

Submitted by: Sushree Sangita Sarangi,
Section :- 2241026
Regd no :- 2241019309

Section :- 2241026

Regd no :- 2241019309

Source Code

```
import java.util.Scanner;
public class AffineChiper{
    public static void main(String[]Args){
        Scanner sc= new Scanner(System.in);
        System.out.println("Enter a paraphrase: ");
        String plaintext=sc.nextLine();

        String ciphertext = encrypt(plaintext);
        String decrypted = decrypt(ciphertext);
        System.out.println("Ciphertext: " + ciphertext);
        System.out.println("Decrypted text: " + decrypted);

    }
    public static String encrypt(String plaintext) {
        int k1 = 7;
        int k2 = 2;
        char[] ciphertext = new char[plaintext.length()];

        for (int i = 0; i < plaintext.length(); i++) {
            char character_check = plaintext.charAt(i);
            if (Character.isUpperCase(character_check)){
                int k3 = plaintext.charAt(i) - 'A';
                int encrypted = (k3 * k1 +k2) % 26;
                ciphertext[i] = (char)(encrypted + 'A');
            }
        }
        else {
            int k3 = plaintext.charAt(i) - 'a';
            int encrypted = (k3 * k1 +k2) % 26;
            ciphertext[i] = (char)(encrypted + 'a');
        }
    }
}
```

Source Code (contd..)

```
return new String(ciphertext);
}
public static String decrypt(String ciphertext) {
    int k1 = 7;
    int k2 = 2;
    char[] plaintext = new char[ciphertext.length()];

    int k1_inverse = 0;
    for (int i = 0; i < 26; i++) {
        if ((k1 * i) % 26 == 1) {
            k1_inverse = i;
            break;
        }
    }

    for (int i = 0; i < ciphertext.length(); i++) {
        char character_check = ciphertext.charAt(i);
        if (Character.isUpperCase(character_check)){
            int k3 = ciphertext.charAt(i) - 'A';
            int decrypted = ((k3 - k2 + 26)* k1_inverse) % 26;
            plaintext[i] = (char)(decrypted + 'A');

        } else {
            int k3 = ciphertext.charAt(i) - 'a';
            int decrypted = ((k3 - k2 + 26)* k1_inverse) % 26;
            plaintext[i] = (char)(decrypted + 'a');
        }
    }
    return new String(plaintext);
}
```

EXECUTION

When the following program is executed the text given by the user is first encrypted then decrypted as per the following:

Enter a paraphrase:

goodmorning

Ciphertext: swwxwrpgps

Decrypted text: goodmorning

Enter a paraphrase:

computer

Ciphertext: qwidmfer

Decrypted text: computer

ABOUT THE PROJECT

The code is a Java program that implements the Affine Cipher, a symmetric encryption algorithm. The program uses the Scanner class to input a message from the user to encrypt, as well as the multiplicative and additive keys.

The `getMultiplicativeInverse` function calculates the multiplicative inverse of the key using a loop and returns it when the product of the key and the inverse is 1 (mod 26). The `encrypt` function takes in a character, the multiplicative key (k_1), and the additive key (k_2), and returns the encrypted character. The calculation is performed by first subtracting the ASCII value of 'a' from the character to get its integer value in the range [0, 25], then multiplying it by k_1 , adding k_2 , and taking the result modulo 26. Finally, the encrypted character is obtained by adding the ASCII value of 'a' to the result.

The `decrypt` function takes in a character and the keys and returns the decrypted character. The calculation involves first subtracting the ASCII value of 'a' from the character, getting the multiplicative inverse of k_1 using the `getMultiplicativeInverse` function, then multiplying this inverse by the result of subtracting k_2 from the character value and adding 26 to handle negative values, and taking the result modulo 26. The decrypted character is then obtained by adding the ASCII value of 'a' to the result.

In the main method, the message and keys are inputted using the Scanner class and stored in variables. A loop is then used to encrypt the message by calling the `encrypt` function for each character, and the encrypted message is stored in the `encrypted` variable. A similar loop is used to decrypt the encrypted message by calling the `decrypt` function for each character, and the decrypted message is stored in the `decrypted` variable. The encrypted and decrypted messages are then printed to the console.