**Name: Dhanraj Subhash Kore**

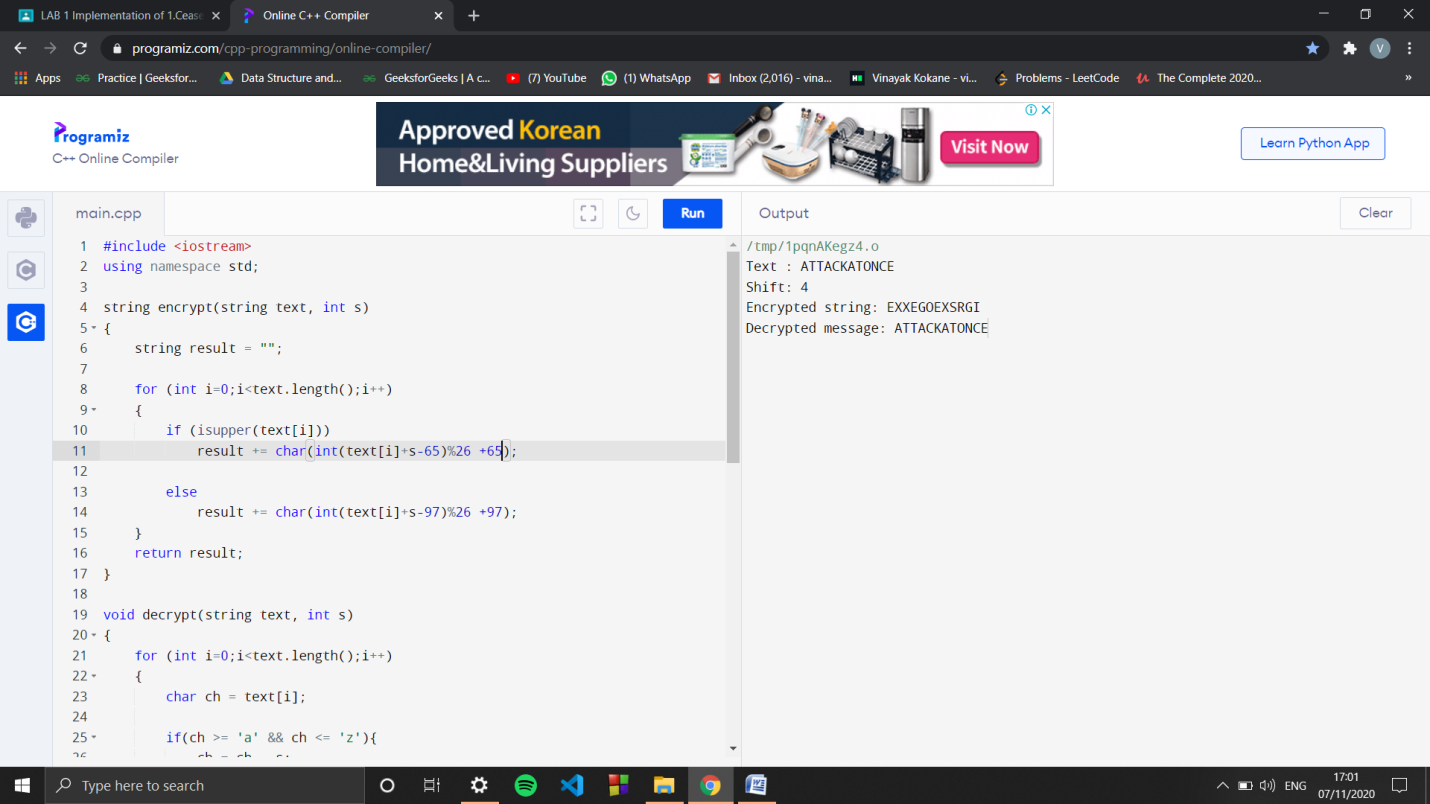
**Div : B Roll No: 60**

**Batch: B-3**

**Sub : CS LAB Assignment**

**LAB 1 Implementation of 1.Ceaser ciphers and 2.Vigenere Cipher**

**1) Ceasor Cipher:**



**Code:**

#include <iostream>

using namespace std;

string encrypt(string text, int s)

{

string result = "";

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

{

if (isupper(text[i]))

result += char(int(text[i]+s-65)%26 +65);

else

result += char(int(text[i]+s-97)%26 +97);

}

return result;

}

void decrypt(string text, int s)

{

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

{

char ch = text[i];

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

ch = ch - s;

if(ch < 'a'){

ch = ch + 'z' - 'a' + 1;

}

text[i] = ch;

}

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

ch = ch - s;

if(ch < 'A'){

ch = ch + 'Z' - 'A' + 1;

}

text[i] = ch;

}

}

cout<<"\nDecrypted message: "<< text;

}

int main()

{

string text="ATTACKATONCE";

int s = 4;

cout << "Text : " << text;

cout << "\nShift: " << s;

string enc = encrypt(text, s);

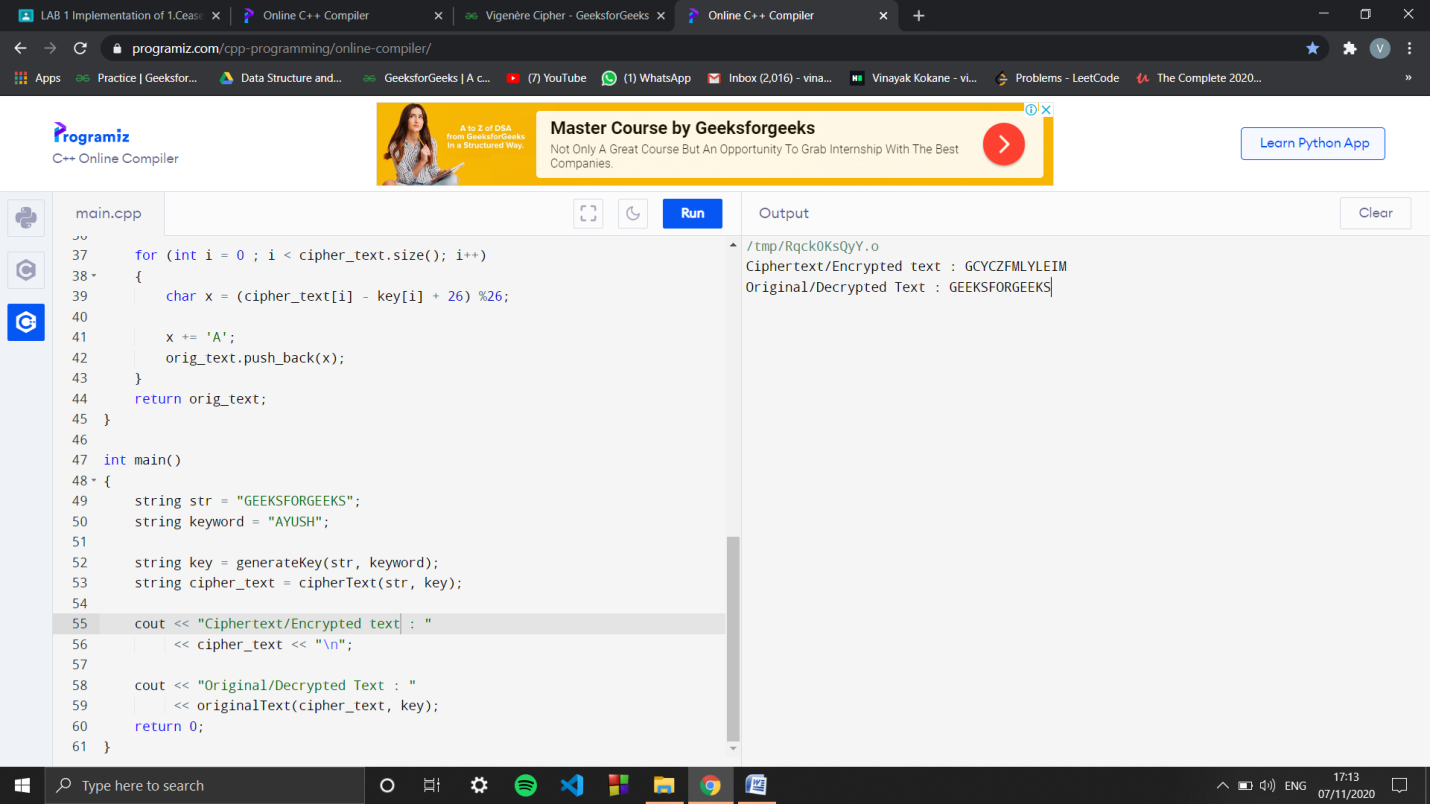
cout << "\nEncrypted string: " << enc;

decrypt(enc,s);

return 0;

}

**1) Vigenere Cipher:**



**Code:**

#include<bits/stdc++.h>

using namespace std;

string generateKey(string str, string key)

{

int x = str.size();

for (int i = 0; ; i++)

{

if (x == i)

i = 0;

if (key.size() == str.size())

break;

key.push\_back(key[i]);

}

return key;

}

string cipherText(string str, string key)

{

string cipher\_text;

for (int i = 0; i < str.size(); i++)

{

char x = (str[i] + key[i]) %26;

x += 'A';

cipher\_text.push\_back(x);

}

return cipher\_text;

}

string originalText(string cipher\_text, string key)

{

string orig\_text;

for (int i = 0 ; i < cipher\_text.size(); i++)

{

char x = (cipher\_text[i] - key[i] + 26) %26;

x += 'A';

orig\_text.push\_back(x);

}

return orig\_text;

}

int main()

{

string str = "GEEKSFORGEEKS";

string keyword = "AYUSH";

string key = generateKey(str, keyword);

string cipher\_text = cipherText(str, key);

cout << "Ciphertext/Encrypted text : "

<< cipher\_text << "\n";

cout << "Original/Decrypted Text : "

<< originalText(cipher\_text, key);

return 0;

}