AHSANULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY

Department of Computer Science and Engineering

Program: Bachelor of Science in Computer Science and Engineering

Course Code: CSE 4174
Course Title: Cyber Security Lab
Academic Semester: Spring 2023

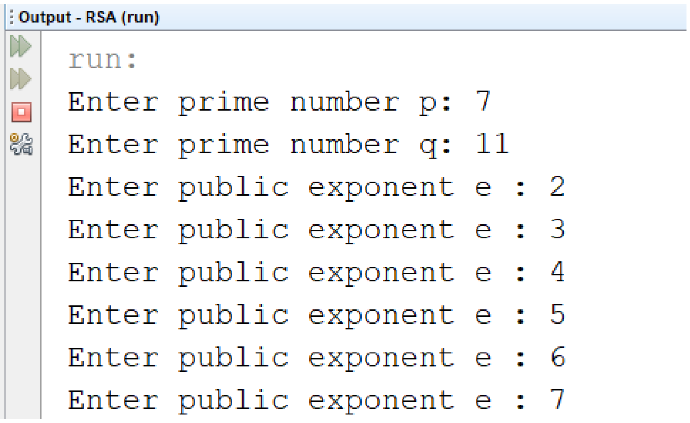
Assignment Topic: RSA (Rivest-Shamir-Adleman) Algorithm

Submitted on: 11/29/2023

Submitted by
Name: Sowppnil Roy
Student ID: 20200104071
Lab Section: B1

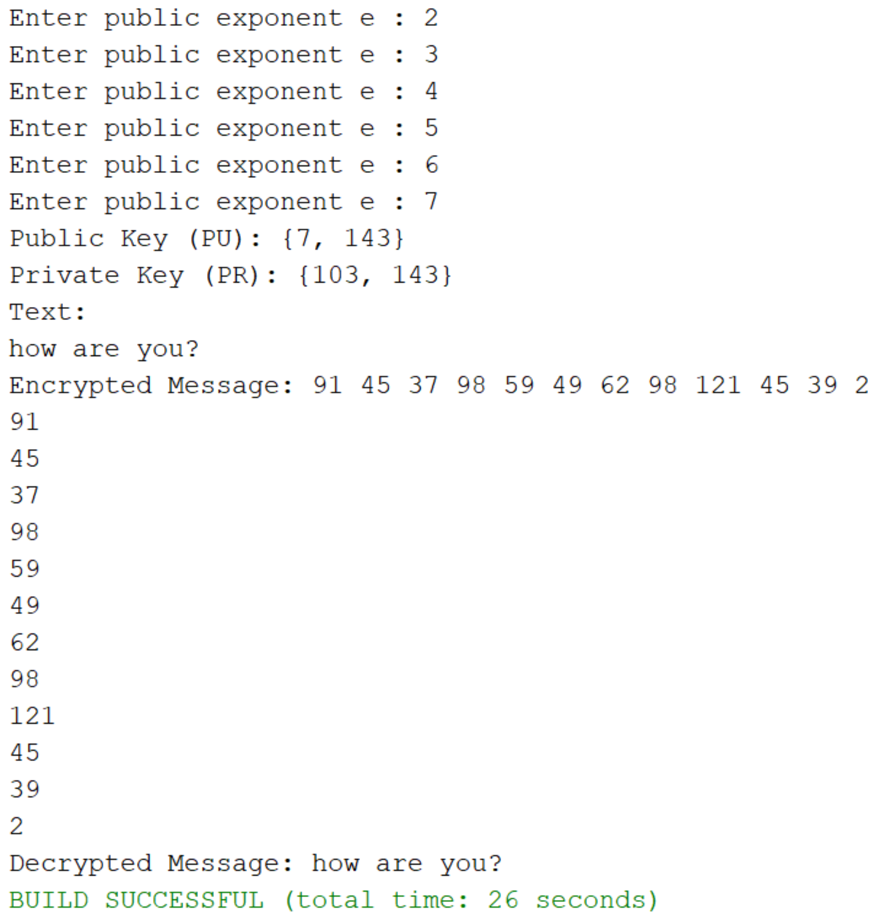
```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

package rsa;
import java.util.Scanner;
import java.math.BigInteger;
/**
 *
 * @author HP
 */
public class RSA {
public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter prime number p: ");
        BigInteger p = new BigInteger(scanner.nextLine());

        System.out.print("Enter prime number q: ");
        BigInteger q = new BigInteger(scanner.nextLine());

        BigInteger n = p.multiply(q);
        BigInteger phiN =
p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));

        BigInteger e;
        do {
            System.out.print("Enter public exponent e : ");
            e = new BigInteger(scanner.nextLine());
        } while (!e.gcd(phiN).equals(BigInteger.ONE));

        BigInteger d = e.modInverse(phiN);

        System.out.println("Public Key (PU): {" + e + ", " + n + "}");
        System.out.println("Private Key (PR): {" + d + ", " + n + "}");

        System.out.println("Text: ");
        String inputText = scanner.nextLine();

        BigInteger[] numericMessage = stringToNumeric(inputText);

        BigInteger[] encryptedMessage = new BigInteger[numericMessage.length];
        for (int i = 0; i < numericMessage.length; i++) {
            encryptedMessage[i] = numericMessage[i].modPow(e, n);
        }

        System.out.print("Encrypted Message: ");
        for (BigInteger value : encryptedMessage) {
            System.out.print(value + " ");
        }
        System.out.println();
        BigInteger[] decryptedMessage = new
BigInteger[encryptedMessage.length];
        for (int i = 0; i < encryptedMessage.length; i++) {
            decryptedMessage[i] = encryptedMessage[i].modPow(d, n);
            System.out.println(encryptedMessage[i] + " " );
        }

        String decryptedText = numericToString(decryptedMessage);
        System.out.println("Decrypted Message: " + decryptedText);

        scanner.close();
    }
    /**
     * @param args the command line arguments
     */
        private static BigInteger[] stringToNumeric(String input) {
        BigInteger[] numericValues = new BigInteger[input.length()];
        for (int i = 0; i < input.length(); i++) {
            numericValues[i] = BigInteger.valueOf(input.charAt(i));
        }
        return numericValues;
    }

    private static String numericToString(BigInteger[] numericValues) {
        StringBuilder result = new StringBuilder();
        for (BigInteger value : numericValues) {
            result.append((char) value.intValue());
        }
        return result.toString();
    }
}
```

input:



```
run:
Enter prime number p: 7
Enter prime number q: 11
Enter public exponent e : 2
Enter public exponent e : 3
Enter public exponent e : 4
Enter public exponent e : 5
Enter public exponent e : 6
Enter public exponent e : 7
```

Output:



```
Enter public exponent e : 2
Enter public exponent e : 3
Enter public exponent e : 4
Enter public exponent e : 5
Enter public exponent e : 6
Enter public exponent e : 7
Public Key (PU): {7, 143}
Private Key (PR): {103, 143}
Text:
how are you?
Encrypted Message: 91 45 37 98 59 49 62 98 121 45 39 2
91
45
37
98
59
49
62
98
121
45
39
2
Decrypted Message: how are you?
BUILD SUCCESSFUL (total time: 26 seconds)
```