**Experiment-1**

**Aim**: Implementation of Extended Euclidean algorithm.

**Theory**:

In arithmetic and computer programming, the extended Euclidean algorithm is an extension to the Euclidean algorithm, and computes, in addition to the greatest common divisor (gcd) of integers a and b, also the coefficients of Bézout's identity, which are integers x and y such that

ax+by = gcd(a,b).

This is a certifying algorithm, because the gcd is the only number that can simultaneously satisfy this equation and divide the inputs. It allows one to compute also, with almost no extra cost, the quotients of a and b by their greatest common divisor.

The extended Euclidean algorithm is particularly useful when a and b are coprime. With that provision, x is the modular multiplicative inverse of a modulo b, and y is the modular multiplicative inverse of b modulo a. In particular, the computation of the modular multiplicative inverse is an essential step in the derivation of key-pairs in the RSA public-key encryption method.

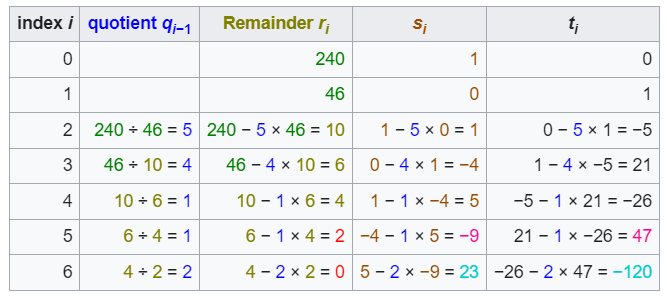
**Procedure:**

The Euclidean Algorithm for finding GCD(A,B) is as follows:

* If A = 0 then GCD(A,B)=B, since the GCD(0,B)=B, and we can stop.
* If B = 0 then GCD(A,B)=A, since the GCD(A,0)=A, and we can stop.
* Write A in quotient remainder form (A = B⋅Q + R)
* Find GCD(B,R) using the Euclidean Algorithm since GCD(A,B) = GCD(B,R)

**Solved Example:**

The following table shows how the extended Euclidean algorithm proceeds with input 240 and 46.



The greatest common divisor is the last non zero entry, 2 in the column "remainder". The computation stops at row 6, because the remainder in it is 0. Bézout coefficients appear in the last two entries of the second-to-last row. In fact, it is easy to verify that −9 × 240 + 47 × 46 = 2. Finally the last two entries 23 and −120 of the last row are, up to the sign, the quotients of the input 46 and 240 by the greatest common divisor 2.

**Program:**

import java.util.Scanner;

public class ExtendedEuclidean {

    public static void main(String[] args) {

        System.out.println("------Extended Euclidean Algorithm------");

        Scanner input = new Scanner(System.in);

        System.out.print("Enter the first number: ");

        int r1 = input.nextInt();

        System.out.print("Enter the second number: ");

        int r2 = input.nextInt();

        int s1 = 1, s2 = 0, t1 = 0, t2 = 1, q = 0, r = 0, s = 0, t = 0, a = r1, b = r2;

        System.out.printf("%-4s %-6s %-6s %-6s %-6s %-6s %-6s %-6s %-6s %-6s\n", "q", "r1", "r2", "r", "s1", "s2", "s",

                "t1", "t2", "t");

        while (r2 > 0) {

            q = r1 / r2;

            r = r1 % r2;

            s = s1 - q \* s2;

            t = t1 - q \* t2;

            System.out.printf("%-4d %-6d %-6d %-6d %-6d %-6d %-6d %-6d %-6d %-6d\n", q, r1, r2, r, s1, s2, s, t1, t2,

                    t);

            r1 = r2;

            r2 = r;

            s1 = s2;

            s2 = s;

            t1 = t2;

            t2 = t;

        }

        System.out.println("\nThe GCD is: " + r1);

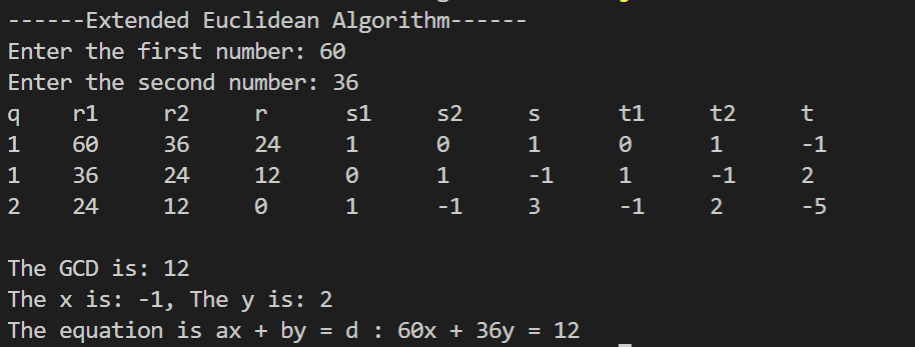
        System.out.println("The x is: " + s1 + ", The y is: " + t1);

        System.out.printf("The equation is ax + by = d : %dx + %dy = %d ", a, b, r1);

    }

}

**Output:**

****