Program: 4

Extended Euclidean algorithm

Date:

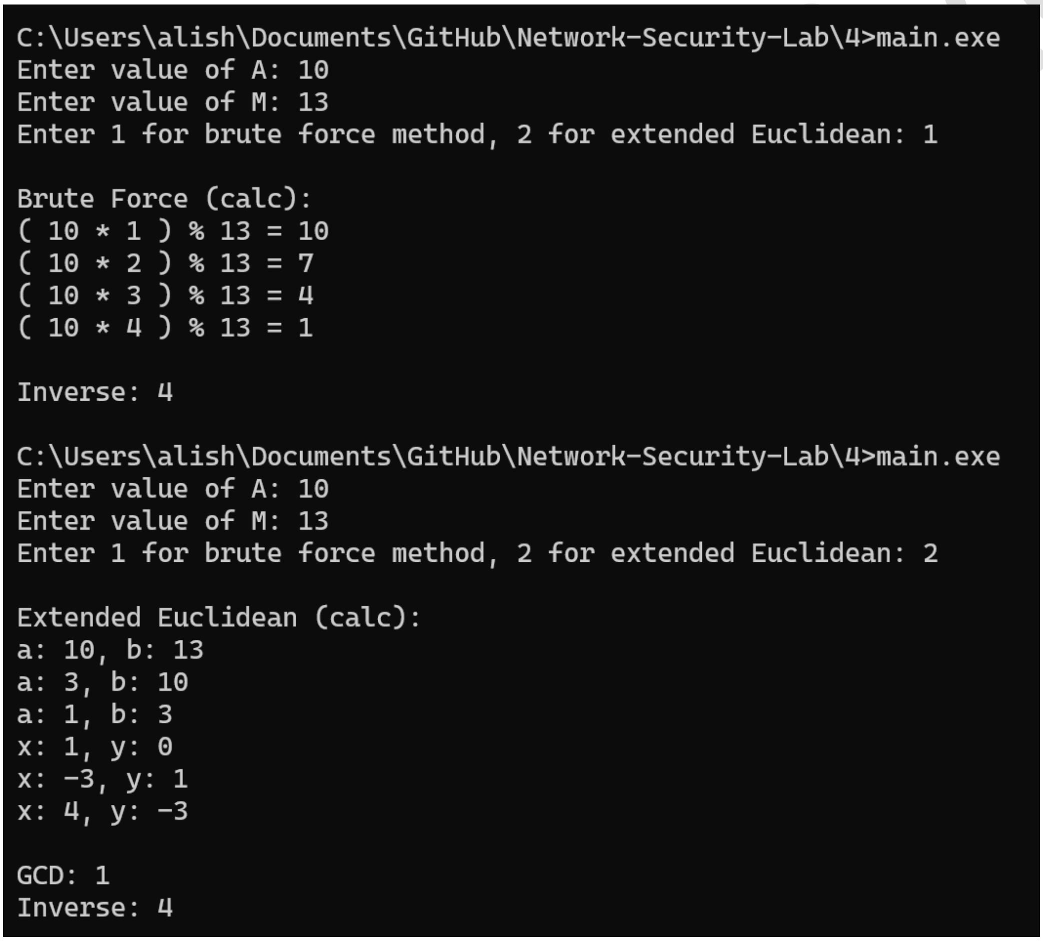
AIM

ALGORITHM

CODE

```
#include <iostream>
using namespace std;

int extendedEuclidean(int a, int b, int &x, int &y) {
    if (a == 0) {
        x = 0;
        y = 1;
        return b;
    }
    cout << "a: " << a << ", b: " << b << endl;
    int x1, y1;
    int gcd = extendedEuclidean(b%a, a, x1, y1);
    x = y1 - (b/a) * x1;
    y = x1;
    cout << "x: " << x << ", y: " << y << endl;
    return gcd;
}

int multiplicativeInverseBF(int A, int M) {
    cout << "\nBrute Force (calc): " << endl;
    for (int X=1; X<M; X++) {
        int res = (A * X) % M;
        cout << "( " << A << " * " << X << " ) % " << M << " = " << res << endl;
        if (res == 1) {
            return X;
        }
    }
    return -1;
}

int multiplicativeInverseUsingExtendedEuclidean(int A, int M) {
    int x, y;
    cout << "\nExtended Euclidean (calc): " << endl;
    int gcd = extendedEuclidean(A, M, x, y);
    cout << "\nGCD: " << gcd << endl;
    if (gcd != 1) {
        return -1;
    }
    int inverse = (x % M + M) % M;
    return inverse;
}

int main() {
    int A, M, choice;
    cout << "Enter value of A: ";
    cin >> A;
    cout << "Enter value of M: ";
    cin >> M;
    while (true) {
        cout << "Enter 1 for brute force method, 2 for extended Euclidean: ";
        cin >> choice;
        if (choice != 1 && choice != 2) {
            cout << "Invalid choice" << endl;
            continue;
        } else {
            break;
        }
    }
    if (choice == 1) {
        int inverse = multiplicativeInverseBF(A, M);
        if (inverse == -1) {
            cout << "Multiplicative inverse doesn't exist." << endl;
        } else {
            cout << "\nInverse: " << inverse << endl;
        }
    } else {
        int inverse = multiplicativeInverseUsingExtendedEuclidean (A, M);
        if (inverse == -1) {
            cout << "Multiplicative inverse doesn't exist." << endl;
        } else {
            cout << "Inverse: " << inverse << endl;
        }
    }
    return 0;
}
```

OUTPUT



```
C:\Users\alish\Documents\GitHub\Network-Security-Lab\4>main.exe
Enter value of A: 10
Enter value of M: 13
Enter 1 for brute force method, 2 for extended Euclidean: 1

Brute Force (calc):
( 10 * 1 ) % 13 = 10
( 10 * 2 ) % 13 = 7
( 10 * 3 ) % 13 = 4
( 10 * 4 ) % 13 = 1

Inverse: 4

C:\Users\alish\Documents\GitHub\Network-Security-Lab\4>main.exe
Enter value of A: 10
Enter value of M: 13
Enter 1 for brute force method, 2 for extended Euclidean: 2

Extended Euclidean (calc):
a: 10, b: 13
a: 3, b: 10
a: 1, b: 3
x: 1, y: 0
x: -3, y: 1
x: 4, y: -3

GCD: 1
Inverse: 4
```

RESULT

Thus, the program to compute Modular Multiplicative Inverse of two numbers is executed successfully.