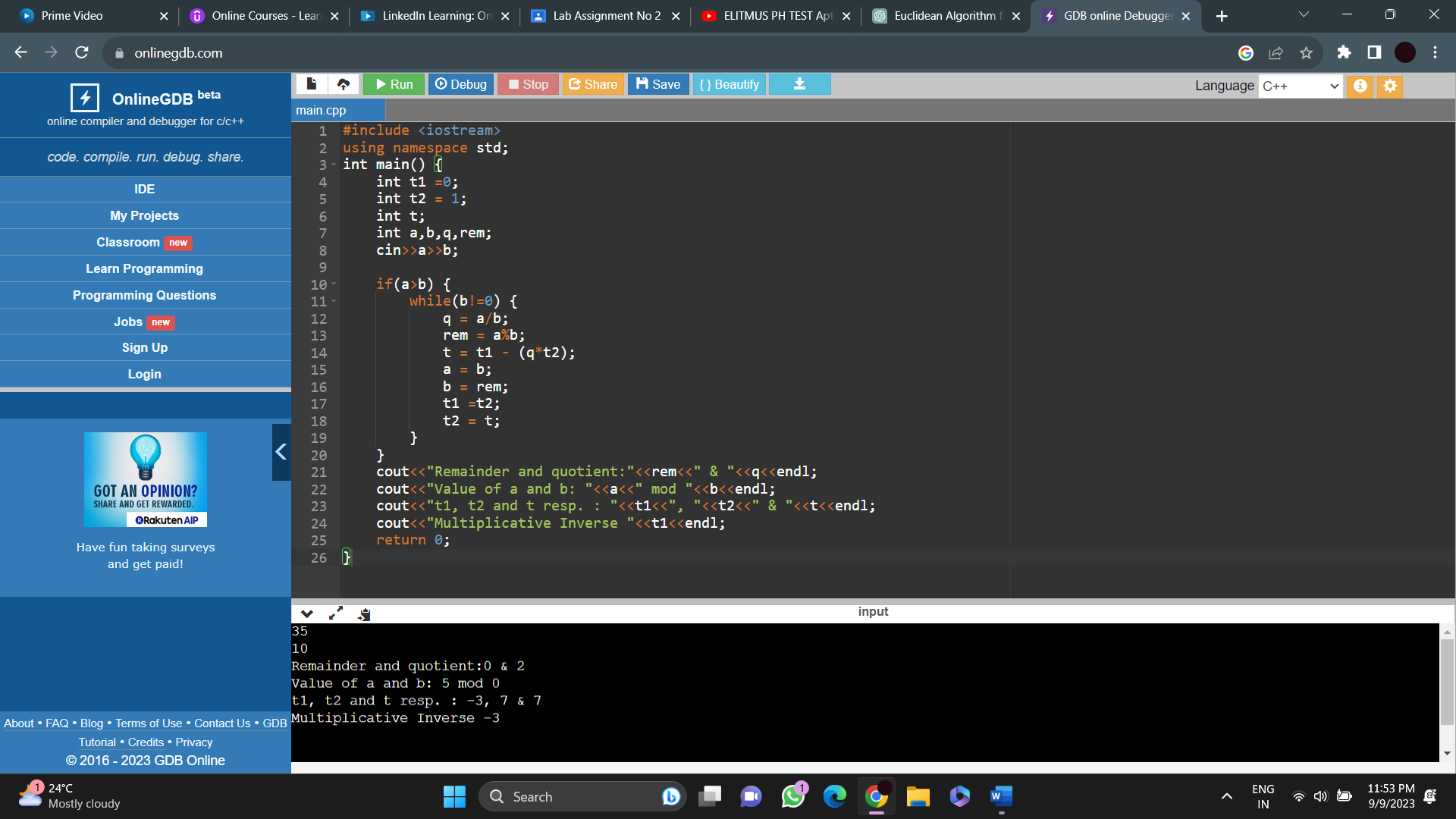
cout<<"t1, t2 and t resp. : "<<t1<<", "<<t2<<" & "<<t<<endl;

cout<<"Multiplicative Inverse "<<t1<<endl;

return 0;

}

**OUTPUT:**



**CONCLUSION:**

We have studied and implemented the Extended Euclidian algorithm.

