Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Experiment 3

Date of Performance : 27-02-2023 **Date of Submission: 27-02-2023**

SAP Id: 60004200107 **Name : Kartik Jolapara**

Div: B **Batch : B1**

Aim of Experiment

Implement simple columnar transposition technique. The columnar transposition rearranges the plaintext letters, based on a matrix filled with letters in the order determined by the secret keyword.

Theory / Algorithm / Conceptual Description

The Columnar Transposition Cipher is a form of transposition cipher just like the Rail Fence Cipher. Columnar Transposition involves writing the plaintext out in rows and then reading the ciphertext off in columns one by one.

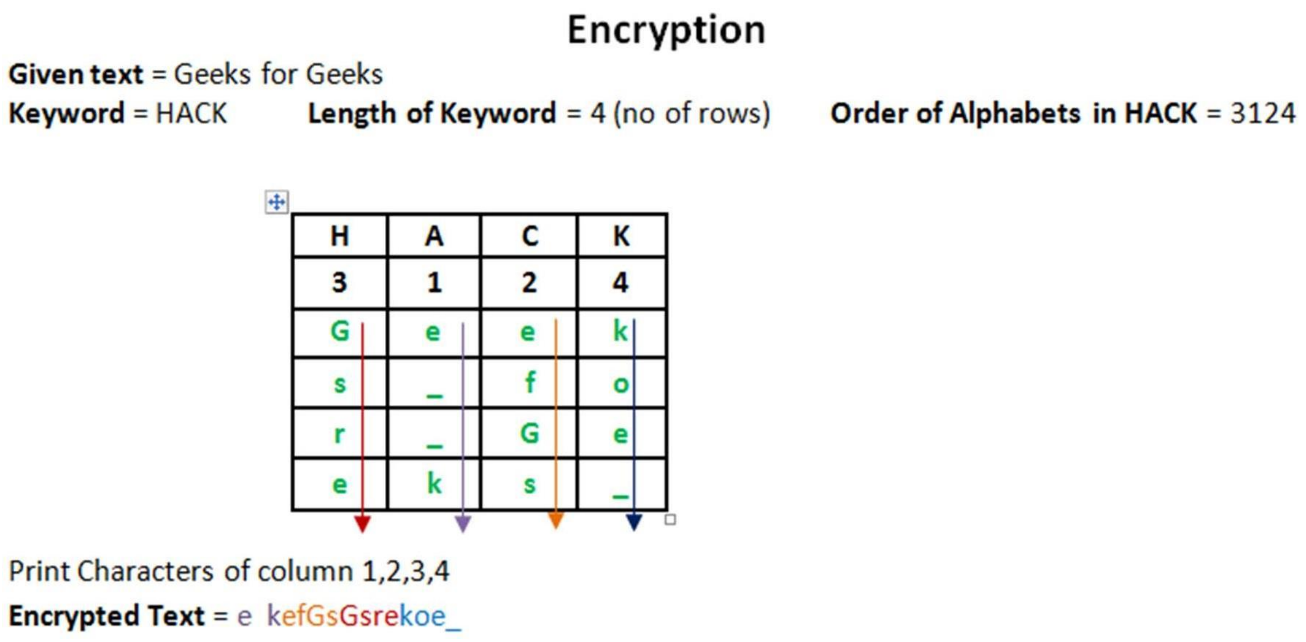
Encryption:

In a transposition cipher, the order of the alphabet is re-arranged to obtain the cipher text:

1. The message is written out in rows of a fixed length, and then read out again column by column, and the columns are chosen in some scrambled order.
2. Width of the rows and the permutation of the columns are usually defined by a keyword.
3. For example, the word HACK is of length 4 (so the rows are of length 4), and the permutation is defined by the alphabetical order of the letters in the keyword. In this case, the order would be "3 1 2 4".
4. Any spare spaces are filled with nulls or left blank or placed by a character (Example: _).
5. Finally, the message is read off in columns, in the order specified by the keyword.

Decryption:

1. To decipher it, the recipient has to work out the column lengths by dividing the message length by the key length.
2. Then, write the message out in columns again, then re-order the columns by reforming the keyword.



Encryption

Given text = Geeks for Geeks

Keyword = HACK **Length of Keyword** = 4 (no of rows) **Order of Alphabets in HACK** = 3124

H	A	C	K
3	1	2	4
G	e	e	k
s	_	f	o
r	_	G	e
e	k	s	_

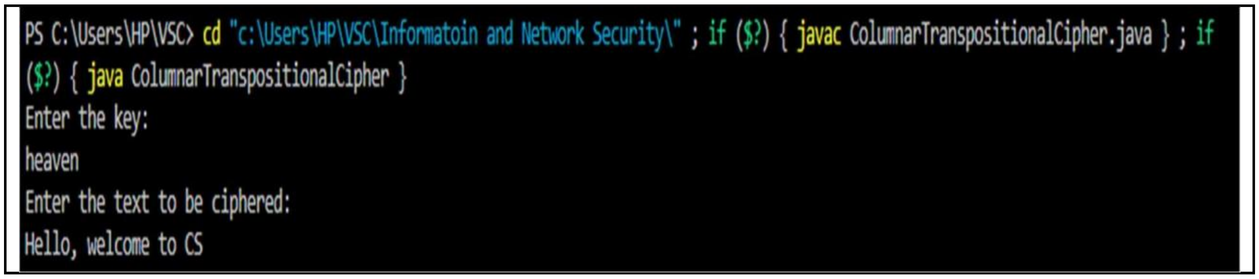
Print Characters of column 1,2,3,4

Encrypted Text = e kefGsGsrekoe_

Program

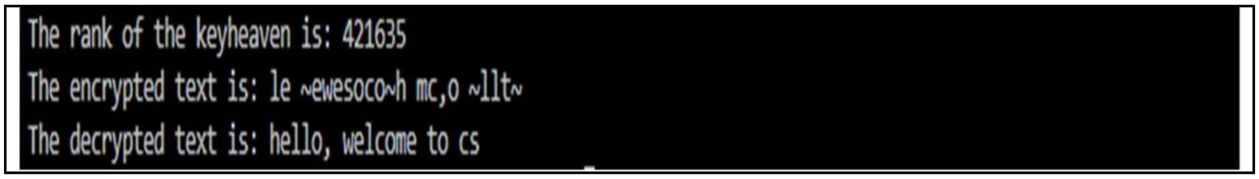
```
import java.util.*;

class ColumnCipher {
    String rank(String key) {
        StringBuilder sb = new StringBuilder();
        char[] ch = key.toLowerCase().toCharArray();
        Arrays.sort(ch);

        for(int i = 0; i < key.length(); i ++) {
            for(int j = 0; j < key.length(); j ++) {
                if(key.toLowerCase().charAt(i) == ch[j]) {
                    ch[j] = '~';
                    sb.append(j + 1);
                    break;
                }
            }
        }

        return sb.toString();
    }

    ArrayList<ArrayList<Character>> encryptMatrix(String text, int n) {
        ArrayList<ArrayList<Character>> arrayList = new ArrayList<>();
        ArrayList<Character> array;
        int row = (int) Math.ceil((double) text.length() / n);
        for(int i = 0; i < row; i ++) {
            array = new ArrayList<>();
            for(int j = 0; j < n; j ++) {
                if(i  *  n  +  j  <  text.length())  {
                    array.add(text.charAt(i * n + j));
                }
            }
            arrayList.add(array);
        }

        while(arrayList.get(arrayList.size() - 1).size() < n) {
            arrayList.get(arrayList.size() - 1).add('~');
        }

        return arrayList;
    }

    String encrypt(String rank, String text) {
        Map<Integer, Integer> map = new HashMap<>();
        ArrayList<ArrayList<Character>> arrayList = encryptMatrix(text.toLowerCase(),
rank.length());
        StringBuilder sb = new StringBuilder();
        for(int i = 0; i < rank.length(); i ++) {
            map.put(Integer.parseInt(String.valueOf(rank.charAt(i))), i);
        }

        for(int i = 0; i < rank.length(); i ++) {
            for(int j = 0; j < (int) Math.ceil((double) text.length() / rank.length()); j ++) {
                sb.append(arrayList.get(j).get(map.get(i + 1)));
            }
        }

        return sb.toString();
    }

    char[][] decryptMatrix(String cipher, Map<Integer, Integer> map, int n) {
        int row = (int) Math.ceil((double) cipher.length() / n);
        char[][] ch = new char[row][n];

        for(int i = 0; i < n; i ++) {
            for(int j = 0; j < row; j ++) {
                ch[j][map.get(i + 1)] = cipher.charAt(i * row + j);
            }
        }

        return ch;
    }

    String decrypt(String rank, String cipher) {
        Map<Integer, Integer> map = new HashMap<>();
        StringBuilder sb = new StringBuilder();
        for(int i = 0; i < rank.length(); i ++) {
            map.put(Integer.parseInt(String.valueOf(rank.charAt(i))), i);
        }

        char[][] ch = decryptMatrix(cipher.toLowerCase(), map, rank.length());

        for (char[] chars : ch) {
            for (char c : chars) {
                sb.append(c);
            }
        }

        return sb.toString().replace("~", "");
    }
}

public class ColumnarTranspositionalCipher {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        ColumnCipher columnCipher = new ColumnCipher();
        System.out.println("Enter the key: ");
        String key = in.nextLine();
        System.out.println("Enter the text to be ciphered: ");
        String text = in.nextLine();
        String rank = columnCipher.rank(key.toLowerCase());
        String encryptedText = columnCipher.encrypt(rank, text);
        String decryptedText = columnCipher.decrypt(rank, encryptedText);

        System.out.println("The rank of the key" + key + " is: " + rank);
        System.out.println("The encrypted text is: " + encryptedText);
        System.out.println("The decrypted text is: " + decryptedText);
    }
}
```

Input



```
PS C:\Users\HP\VSC> cd "c:\Users\HP\VSC\Informatoin and Network Security\" ; if ($?) { javac ColumnarTranspositionalCipher.java } ; if
($?) { java ColumnarTranspositionalCipher }
Enter the key:
heaven
Enter the text to be ciphered:
Hello, welcome to CS
```

Output



```
The rank of the keyheaven is: 421635
The encrypted text is: le ~ewesoco~h mc,o ~llt~
The decrypted text is: hello, welcome to cs
```