Experiment-3

Aim: Study and implement program for Transposition(Columnar) cipher to encrypt and decrypt the message.

Introduction:

- ➔The Columnar Transposition Cipher is a type of transposition cipher. In a transposition cipher, the order of the letters in the message is changed, but the letters themselves are not replaced.
- ➔The Columnar Transposition Cipher works by dividing the message into a number of rows, equal to the length of the key. The message is then written out in columns, starting with the first column, then the second column, and so on. The columns are then read in a predetermined order, which is defined by the key.
- ➔For example, let's say we have the message "hello world" and the key "secret". The message would be divided into 4 rows, since the key is 4 characters long. The message would then be written out in columns, starting with the first column, then the second column, and so on. This would give us the following:

  row 1: h l o
  row 2: w o r
  row 3: d

➔The columns would then be read in the order of the key, which is "secret". This would give us the encrypted message "swolderohw".

➔To decrypt the message, we would simply reverse the process. We would first read the columns in the order of the key. Then, we would write the letters out in rows, starting with the first row, then the second row, and so on.

➔The Columnar Transposition Cipher is a relatively simple cipher, but it is still effective. It is also relatively easy to implement, which makes it a good choice for beginners.

➔Advantages:

- It is relatively simple to understand and implement.
- It is relatively secure, as long as the key is kept secret.
- It can be used to encrypt messages of any length.

➔Disadvantages:

- It is vulnerable to frequency analysis, if the key is not long enough.
- It can be slow to encrypt and decrypt messages.

➔Overall, the Columnar Transposition Cipher is a simple and effective cipher that can be used to encrypt messages of any length. It is a good choice for beginners, but it is important to remember that it is not unbreakable.

Program(Source Code):

```
import math

def encryption (massage, key):
    massage = massage.replace(" ", "")
    msg = []
    x = 0
    count = 0
    num_of_rows = math.ceil((len(massage)/len(key)))
    padding = int(num_of_rows * len(key)) - len(massage)
    massage += padding * "x"

    for rows in range(num_of_rows):
        row = []
        for i in range(len(key)):
            row.append(massage[i+x])
            count += 1
        x = count
        msg.append(row)
    # print(msg)

    key = key.upper()
    key_list = [ord(i) for i in key]
    # print(key_list)
    sorted_key_list = sorted(key_list)
    # print(sorted_key_list)
    # print(key_list.index(65))
    # print(key_list.index(66))
    col_index = [key_list.index(i)  for i in sorted_key_list]
    # print(col_index)

    e_msg = []
    for i in col_index:
        for j in range(len(msg)):
            e_msg.append(msg[j][i])
    return "".join(e_msg)

def decryption(e_msg, key):
    rows = len(e_msg) // len(key)
    # print(rows)

    key = key.upper()
    key_list = [ord(i) for i in key]
    sorted_key_list = sorted(key_list)
```

```
    col_index = [key_list.index(i) for i in sorted_key_list]
    # print(col_index)

    sequence = [char for char in key]
    count = 1
    for i in col_index:
        sequence[sequence.index(chr(key_list[i]))] = count
        count += 1
    # print(sequence)

    temp = []
    for i in sequence:
        for j in range(((i-1) * rows), (i * rows)):
            temp.append(e_msg[j])

    d_msg = []
    for i in range(rows):
        count = 0
        for j in range(len(key)):
            if temp[count+i] == "x":
                d_msg.append("")
            else:
                d_msg.append(temp[count+i])
                count += rows
    return "".join(d_msg)

def main():
    massage = input("Enter message to encrypt: ")
    key = input("Enter key: ")
    print()

    e_msg = encryption (massage, key)
    print("Encrypted message: ", e_msg)
    print()

    d_msg = decryption (e_msg, key)
    print("Decrypted message: ", d_msg)
    print()

if __name__ =="__main__":
    main()
```

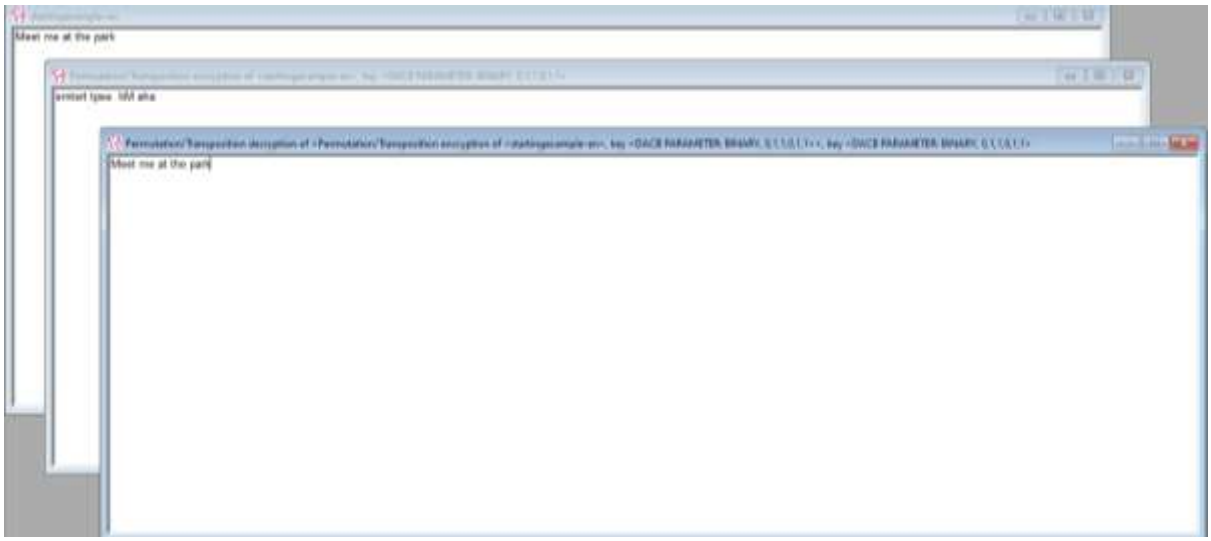
Output(Program):

```
Enter message to encrypt: hello
Enter key: 21

Encrypted message:  elxhlo

Decrypted message:  hello
```

Output(Cryptool):



[Encryption and Decryption]

Cryptanalysis:

Cryptanalysis is the study of ciphers, codes, and encrypted text in order to understand how they work and find ways to decrypt them without knowing the secret key. Cryptanalysts use a variety of techniques to break ciphers, including:

- Frequency analysis: This technique relies on the fact that certain letters are more common in English than others. For example, the letter "e" is the most common letter in English, followed by "t" and "a". By analyzing the

frequency of letters in the encrypted message, it is possible to deduce the key and decrypt the message.

- Keyword guessing: If the attacker knows the general length of the keyword, they can try to guess the keyword by hand. This can be done by looking for common words or phrases that are the same length as the keyword.
- Computer-aided cryptanalysis: There are a number of computer programs that can be used to break the Columnar Transposition Cipher. These programs use a variety of techniques, including frequency analysis and keyword guessing, to crack the cipher.

Application:

- Military: The Columnar Transposition Cipher was used by the military during World War I and World War II. It was also used by the American Civil War.
- Espionage: The Columnar Transposition Cipher was used by spies to communicate with each other.
- Business: The Columnar Transposition Cipher was used by businesses to encrypt sensitive information, such as financial data.
- Personal: The Columnar Transposition Cipher can be used by individuals to encrypt personal messages, such as love letters or diary entries.

References:

1. https://www.geeksforgeeks.org/caesar-cipher-in-cryptography/
2. https://bard.google.com/