#include <iostream>

#include<string.h>

using namespace std;

// Caesar Cipher Function (Encrypt/Decrypt)

string caesarCipher(string text, int shift, bool decrypt = false)

{

if (decrypt)

{

shift = -shift;

}

for (int i = 0; i < text.length(); i++)

{

char ch = text[i];

if (isupper(ch))

{

ch = ((ch - 'A' + shift + 26) % 26) + 'A';

}

else if (islower(ch))

{

ch = ((ch - 'a' + shift + 26) % 26) + 'a';

}

text[i] = ch;

}

return text;

}

// Build QWERTY Mapping for encryption and decryption

void buildQwertyMapping(char map[], char reverseMap[]) {

string normal = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 ";

string qwerty = "qwertyuiopasdfghjklzxcvbnmQWERTYUIOPASDFGHJKLZXCVBNM1234567890 ";

for (int i = 0; i < normal.length(); i++) {

map[(int)normal[i]] = qwerty[i];

reverseMap[(int)qwerty[i]] = normal[i];

}

}

// Encrypt using QWERTY map

string qwertyEncrypt(string text, char map[]) {

for (int i = 0; i < text.length(); i++) {

if (map[(int)text[i]]) {

text[i] = map[(int)text[i]];

}

}

return text;

}

// Decrypt using reverse QWERTY map

string qwertyDecrypt(string text, char reverseMap[]) {

for (int i = 0; i < text.length(); i++) {

if (reverseMap[(int)text[i]]) {

text[i] = reverseMap[(int)text[i]];

}

}

return text;

}

int main() {

string text;

cout << "Enter text to encrypt: ";

getline(cin,text);

cout << "1 - Caesar Cipher\n";

cout << "2 - Modified Caesar Cipher\n";

cout << "3 - QWERTY Cipher\n";

cout << "4 - Exit\n";

int choice, shift;

cin >> choice;

switch (choice) {

case 1:

cout << "Caesar Cipher:\n";

text = caesarCipher(text, 3); // Using Caesar Cipher with shift 3

cout << "Encrypted: " << text << endl;

cout << "Decrypted: " << caesarCipher(text, 3, true) << endl;

break;

case 2:

cout << "Enter shift value: ";

cin >> shift;

cout << "Modified Caesar Cipher:\n";

text = caesarCipher(text, shift); // Using Caesar Cipher with user-defined shift

cout << "Encrypted: " << text << endl;

cout << "Decrypted: " << caesarCipher(text, shift, true) << endl;

break;

case 3: {

char map[256] = {}, reverseMap[256] = {};

buildQwertyMapping(map, reverseMap);

cout << "QWERTY Cipher:\n";

text = qwertyEncrypt(text, map); // Encrypt using QWERTY mapping

cout << "Encrypted: " << text << endl;

cout << "Decrypted: " << qwertyDecrypt(text, reverseMap) << endl;

break;

}

case 4:

cout << "Exiting...\n";

break;

default:

cout << "Invalid choice!\n";

}

return 0;

}