جامعة الأميرة
نورة بنت عبد الرحمن
كلية علوم الحاسب والمعلومات

Princess Nourah bint
Abdulrahman University
College of Computer and
Information Sciences

IT 311 T Information Technology
Term Project Report

Project Topic: creating a new encryption algorithm

Section: 5CA

No.	Student Name	Student Number
1	[redacted]	[redacted]
2		
3	مجد محمد العمري	443007585

First Semester 2023

Contents

1. Introduction

Using Java programming, we have implemented a bit-oriented encryption algorithm known as the R-R CIPHER. The algorithm is designed to encrypt and decrypt our names by rotating the binary plaintext using a key given by the user, followed by reverse. The objective is to enhance the algorithm's security and strengthen authentication, reducing potential attacks.

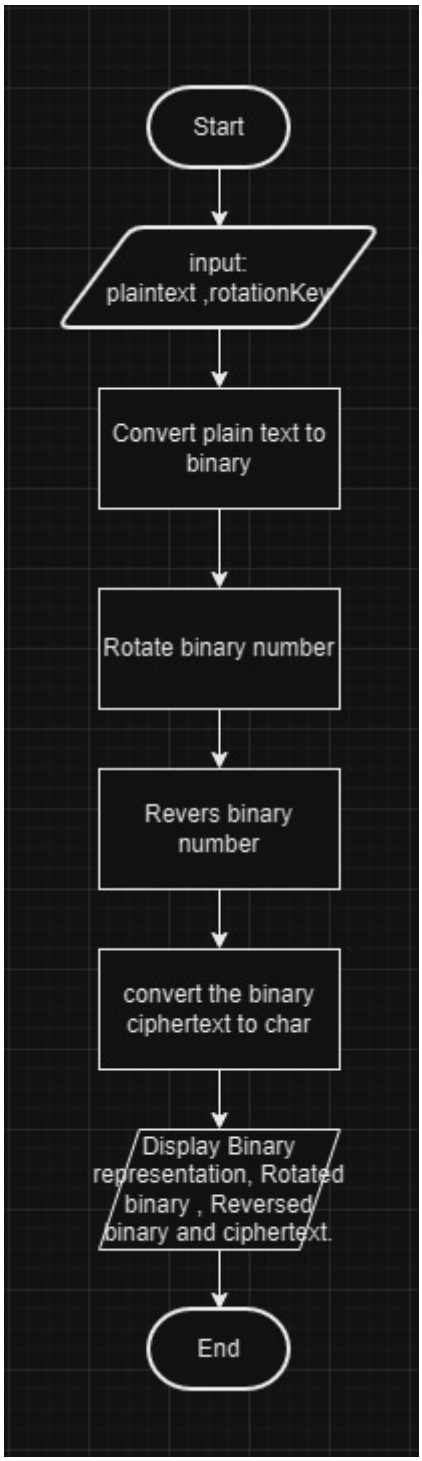
Algorithm explanation:

The algorithm rotates plaintext bits using a user-provided key for a specified number of rotations and then reverses the result:-

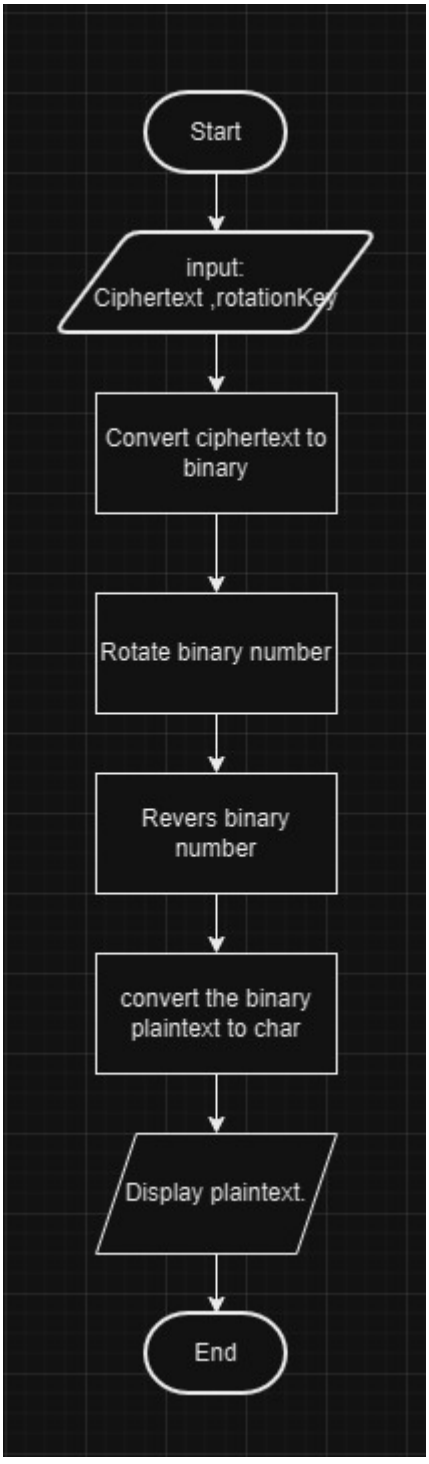
a. Input: the plaintext and the rotation key. rotation key determines the number of shifts during rotation.
b. Rotation: The algorithm performs a bitwise rotation on the plaintext bits according to the value of the rotation key.
c. Reversal: After the rotation, the algorithm reverses the rotated bits, switching 1s to 0s and 0s to 1s.
d. Output: The result of this process, after rotation and reversal, is the encrypted data.

2. Flowchart

Encryption:



Decryption:



3. Implementation

Plaintext: Reema Albogami

```
run:
Enter plain text: Reema Albogami
Binary representation: 0101001001100101011001010110110101100001001000000100000101101100011000100110111101100111011000010110110101101001
Enter rotation amount: 4
Rotated binary: 0010011001010110010101101101011000010010000001000001011011000110001001101111011001110110000101101101011010010101
Reversed binary: 1101100110101001101010010010100111101101111110111110100100111001110110010000100110001001111010010010100101101010
Decoded plain text: Ù©©)íûé9Ù  □é)j
Original plain text: Reema Albogami
BUILD SUCCESSFUL (total time: 16 seconds)
```

Plaintext: Reema Abdullah

```
run:
Enter plain text: Reema Abdullah
Binary representation: 0101001001100101011001010110110101100001001000000100000101100010011001000111010101101100011011000110000101101000
Enter rotation amount: 5
Rotated binary: 0100110010101100101011011010110000100100000010000010110001001100100011101010110110001101100011000010110100001010
Reversed binary: 1011001101010011010100100101001111011011111101111101001110110011011100010101001001110010011100111101001011110101
Decoded plain text: ³SRSÛ÷Ó³qRrsÒõ
Original plain text: Reema Abdullah
BUILD SUCCESSFUL (total time: 13 seconds)
```

Plaintext: Mjd Alamri

```
run:
Enter plain text: Majed Alamri
Binary representation: 0100110101100001011010100110010101100100001000000100000101101100011000010110110101110010011010001
Enter rotation amount: 2
Rotated binary: 0011010110000101101010011001010110010000100000010000010110110001100001011011010111001001101001 01
Reversed binary: 1100101001111010010101100110101001101111011111101111101001001110011111010010010100011011001011010
Decoded plain text: ÊzVjo~úNzJ6Z
Original plain text: Majed Alamri
BUILD SUCCESSFUL (total time: 20 seconds)
```

4. References

5. Appendix

```
import java.util.Scanner;


public class BitRotation {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);


      // Step 1: Get plain text from the user
      System.out.print("Enter plain text: ");
      String plainText = scanner.nextLine();


      // Step 2: Convert plain text to binary
      String binaryText = convertToBinary(plainText);
      System.out.println("Binary representation: " + binaryText);


      // Step 3: Get the rotation amount from the user
      System.out.print("Enter rotation amount: ");
      int rotationAmount = scanner.nextInt();


      // Step 4: Rotate the bits from left to right
      String rotatedBinaryText = rotateBits(binaryText, rotationAmount);
      System.out.println("Rotated binary: " + rotatedBinaryText);


      // Step 5: Reverse each bit
      String reversedBinaryText = reverseBits(rotatedBinaryText);
      System.out.println("Reversed binary: " + reversedBinaryText);
```

```
        // Step 6: Decode the binary and return it to plain text
        String decodedPlainText = decodeBinary(reversedBinaryText);
        System.out.println("Decoded plain text: " + decodedPlainText);

        // Display original plain text
        System.out.println("Original plain text: " + plainText);
    }

    private static String convertToBinary(String plainText) {
        StringBuilder binaryText = new StringBuilder();
        for (char c : plainText.toCharArray()) {
            String binary = Integer.toBinaryString(c);
            binaryText.append(String.format("%8s", binary).replace(' ', '0'));
        }
        return binaryText.toString();
    }

    private static String rotateBits(String binaryText, int rotationAmount) {
        int length = binaryText.length();
        rotationAmount = rotationAmount % length;
        return binaryText.substring(rotationAmount) + binaryText.substring(0,
rotationAmount);
    }

    private static String reverseBits(String binaryText) {
        StringBuilder reversedBinaryText = new StringBuilder();
        for (char c : binaryText.toCharArray()) {
```

```
            if (c == '0') {
                reversedBinaryText.append('1');
            } else {
                reversedBinaryText.append('0');
            }
        }
        return reversedBinaryText.toString();
    }

    private static String decodeBinary(String binaryText) {
        StringBuilder plainText = new StringBuilder();
        for (int i = 0; i < binaryText.length(); i += 8) {
            String binaryByte = binaryText.substring(i, i + 8);
            int decimal = Integer.parseInt(binaryByte, 2);
            plainText.append((char) decimal);
        }
        return plainText.toString();
    }
}
```