**Final Year B.Tech. (CSE) – VII [ 2024-25]**

**6CS451: Cryptography and Network Security Lab (C&NS Lab)**

**Date: 12/08/2024**

**Assignment 3**

**PRN:** 21510017  **Name:** Onkar Anand Yemul

1. **Implementation of Euclidean and Extended Euclidean Algorithm**

**Ans:**

The Euclidean and Extended Euclidean algorithms are essential for finding the greatest common divisor (GCD) of two integers. The Extended Euclidean algorithm also finds the coefficients of Bézout's identity, which are useful in solving linear Diophantine equations and in modular arithmetic.

**Euclidean Algorithm**

The Euclidean algorithm finds the GCD of two numbers by repeatedly applying the following rule: gcd(a, b) = gcd(b, a % b) until b becomes zero. The GCD is then the non-zero remainder.

**Extended Euclidean Algorithm**

The Extended Euclidean algorithm not only computes the GCD of two integers a and b, but also finds integers x and y such that ax + by = gcd(a, b).

**Python Code:**

def euclidean\_algorithm(a, b):

    """

    Compute the GCD of a and b using the Euclidean algorithm.

    Parameters:

    a (int): First integer.

    b (int): Second integer.

    Returns:

    int: The GCD of a and b.

    """

    while b != 0:

        a, b = b, a % b

    return a

def extended\_euclidean\_algorithm(a, b):

    """

    Compute the GCD of a and b, as well as the coefficients x and y

    such that ax + by = gcd(a, b) using the Extended Euclidean algorithm.

    Parameters:

    a (int): First integer.

    b (int): Second integer.

    Returns:

    tuple: (gcd, x, y) where gcd is the GCD of a and b, and x, y are

    the coefficients of Bézout's identity.

    """

    if b == 0:

        return a, 1, 0

    else:

        gcd, x1, y1 = extended\_euclidean\_algorithm(b, a % b)

        x = y1

        y = x1 - (a // b) \* y1

        return gcd, x, y

def main():

    """

    The main function to run the program.

    """

    while True:

        print("\nEuclidean and Extended Euclidean Algorithm")

        print("1. Compute GCD using Euclidean Algorithm")

        print("2. Compute GCD and coefficients using Extended Euclidean Algorithm")

        print("3. Exit")

        choice = input("Enter your choice: ")

        if choice == '1':

            a = int(input("\nEnter the first integer (a): "))

            b = int(input("Enter the second integer (b): "))

            gcd = euclidean\_algorithm(a, b)

            print(f"\nGCD of {a} and {b} is: {gcd}")

        elif choice == '2':

            a = int(input("\nEnter the first integer (a): "))

            b = int(input("Enter the second integer (b): "))

            gcd, x, y = extended\_euclidean\_algorithm(a, b)

            print(f"\nGCD of {a} and {b} is: {gcd}")

            print(f"Coefficients x and y are: x = {x}, y = {y}")

            print(f"\nBézout's identity: {a}\*({x}) + {b}\*({y}) = {gcd}")

        elif choice == '3':

            print("Exiting the program.")

            break

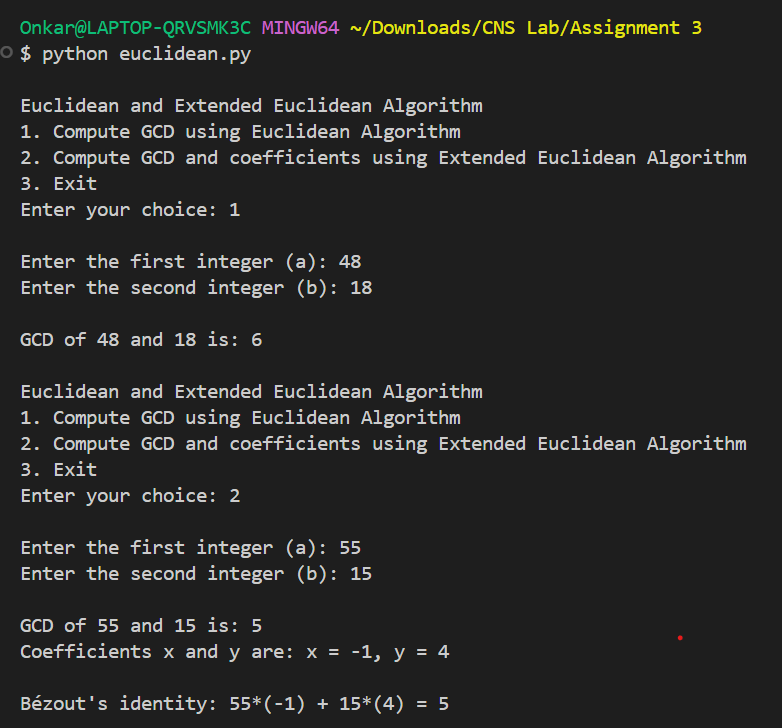
        else:

            print("Invalid choice. Please try again.")

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

    main()

**Output:**



This implementation of the Euclidean and Extended Euclidean algorithms is fundamental in cryptography, number theory, and algorithms related to modular arithmetic.