| | |
|---|---|
| **Name** | Piyush Rathi |
| **UID no.** | 2023300186 |
| **Experiment No.** | 1a |

| | |
|---|---|
| **AIM:** | To implement i) Euclidean algorithm for GCD, ii) Extended Euclidean Algorithm for GCD and (s,t) pair and iii) Euclidean Algorithm to find Multiplicative Inverse (MI) |
| **THEORY:** | A) Basic Euclidean algorithm [Tabular Method] for GCD – This algorithm has three columns: q, r1, r2 and r. The first column (q) is for quotient (i.e. $\lfloor r1/r2 \rfloor$). The next two columns are r1 and r2 represents two integer numbers in which r1 is largest number and r2 is second integer number. The last column r is remainder of division a/b (i.e. r1 mod r2). This tabular method is iterated by shifting r2 and r to r1 and r2 respectively. This process is repeated until no further division r1/r2 is not possible. The last updated r1 represents GCD of two input numbers.<br><br>B) Extended Euclidean algorithm [Tabular Method] for GCD and (s,t) – This algorithm has additional six columns to Basic Euclidean algorithm. The additional six columns are s1, s2, s, t1, t2 and t. The initial values of s1 and s2 are 1 and 0 respectively. The initial values of t1 and t2 are 0 and 1 respectively. To compute q, r, s and t, this algorithm uses the following three equations: $q = \lfloor r1/r2 \rfloor$, r= r1 mod r2, s=s1-s2*q and t=t1-t2*q. The tabular Extended Euclidean algorithm is iterated by performing six shifting (i.e. r2, r, s2, s, t2 and t to r1, r2, s1, s2, t1 and t2 respectively) and computing q, r, s and t in each iteration. This process is repeated until no further division r1/r2 is not possible. The last updated r1 s1, and t1 represents GCD, s and t respectively.<br><br>C) Euclidean algorithm [Tabular Method] for GCD and MI – This algorithm |

| | |
|---|---|
| | uses only 7 columns of Extended<br>Euclidean algorithm which are q, r1, r2, r, t1, t2 and t. This algorithm is also iterated by performing four shifts and<br>computation of q, r and t in each iteration. This process is repeated until no further division r1/r2 is not possible. The<br>last updated r1 and t1 represents GCD and MI respectively. |
| **Program 1** | |
| **PROBLEM STATEMENT :** | Problem Definition – The greatest common divisor GCD(a, b) of two natural numbers a and b is the greatest number<br>that divides both a and b. This number plays an important role in the number theory and public cryptography. The<br>algorithm is based on the following fact (assume $a \geq b > 0$): $a = q \cdot b + r$; $0 \leq r < b$. Here q is the quotient, i.e. $\lfloor a/b \rfloor$,<br>and r is the remainder r = a mod b. There are three algorithms to find GCD and some additional information.<br><br>A) Basic Euclidean algorithm [Tabular Method] for GCD – This algorithm has three columns: q, r1, r2 and r. The first<br>column (q) is for quotient (i.e. $\lfloor r1/r2 \rfloor$). The next two columns are r1 and r2 represents two integer numbers in<br>which r1 is largest number and r2 is second integer number. The last column r is remainder of division a/b (i.e. r1<br>mod r2). This tabular method is iterated by shifting r2 and r to r1 and r2 respectively. This process is repeated until<br>no further division r1/r2 is not possible. The last updated r1 represents GCD of two input numbers.<br>B) Extended Euclidean algorithm [Tabular Method] for GCD and (s,t) – This algorithm has additional six columns to<br>Basic Euclidean algorithm. The additional six columns are s1, s2, s, t1, t2 and t. The initial values of s1 and s2 are 1<br>and 0 respectively. The initial values of t1 and t2 are 0 and 1 respectively. To compute q, r, s and t, this algorithm<br>uses the following three equations: $q = \lfloor r1/r2 \rfloor$, r= r1 mod r2, s=s1-s2*q and t=t1-t2*q. The tabular Extended<br>Euclidean algorithm is iterated by performing six shifting (i.e. r2, r, s2, s, t2 and t to r1, r2, s1, s2, t1 and t2<br>respectively) and computing q, r, s and t in each iteration. This process is repeated until no further division r1/r2 is |

| | |
|---|---|
| | not possible. The last updated r1 s1, and t1 represents GCD, s and t respectively.<br>C) Euclidean algorithm [Tabular Method] for GCD and MI – This algorithm uses only 7 columns of Extended<br>Euclidean algorithm which are q, r1, r2, r, t1, t2 and t. This algorithm is also iterated by performing four shifts and<br>computation of q, r and t in each iteration. This process is repeated until no further division r1/r2 is not possible. The<br>last updated r1 and t1 represents GCD and MI respectively. |
| **PROGRAM:** | package Exp1_Euclidian;<br>import java.util.Scanner;<br><br>public class menu {<br><br>// A) Basic Euclidean Algorithm (Tabular Method)<br>static void basicEuclidean(int a, int b) {<br>int r1 = a, r2 = b;<br>System.out.println("q\tr1\tr2\tr");<br>while (r2 != 0) {<br>int q = r1 / r2;<br>int r = r1 % r2;<br>System.out.println(q + "\t" + r1 + "\t" + r2 + "\t" + r);<br>r1 = r2;<br>r2 = r;<br><br>System.out.println("GCD = " + r1);<br>}<br><br>// B) Extended Euclidean Algorithm (Tabular Method)<br>static void extendedEuclidean(int a, int b) {<br>int r1 = a, r2 = b;<br>int s1 = 1, s2 = 0;<br>int t1 = 0, t2 = 1;<br><br>System.out.println("q\tr1\tr2\tr\ts1\ts2\ts\tt1\tt2\tt");<br>while (r2 != 0) {<br>int q = r1 / r2;<br>int r = r1 % r2;<br>int s = s1 - s2 * q; |

```
        int t = t1 - t2 * q;

        System.out.println(q + "\t" + r1 + "\t" + r2 + "\t" + r +
                "\t" + s1 + "\t" + s2 + "\t" + s +
                "\t" + t1 + "\t" + t2 + "\t" + t);

        // Shift
        r1 = r2;
        r2 = r;
        s1 = s2;
        s2 = s;
        t1 = t2;
        t2 = t;
    }
    System.out.println("GCD = " + r1 + ", s = " + s1 + ", t = " + t1);
}

// C) Euclidean Algorithm for GCD and Multiplicative Inverse (MI)
static void multiplicativeInverse(int a, int b) {
    int r1 = a, r2 = b;
    int t1 = 0, t2 = 1;

    System.out.println("q\tr1\tr2\tr\tt1\tt2\tt");
    while (r2 != 0) {
        int q = r1 / r2;
        int r = r1 % r2;
        int t = t1 - t2 * q;

        System.out.println(q + "\t" + r1 + "\t" + r2 + "\t" + r +
                "\t" + t1 + "\t" + t2 + "\t" + t);

        // Shift
        r1 = r2;
        r2 = r;
        t1 = t2;
        t2 = t;
    }
    System.out.println("GCD = " + r1);
    if (r1 == 1) {
        int mi = (t1 % b + b) % b; // ensure positive
```

```
            System.out.println("Multiplicative Inverse of " + a + " mod " + b + "
= " + mi);
        } else {
            System.out.println("No Multiplicative Inverse exists (since GCD ≠
1)");
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (true) {
            System.out.println("\n--- Menu ---");
            System.out.println("1. Basic Euclidean Algorithm (GCD)");
            System.out.println("2. Extended Euclidean Algorithm (GCD, s, t)");
            System.out.println("3. Multiplicative Inverse");
            System.out.println("4. Quit");
            System.out.print("Enter choice: ");
            int choice = sc.nextInt();

            if (choice == 4) {
                System.out.println("Exiting program...");
                break;
            }

            System.out.print("Enter first number (a): ");
            int a = sc.nextInt();
            System.out.print("Enter second number (b): ");
            int b = sc.nextInt();

            switch (choice) {
                case 1:
                    basicEuclidean(a, b);
                    break;
                case 2:
                    extendedEuclidean(a, b);
                    break;
                case 3:
                    multiplicativeInverse(a, b);
                    break;
                default:
```

```
                System.out.println("Invalid choice!");
            }
        }
        sc.close();
    }
}
```

**RESULT:**

```
PS C:\Users\Piyush\Documents\College\Cryptography and Network Security> java Exp1_Euclidian.menu

--- Menu ---
1. Basic Euclidean Algorithm (GCD)
2. Extended Euclidean Algorithm (GCD, s, t)
3. Multiplicative Inverse
4. Quit
Enter choice: 1
Enter first number (a): 8
Enter second number (b): 12
q       r1      r2      r
0       8       12      8
GCD = 12
1       12      8       4
GCD = 8
2       8       4       0
GCD = 4
```

```
--- Menu ---
1. Basic Euclidean Algorithm (GCD)
2. Extended Euclidean Algorithm (GCD, s, t)
3. Multiplicative Inverse
4. Quit
Enter choice: 2
Enter first number (a): 340
Enter second number (b): 420
q       r1      r2      r       s1      s2      s       t1      t2      t
0       340     420     340     1       0       1       0       1       0
1       420     340     80      0       1       -1      1       0       1
4       340     80      20      1       -1      5       0       1       -4
4       80      20      0       -1      5       -21     1       -4      17
GCD = 20, s = 5, t = -4
```

```
--- Menu ---
1. Basic Euclidean Algorithm (GCD)
2. Extended Euclidean Algorithm (GCD, s, t)
3. Multiplicative Inverse
4. Quit
Enter choice: 3
Enter first number (a): 120
Enter second number (b): 325
q       r1      r2      r       t1      t2      t
0       120     325     120     0       1       0
2       325     120     85      1       0       1
1       120     85      35      0       1       -1
2       85      35      15      1       -1      3
2       35      15      5       -1      3       -7
3       15      5       0       3       -7      24
GCD = 5
No Multiplicative Inverse exists (since GCD â?  1)
```

```
--- Menu ---
1. Basic Euclidean Algorithm (GCD)
2. Extended Euclidean Algorithm (GCD, s, t)
3. Multiplicative Inverse
4. Quit
Enter choice: 3
Enter first number (a): 20
Enter second number (b): 23
q       r1      r2      r       t1      t2      t
0       20      23      20      0       1       0
1       23      20      3       1       0       1
6       20      3       2       0       1       -6
1       3       2       1       1       -6      7
2       2       1       0       -6      7       -20
GCD = 1
Multiplicative Inverse of 20 mod 23 = 7
```

```
--- Menu ---
1. Basic Euclidean Algorithm (GCD)
2. Extended Euclidean Algorithm (GCD, s, t)
3. Multiplicative Inverse
4. Quit
Enter choice: 4
Exiting program...
PS C:\Users\Piyush\Documents\College\Cryptography and Network Security>
```

| | |
|---|---|
| **CONCLUSION:** | In this experiment, we explored three variants of the Euclidean algorithm using the tabular method: the Basic Euclidean Algorithm, the Extended Euclidean Algorithm, and the Euclidean Algorithm for Modular Inverse (MI). The Basic Euclidean Algorithm efficiently computes the greatest common divisor (GCD) of two numbers through repeated division. The Extended Euclidean Algorithm extends this approach by also calculating the coefficients<br>*s*<br>s and<br>*t*<br>t that satisfy Bézout's identity, which has important applications in number theory and cryptography. Finally, the Modular Inverse algorithm, derived from the extended method, focuses on computing the modular inverse, which is essential in public-key cryptography such as RSA.<br><br>Overall, these tabular methods not only simplify the step-by-step computation but also highlight the mathematical foundation of division, remainders, and modular arithmetic. Through this project, we conclude that the Euclidean family of algorithms remains fundamental for solving problems in mathematics and computer science, especially in domains like cryptography, coding theory, and algorithm design. |