Suraj Raju Lokapure | 2020BTECS00081 | B5 batch

CNS lab | Experiment 04

Aim: To implement Railfence cipher

Theory: The Rail Fence Cipher is a type of transposition cipher. It works by writing the plaintext in a zigzag pattern across a specified number of lines (rails). The characters are then read off in rows to create the ciphertext. To decrypt, the process is reversed using the same number of rails to reveal the original message.

Code:

```
#include <bits/stdc++.h>
using namespace std;

string format(string &str) {
    for (char c : str) {
        if (isalpha(c)) {
            c+=tolower(c);
        }
    }
    return str;
}

string encrypt(string &plain, int key) {
    vector<vector<char>> matrix(key);

    int rowNumber = 0;
    int flag = 1;
    for (int i = 0; i < plain.size(); i++) {
        matrix[rowNumber].push_back(plain[i]);
        rowNumber += flag;
        if (rowNumber == 0)
            flag = 1;

        if (rowNumber == key - 1)
            flag = -1;
    }

    string cipher;
    for (int i = 0; i < key; i++) {
        for (int j = 0; j < matrix[i].size(); j++)
            cipher += matrix[i][j];
    }
```

```
    return cipher;
}

string decrypt(string &cipher, int key) {
    vector<vector<int>> matrixDecry(key);
    int rowNumber = 0;
    int flag = 1;

    int n = cipher.length();

    for (int i = 0; i < n; i++) {
        matrixDecry[rowNumber].push_back(i);
        rowNumber += flag;
        if (rowNumber == (key - 1))
            flag = -1;
        if (rowNumber == 0)
            flag = 1;
    }

    vector<int> mapping;
    for (int i = 0; i < key; i++) {
        for (int j = 0; j < matrixDecry[i].size(); j++)
            mapping.push_back(matrixDecry[i][j]);
    }

    map<int, char> m;
    for (int i = 0; i < n; i++)
        m[mapping[i]] = cipher[i];

    string plain;
    for (int i = 0; i < n; i++)
        plain += m[i];

    return plain;
}

int main() {
    int choice;
    cout << "1. Encrypt\n2. Decrypt\nEnter your choice: ";
    cin >> choice;
    cin.get();

    if (choice == 1) {
        string plain;
        int key;
        cout << "\nEnter plain text: ";
        getline(cin, plain);
        plain = format(plain);
```

```
        cout << "\nEnter key: integer value: ";
        cin >> key;

        string cipher = encrypt(plain, key);

        cout << "\nEncrypted text is : " << cipher << endl;
    } else if (choice == 2) {
        string cipher;
        int key;
        cout << "\nEnter cipher text: ";
        getline(cin, cipher);
        cipher = format(cipher);

        cout << "\nEnter key: integer value: ";
        cin >> key;

        string plain = decrypt(cipher, key);

        cout << "\nDecrypted text is : " << plain << endl;
    }

    return 0;
}
```

Output:

```
PS C:\Users\shree\Documents\My workspace 2\CNS lab\Exp4> c
nce_cipher } ; if ($?) { .\railfence_cipher }
1. Encrypt
2. Decrypt
Enter your choice: 1

Enter plain text: weaponhidedintheoffice

Enter key: integer value: 4

Encrypted text is : whnfeniitfiaoddhocpeee
PS C:\Users\shree\Documents\My workspace 2\CNS lab\Exp4> c
nce_cipher } ; if ($?) { .\railfence_cipher }
1. Encrypt
2. Decrypt
Enter your choice: 2

Enter cipher text: whnfeniitfiaoddhocpeee

Enter key: integer value: 4

Decrypted text is : weaponhidedintheoffice
PS C:\Users\shree\Documents\My workspace 2\CNS lab\Exp4>
```