**crypto Lab session 1st**

**1. Caesar Cipher**

**Code:**

#include <stdio.h>

#include <string.h>

void caesarCipher(char text[], int key) {

for (int i = 0; text[i] != '\0'; i++) {

char ch = text[i];

if (ch >= 'a' && ch <= 'z') {

text[i] = ((ch - 'a' + key) % 26) + 'a';

} else if (ch >= 'A' && ch <= 'Z') {

text[i] = ((ch - 'A' + key) % 26) + 'A';

}

}

}

int main() {

char text[100];

int key;

printf("Enter text: ");

gets(text);

printf("Enter key (1-25): ");

scanf("%d", &key);

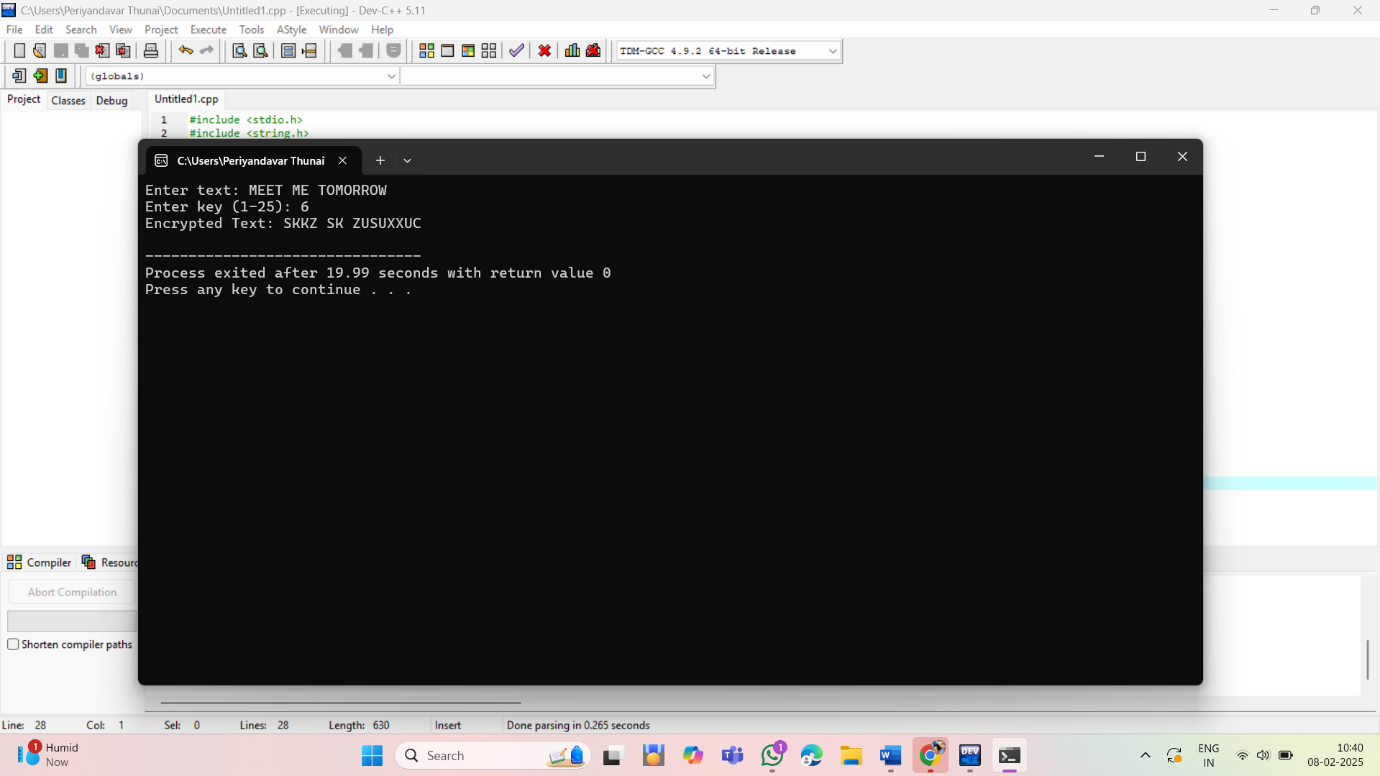
caesarCipher(text, key);

printf("Encrypted Text: %s\n", text);

return 0;

}

**OUTPUT:**

****

**2. Monoalphabetic Substitution Cipher**

**Code:**

#include <stdio.h>

#include <string.h>

void monoalphabeticCipher(char text[], char mapping[]) {

for (int i = 0; text[i] != '\0'; i++) {

if (text[i] >= 'a' && text[i] <= 'z') {

text[i] = mapping[text[i] - 'a'];

} else if (text[i] >= 'A' && text[i] <= 'Z') {

text[i] = mapping[text[i] - 'A'] - 32;

}

}

}

int main() {

char text[100];

char mapping[] = "QWERTYUIOPLKJHGFDSAZXCVBNM"; // Example mapping

printf("Enter text: ");

gets(text);

monoalphabeticCipher(text, mapping);

printf("Encrypted Text: %s\n", text);

return 0;

}

**OUTPUT:**

**A computer screen shot of a black screen

AI-generated content may be incorrect.**

**3. Playfair Cipher**

**Code:**

#include <stdio.h>

#include <string.h>

#define SIZE 5

char matrix[SIZE][SIZE] = {

{'M', 'F', 'H', 'I', 'J'},

{'U', 'N', 'O', 'P', 'Q'},

{'Z', 'V', 'W', 'X', 'Y'},

{'E', 'L', 'A', 'R', 'G'},

{'D', 'S', 'T', 'B', 'C'}

};

void findPosition(char ch, int \*row, int \*col) {

if (ch == 'J') ch = 'I';

for (int i = 0; i < SIZE; i++) {

for (int j = 0; j < SIZE; j++) {

if (matrix[i][j] == ch) {

\*row = i;

\*col = j;

return;

}

}

}

}

void playfairEncrypt(char text[]) {

int i = 0;

while (text[i] != '\0') {

char a = text[i], b = text[i + 1];

int row1, col1, row2, col2;

findPosition(a, &row1, &col1);

findPosition(b, &row2, &col2);

if (row1 == row2) {

text[i] = matrix[row1][(col1 + 1) % SIZE];

text[i + 1] = matrix[row2][(col2 + 1) % SIZE];

} else if (col1 == col2) {

text[i] = matrix[(row1 + 1) % SIZE][col1];

text[i + 1] = matrix[(row2 + 1) % SIZE][col2];

} else {

text[i] = matrix[row1][col2];

text[i + 1] = matrix[row2][col1];

}

i += 2;

}

}

int main() {

char text[] = "MUSTSEEYOUOVERCADOGANWESTCOMINGATONCE";

playfairEncrypt(text);

printf("Encrypted Text: %s\n", text);

return 0;

}

**OUTPUT:**

**A computer screen with a black screen

AI-generated content may be incorrect.**

**4. Frequency Analysis Decryption Helper**

**Code:**

#include <stdio.h>

#include <string.h>

void frequencyAnalysis(char text[]) {

int freq[26] = {0};

for (int i = 0; text[i] != '\0'; i++) {

if (text[i] >= 'a' && text[i] <= 'z') {

freq[text[i] - 'a']++;

} else if (text[i] >= 'A' && text[i] <= 'Z') {

freq[text[i] - 'A']++;

}

}

printf("Letter Frequency Analysis:\n");

for (int i = 0; i < 26; i++) {

printf("%c: %d\n", i + 'A', freq[i]);

}

}

int main() {

char text[200];

printf("Enter encrypted text: ");

gets(text);

frequencyAnalysis(text);

return 0;

}

**OUTPUT:**

**A computer screen with a black screen

AI-generated content may be incorrect.**

**5. Keyword-based Monoalphabetic Cipher**

**Code**:

#include <stdio.h>

#include <string.h>

void generateCipher(char key[], char cipher[]) {

int used[26] = {0}, index = 0;

for (int i = 0; key[i] != '\0'; i++) {

if (!used[key[i] - 'A']) {

cipher[index++] = key[i];

used[key[i] - 'A'] = 1;

}

}

for (char ch = 'A'; ch <= 'Z'; ch++) {

if (!used[ch - 'A']) {

cipher[index++] = ch;

}

}

}

void monoalphabeticEncrypt(char text[], char cipher[]) {

for (int i = 0; text[i] != '\0'; i++) {

if (text[i] >= 'A' && text[i] <= 'Z') {

text[i] = cipher[text[i] - 'A'];

} else if (text[i] >= 'a' && text[i] <= 'z') {

text[i] = cipher[text[i] - 'a'] + 32;

}

}

}

int main() {

char text[100], key[] = "CIPHER", cipher[26];

generateCipher(key, cipher);

printf("Enter text: ");

gets(text);

monoalphabeticEncrypt(text, cipher);

printf("Encrypted Text: %s\n", text);

return 0;

}

**OUTPUT:**

**A computer screen with a black screen

AI-generated content may be incorrect.**

**6. Hill Cipher**

**Code:**

#include <stdio.h>

#include <string.h>

#define MOD 26

void hillCipher(int key[2][2], char text[]) {

for (int i = 0; i < strlen(text); i += 2) {

int x = text[i] - 'A';

int y = text[i + 1] - 'A';

int newX = (key[0][0] \* x + key[0][1] \* y) % MOD;

int newY = (key[1][0] \* x + key[1][1] \* y) % MOD;

text[i] = newX + 'A';

text[i + 1] = newY + 'A';

}

}

int main() {

int key[2][2] = {{9, 4}, {5, 7}};

char text[] = "MEETMEATTHEUSUALPLACEATTENRATHERTHANEIGHTOCLOCK";

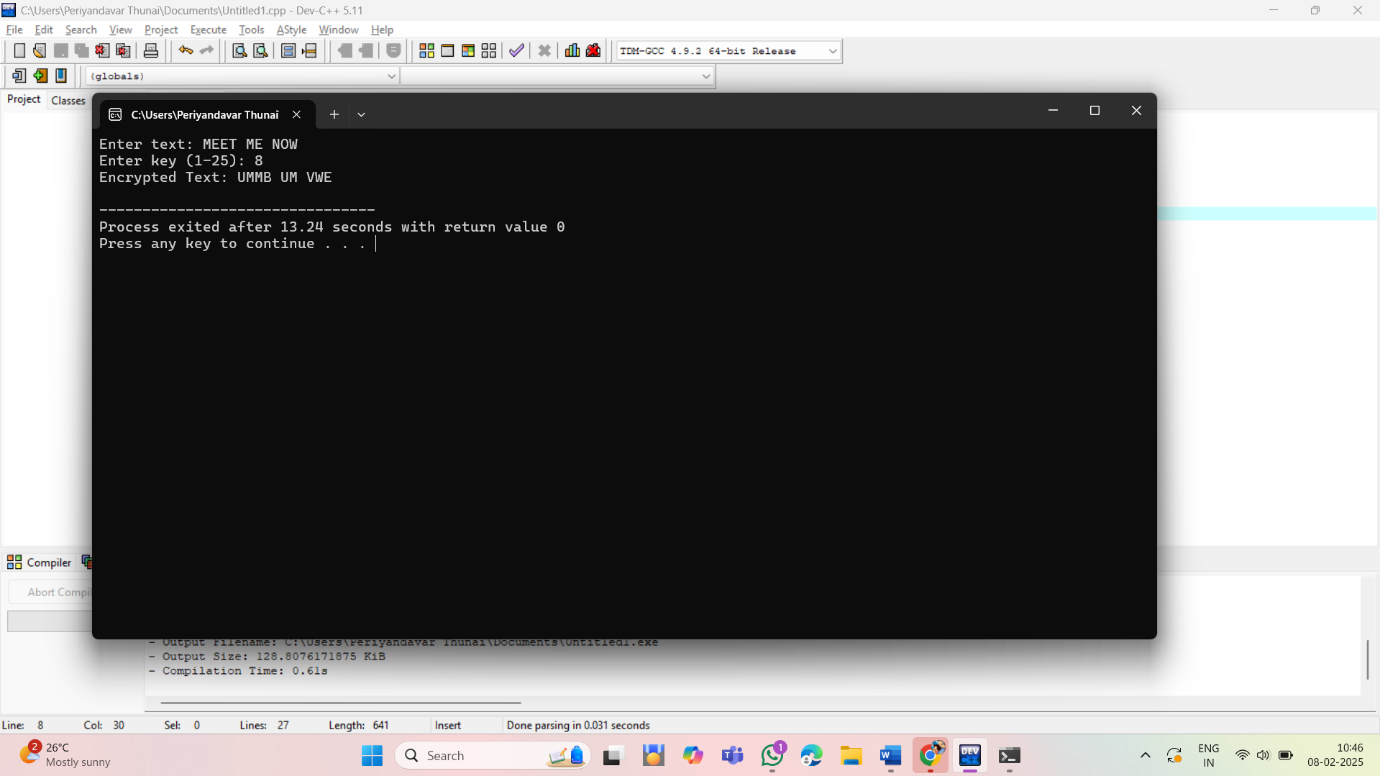
hillCipher(key, text);

printf("Encrypted Text: %s\n", text);

return 0;

}

**OUTPUT:**

****