AHSANULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY

Department of Computer Science and Engineering

Program: Bachelor of Science in Computer Science and Engineering

Course Code : **CSE 4174**
Course Title : **Cyber Security Lab**
Academic Semester : **Spring 2023**

Assignment Topic : **Substitution & Transposition Ciphers**

Submitted on: **19-11-2023**

Submitted by

Name: **Meherin Sultana**
Student ID: **20200104036**
Lab Section: **A2**

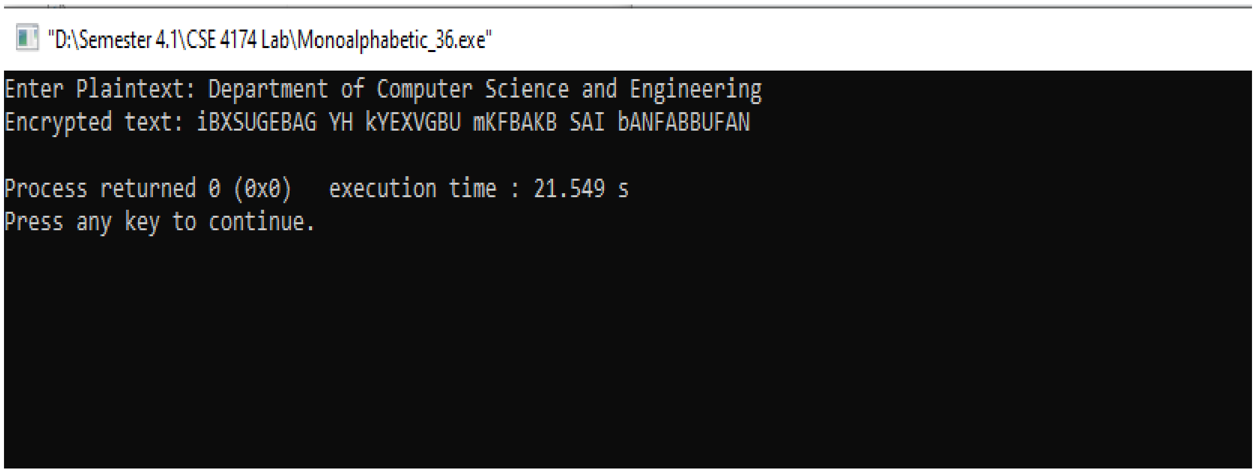
Question 1:

Devise a code for implementation of Monoalphabetic cipher.

Code (implemented in C):

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

#define ALPHABET_SIZE 26

void generateKey(char key[ALPHABET_SIZE]);
void encrypt(const char *plaintext, const char key[ALPHABET_SIZE],
char *ciphertext);

int main()
{
        char key[ALPHABET_SIZE];
        char plaintext[100];
        char ciphertext[100];
        generateKey(key);
        printf("Enter Plaintext: ");

        fgets(plaintext, sizeof(plaintext), stdin);

        if (strlen(plaintext) > 0 && plaintext[strlen(plaintext) - 1] ==
'\n')
        {
        plaintext[strlen(plaintext) - 1] = '\0';
        }
        encrypt(plaintext, key, ciphertext);
        printf("Encrypted text: %s\n", ciphertext);
        return 0;
}

void generateKey(char key[ALPHABET_SIZE])
{
        char alphabet[ALPHABET_SIZE] =
"ABCDEFGHIJKLMNOPQRSTUVWXYZ";
        int i, j, temp;
        for (i = ALPHABET_SIZE - 1; i > 0; i--)
        {
        j = rand() % (i + 1);
        srand( time(0));
        temp = alphabet[i];
        alphabet[i] = alphabet[j];
        alphabet[j] = temp;
        }

        strcpy(key, alphabet);
}

void encrypt(const char* plaintext, const char key[ALPHABET_SIZE],
char* ciphertext)
{
        int i;
        for (i = 0; plaintext[i] != '\0'; i++)
        {
        if (isalpha(plaintext[i]))
        {
        char originalChar = toupper(plaintext[i]);
        int index = originalChar - 'A';
        char encryptedChar = islower(plaintext[i]) ? key[index] :
tolower(key[index]);
        ciphertext[i] = encryptedChar;
        }
        else
        {
        ciphertext[i] = plaintext[i];
        }
        }
        ciphertext[i] = '\0';
}
```

Output:



"D:\Semester 4.1\CSE 4174 Lab\Monoalphabetic_36.exe"

```
Enter Plaintext: Department of Computer Science and Engineering
Encrypted text: iBXSUGEBAG YH kYEXVGBU mKFBAKB SAI bANFABBUFAN

Process returned 0 (0x0)   execution time : 21.549 s
Press any key to continue.
```

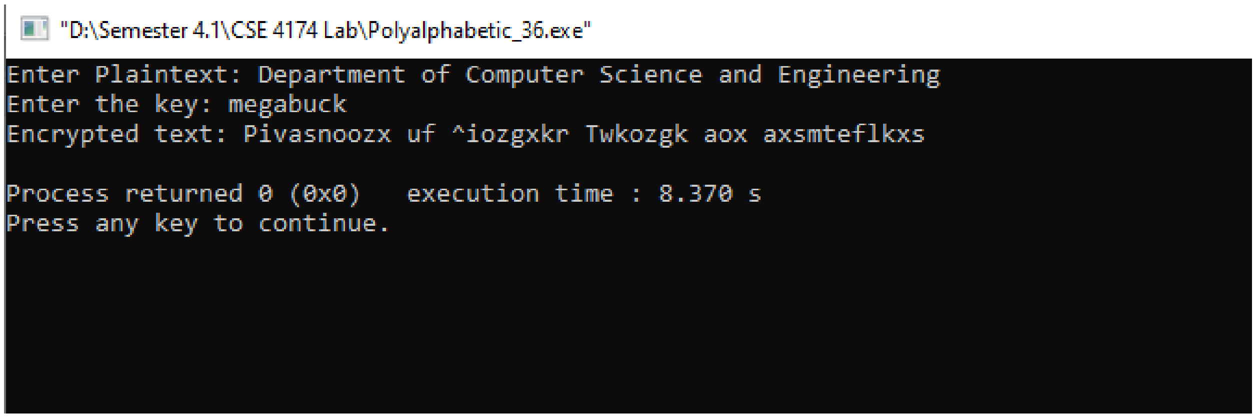
Question 2:

Devise a code for implementation of Polyalphabetic cipher.

Code (implemented in C):

```
#include <stdio.h>
#include <ctype.h>
#include <stdlib.h>
#include <string.h>
void encr(const char* plaintext, const char* key, char* ciphertext);
int main(){
        char key[100];
        char plaintext[100];
        char ciphertext[100];

        printf("Enter Plaintext: ");

        fgets(plaintext, sizeof(plaintext), stdin);
        if (strlen(plaintext)> 0 && plaintext[strlen(plaintext) - 1] == '\n')
        {
        plaintext[strlen(plaintext) - 1] = '\0';
        }
        printf("Enter the key: ");
        fgets(key, sizeof(key), stdin);

        if (strlen(key) > 0 && key[strlen(key) - 1] == '\n')
        {
        key[strlen(key) - 1] = '\0';
        }
        encr(plaintext, key, ciphertext);
        printf("Encrypted text: %s\n", ciphertext);
        return 0;
}
void encr(const char* plaintext, const char* key, char* ciphertext)
{
        int i, j;
        int keyLength = strlen(key);
        for (i = 0, j = 0; plaintext[i] != '\0'; i++)
        {
        if (isalpha(plaintext[i]))
        {
        char shift = isupper(key[j]) ? 'A' : 'a';
        ciphertext[i] = ((plaintext[i] + key[j] - 2 * shift) % 26) + shift;
        j = (j + 1) % keyLength;
        }
        else
        {
        ciphertext[i] = plaintext[i];
        }
        }
        ciphertext[i] = '\0';
}
```

Output:



"D:\Semester 4.1\CSE 4174 Lab\Polyalphabetic_36.exe"

```
Enter Plaintext: Department of Computer Science and Engineering
Enter the key: megabuck
Encrypted text: Pivasnoozx uf ^iozgxkr Twkozgk aox axsmteflkxs

Process returned 0 (0x0)   execution time : 8.370 s
Press any key to continue.
```

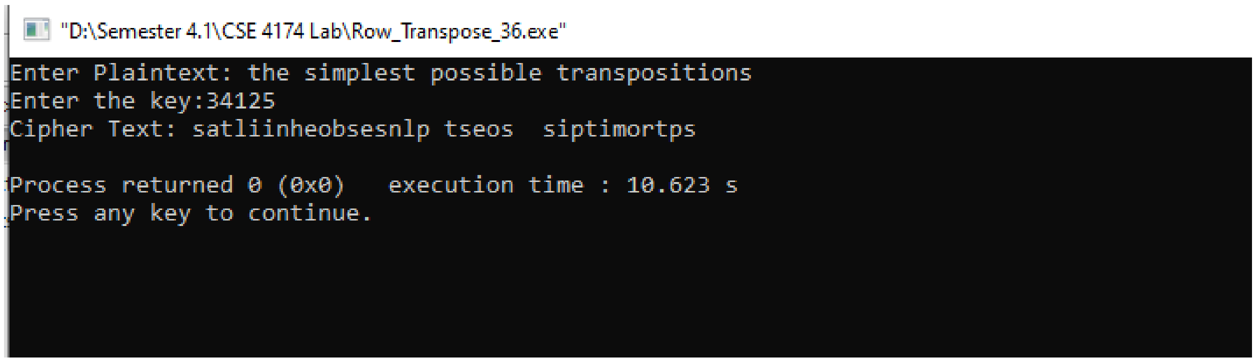
Question 3:

Devise a code for implementation of Row Transposition cipher.

Code (implemented in C):

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void encr(const char* plaintext, const char* key, char* ciphertext);
void createPermutationIndices(const char* key, int* in, int length);

int main()
{
        char key[100];
        char plaintext[100];
        char ciphertext[100];

        printf("Enter Plaintext: ");
        fgets(plaintext, sizeof(plaintext), stdin);

        if (strlen(plaintext)> 0 && plaintext[strlen(plaintext) - 1] == '\n')
        {
        plaintext[strlen(plaintext)- 1] = '\0';
        }
        printf("Enter the key:");
        fgets(key, sizeof(key), stdin);

        if (strlen(key)>0 && key[strlen(key) -1] == '\n')
        {
        key[strlen(key) - 1] = '\0';
        }
        encr(plaintext, key, ciphertext);
        printf("Cipher Text: %s\n", ciphertext);
        return 0;
}

void encr(const char* plaintext, const char* key, char* ciphertext)
{
        int i, row, col;
        int index = 0;
        int textLength = strlen(plaintext);
        int keyLength = strlen(key);
        int numRows = (textLength + keyLength - 1) / keyLength;
        int in[keyLength];
        createPermutationIndices(key, in, keyLength);

        for(i =0; i < textLength; i++)
        {
        ciphertext[i] = 'X';
        }
        ciphertext[textLength] = '\0';

        for(row =0; row < numRows; row++)
        {
        for(col= 0; col < keyLength; col++)
        {
        int position = row + in[col] * numRows;
        if (position < textLength)
        {
                ciphertext[index++] = plaintext[position];
        }
        }
        }
}
void createPermutationIndices(const char* key, int* in, int length)
{
        for (int i = 0; i < length; i++)
        {
        in[i] = i;
        }

        for (int i =0; i < length-1; i++)
        {
        for (int j= i+1; j < length; j++)
        {
        if(key[in[i]] > key[in[j]])
        {
                int te = in[i];
                in[i]= in[j];
                in[j]= te;
        }
        }
        }
}
```

Output:



"D:\Semester 4.1\CSE 4174 Lab\Row_Transpose_36.exe"

```
Enter Plaintext: the simplest possible transpositions
Enter the key:34125
Cipher Text: satliinheobsesnlp tseos  siptimortps

Process returned 0 (0x0)   execution time : 10.623 s
Press any key to continue.
```

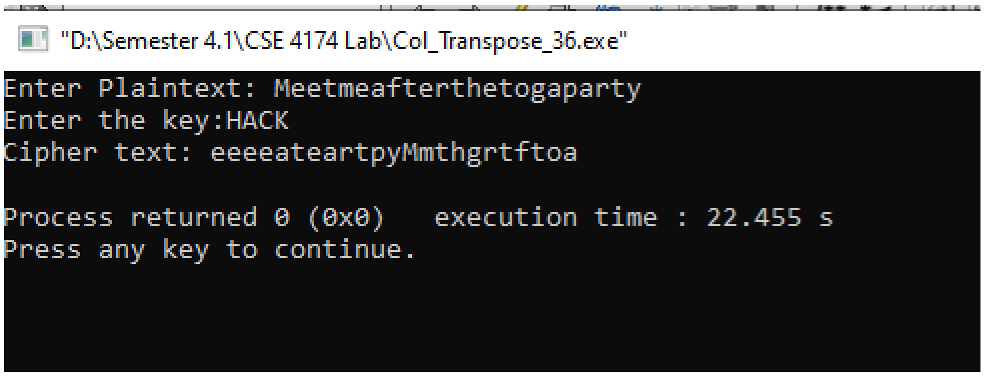
Question 4:

Devise a code for implementation of Column Transposition cipher.

Code (implemented in C):

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void encr(const char* plaintext, const char* key, char* ciphertext);
void createPermutationIndices(const char* key, int* in, int length);
int main()
{
        char key[100];
        char plaintext[70];
        char ciphertext[70];
        printf("Enter Plaintext: ");

        fgets(plaintext, sizeof(plaintext), stdin);

        if (strlen(plaintext) >0 && plaintext[strlen(plaintext) - 1] == '\n')
        {
        plaintext[strlen(plaintext) - 1] = '\0';
        }
        printf("Enter the key:");
        fgets(key, sizeof(key), stdin);

        if (strlen(key)>0 && key[strlen(key)-1] == '\n')
        {
        key[strlen(key)- 1] = '\0';
        }
        encr(plaintext, key, ciphertext);
        printf("Cipher text: %s\n", ciphertext);
        return 0;
}

void encr(const char* plaintext, const char* key, char* ciphertext)
{
        int i, row, col;
        int index = 0;
        int textLength = strlen(plaintext);
        int keyLength = strlen(key);
        int numCols = keyLength;
        int numRows = (textLength + numCols - 1) / numCols;
        int in[keyLength];

        createPermutationIndices(key, in, keyLength);

        for(i = 0; i< textLength; i++)
        {
        ciphertext[i] = 'X';
        }
        ciphertext[textLength] = '\0';

        for(col =0; col < numCols; col++)
        {
        for(row=0; row < numRows; row++)
        {
        int position = row * numCols + in[col];
        if(position< textLength)
        {
                ciphertext[index++] = plaintext[position];
        }
        }
        }
}

void createPermutationIndices(const char* key, int* in, int length)
{
        int j,t;

        for(int i =0; i < length; i++)
        {
        in[i] = i;
        }
        for(int i=0; i < length - 1; i++)
        {
        for(j = i+1; j < length; j++)
        {
        if(key[in[i]] > key[in[j]])
        {
                t= in[i];
                in[i] = in[j];
                in[j] = t;
        }
        }
        }
}
```

Output:



```
"D:\Semester 4.1\CSE 4174 Lab\Col_Transpose_36.exe"

Enter Plaintext: Meetmeafterthetogaparty
Enter the key:HACK
Cipher text: eeeeateartpyMmthgrtftoa

Process returned 0 (0x0)   execution time : 22.455 s
Press any key to continue.
```