One Time Pad Algorithm

import java.io.\*;

public class GFG {

    // function which returns encryptedText

    public static String stringEncryption(String text,

                                          String key)

    {

        // initializing cipherText

        String cipherText = "";

        // initialize cipher array of key length

        // which stores the sum of corresponding no.'s

        // of plainText and key.

        int cipher[] = new int[key.length()];

        for (int i = 0; i < key.length(); i++)

        {

            cipher[i] = text.charAt(i) - 'A' + key.charAt(i)

                        - 'A';

        }

        // if the sum is greater than 25

        // subtract 26 from it and store that resulting

        // value

        for (int i = 0; i < key.length(); i++)

        {

            if (cipher[i] > 25)

            {

                cipher[i] = cipher[i] - 26;

            }

        }

        // convert the no.'s into integers

        // convert these integers to corresponding

        // characters and add them up to cipherText

        for (int i = 0; i < key.length(); i++)

        {

            int x = cipher[i] + 'A';

            cipherText += (char)x;

        }

        // returning the cipherText

        return cipherText;

    }

    // function which returns plainText

    public static String stringDecryption(String s,String key)

    {

        // initializing plainText

        String plainText = "";

        // initializing integer array of key length

        // which stores difference of corresponding no.'s of

        // each character of cipherText and key

        int plain[] = new int[key.length()];

        // running for loop for each character

        // subtracting and storing in the array

        for (int i = 0; i < key.length(); i++)

        {

            plain[i]= s.charAt(i) - 'A' - (key.charAt(i) - 'A');

        }

        // if the difference is less than 0

        // add 26 and store it in the array.

        for (int i = 0; i < key.length(); i++)

        {

            if (plain[i] < 0)

            {

                plain[i] = plain[i] + 26;

            }

        }

        // convert int to corresponding char

        // add them up to plainText

        for (int i = 0; i < key.length(); i++)

        {

            int x = plain[i] + 'A';

            plainText += (char)x;

        }

        // returning plainText

        return plainText;

    }

    // main function

    public static void main(String[] args)

    {

        // declaration of plain text

        String plainText = "Hello";

        // declaration of key

        String key = "MONEY";

        // converting plain text to toUpperCase

        // function call to stringEncryption

        // with plainText and key as parameters

        String encryptedText =

               stringEncryption(plainText.toUpperCase(), key.toUpperCase());

        // printing cipher Text

        System.out.println("Cipher Text - "+ encryptedText);

        // function call to stringDecryption

        // with encryptedText and key as parameters

        System.out.println("Message - "

                 + stringDecryption(encryptedText,

                                    key.toUpperCase()));

    }

}