EXPERIMENT-4

AIM: To implement polyalphabetic cipher encryption-decryption.

THEORY:

The **Polyalphabetic Cipher** is a type of substitution cipher that uses multiple substitution alphabets to encrypt the plaintext, making it more secure than simple monoalphabetic ciphers like the Caesar cipher. In this method, each letter in the plaintext is substituted with a letter from one of several alphabets, determined by a repeating keyword. The most common example is the **Vigenère Cipher**, where each letter of the key defines a shift value for encrypting the corresponding letter of the plaintext. This repetition of multiple substitution alphabets reduces the chances of frequency analysis attacks, as the same plaintext letter can be represented by different ciphertext letters depending on its position and the key used. Decryption is performed by reversing the shifting process using the same key.

CODE:

```
// To implement Polyalphabetic cipher encryption decryption.
Encryption/Decryption: Based on substitution, using multiple
substitution Alphabets
// By Aaditya Bhatia 23/CS/004

#include <iostream>
#include <string>
#include <cctype>

using namespace std;

string toUpper(string s) {
for (char &c : s) {
```

```
c = toupper(c);
}
return s;
}

string encrypt(string plain_text, string key) {
   string cipher_text = "";
   key = toUpper(key);
int key_len = key.length();
int key_idx = 0;

   for (char c : plain_text) {
       if (isalpha(c)) {
           char base = isupper(c) ? 'A' : 'a';
           cipher_text += char(((c - base + (key[key_idx % key_len] -
'A')) % 26) + base);
           key_idx++;
       } else {
           cipher_text += c;
       }
   }
   return cipher_text;
}

string decrypt(string cipher_text, string key) {
   string plain_text = "";
   key = toUpper(key);
int key_len = key.length();
int key_idx = 0;

   for (char c : cipher_text) {
       if (isalpha(c)) {
           char base = isupper(c) ? 'A' : 'a';
           plain_text += char(((c - base - (key[key_idx % key_len] -
'A') + 26) % 26) + base);
```

```
            key_idx++;
        } else {
            plain_text += c;
        }
    }
    return plain_text;
}

int main() {
    string plain_text, key;
    cout << "Enter Text :";
    cin >> plain_text;
    cout << "Enter Key :";
    cin >> key;
    string cipher_text;
    cipher_text = encrypt(plain_text, key);
    cout << "Cipher Text :" << cipher_text << endl;
    cout << "Decryption :" << decrypt(cipher_text, key) << endl;
    return 0;
}
```

OUTPUT:

```
gcc           gcc-13        gcc-ar        gcc-ar-13     gcc-nm        gc
(AI) aadi@Joshua:~/Projects/LABS2025/INS Lab$ g++ 4.cpp -o 4 && .4
.4: command not found
(AI) aadi@Joshua:~/Projects/LABS2025/INS Lab$ g++ 4.cpp -o 4 && ./4
Enter Text :HITHISISAADI
Enter Key :FGASDT
Cipher Text :MOTZLLNYASGB
Decryption :HITHISISAADI
(AI) aadi@Joshua:~/Projects/LABS2025/INS Lab$
```

LEARNING OUTCOME:

To understand and implement polyalphabetic cipher encryption and decryption using multiple substitution alphabets for enhanced data security.