**Чернівецький національний університет імені Юрія Федьковича**

**Інститут фізико – технічних та комп’ютерних наук**

Лабораторна робота № 5

на тему

**“** **Алгоритми злиття та пошуку”**

Виконала : студентка

групи № 244(А)

Мельничук А.Г.

Затверджено : доцентом ЧНУ

Угрин Д.І.

Чернівці 2022

**Хід роботи**

**Тема :** алгоритми злиття та пошуку.

**Мета роботи :** навчитись реалізовувати різні алгоритми пошуку.

**Завдання до роботи :** скласти схеми алгоритмів та написати програми реалізації алгоритмів Рабіна-Карпа та КМП. Провести порівняльний аналіз алгоритмів пошуку в рядках.

*Код програми :*

#include <stdio.h>

#include <string.h>

#include <math.h>

#include <iostream>

#include <time.h>

#include <vector>

using namespace std;

#define tonum(c) (c >= 'A' && c <= 'Z' ? c - 'A' : c - 'a' + 26)

int mod(int a, int p, int m)

{

if (p == 0)

return 1;

int sqr = mod(a, p / 2, m) % m;

if (p & 1)

return ((a % m) \* sqr) % m;

else

return sqr;

}

int RabinKarpMatch(char\* T, char\* P, int d, int q)

{

int i, j, p, t, n, m, h, found;

n = strlen(T);

m = strlen(P);

h = mod(d, m - 1, q);

p = t = 0;

for (i = 0; i < m; i++)

{

p = (d \* p + tonum(P[i])) % q;

t = (d \* t + tonum(T[i])) % q;

}

for (i = 0; i <= n; i++)

{

if (p == t)

{

found = 1;

for (j = 0; j < m; j++)

if (P[j] != T[i + j])

{

found = 0;

break;

}

if (found)

return i + 1;

}

else

{

t = (d \* (t - ((tonum(T[i]) \* h) % q)) + tonum(T[i + m])) % q;

}

}

return -1;

}

vector<int> KMP(string S, string K)

{

vector<int> T(K.size() +1, -1);

vector<int> matches;

if (K.size() == 0)

{

matches.push\_back(0);

return matches;

}

for (int i = 1; i <= K.size(); i++)

{

int pos = T[i - 1];

while (pos != -1 && K[pos] != K[i - 1]) pos = T[pos];

T[i] = pos + 1;

}

int sp = 0;

int kp = 0;

cout << "Значення префіксної функції: ";

while (sp < S.size())

{

while (kp != -1 && (kp == K.size() || K[kp] != S[sp])) kp = T[kp];

kp++;

sp++;

if (kp == K.size()) matches.push\_back(sp - K.size());

cout << sp - kp << ' ';

}

cout << endl;

return matches;

}

int main(int argc, char\* argv[])

{

setlocale(LC\_ALL, "Ukrainian");

//Алгоритм Рабіна-Карпа...........

long t11 = clock();

int spiv;

int d = 1, q = 1000;

char T[200];

char P[100];

cout << "Алгоритм Рабіна-Карпа: " << endl;

cout << "\n============================= " << endl;

cout << "Введіть рядок T: " << endl; //T = nmnsfwfkwjf

cin >> T;

cout << "Введіть рядок, який хочете знайти (підрядок): " << endl; //edece

cin >> P;

spiv = RabinKarpMatch(T, P, d, q);

if (spiv)

cout << " Співпадіння в " << spiv << " позиції" << endl;

else

cout << "Співпадінь не знайдено!!!";

long t22 = clock();

cout << "Час виконання: " << (t22 - t11) / (double(CLOCKS\_PER\_SEC)) << endl;

cout << "\n============================= " << endl;

//Алгоритм Кнута-Морриса Пратта (КМП)...........

long t1 = clock();

string A, B;

cout << "Введіть рядок T: " << endl;

cin >> A;

cout << "Введіть рядок, який хочете знайти (підрядок): " << endl;

cin >> B;

cout << endl;

vector<int> returnValue = KMP(A, B);

cout << endl;

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

{

cout << " Співпадіння в " << returnValue[i] << " позиції" << endl;

cout << endl;

}

long t2 = clock();

cout << "Час виконання: " << (t2 - t1) / (double(CLOCKS\_PER\_SEC)) << endl;

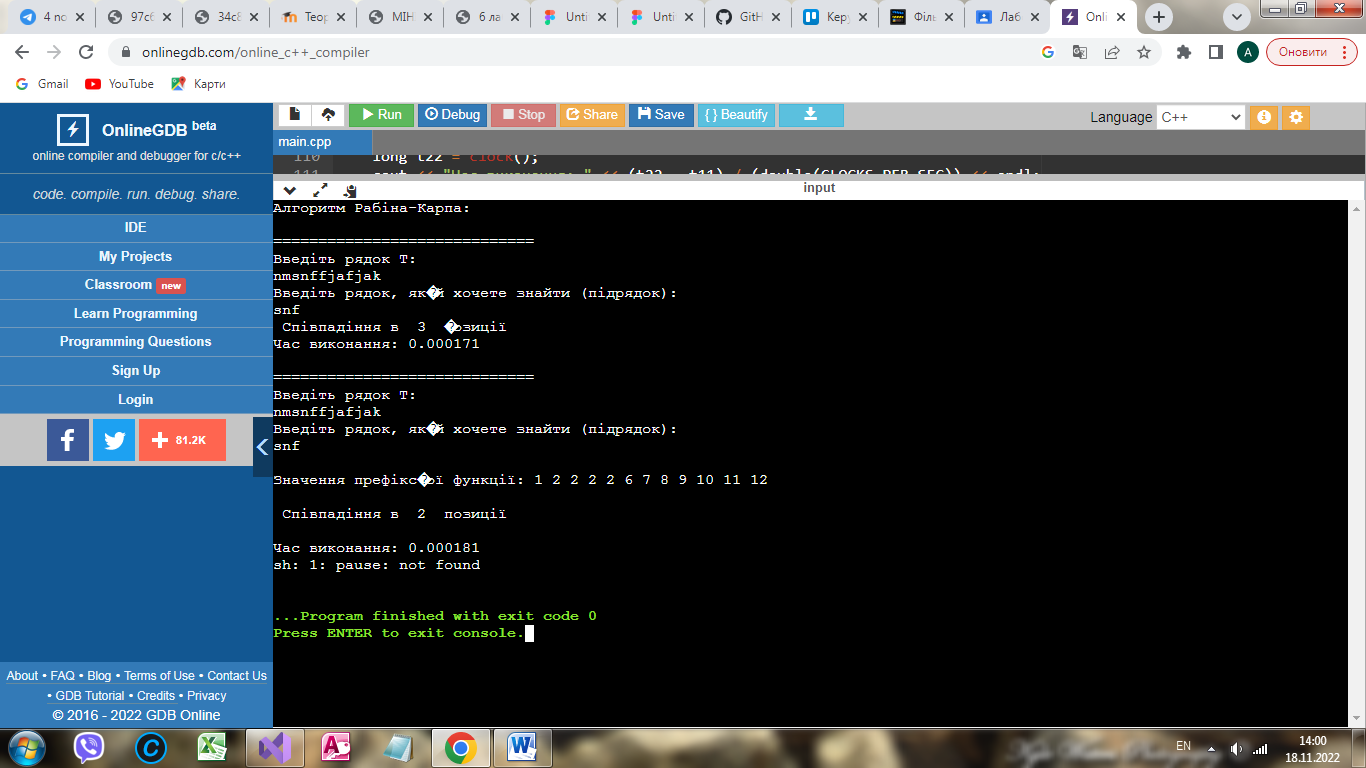
system("pause");

return 0;

}

Мій опис виконаного завдання : як і було вказано в завданні, потрібно було реалізувати два алгоритми : Рабіна-Карпа та КМП , також порівняти їх між собою, і дізнатись який з цих методів більш ефективний. Отже дана програма описує реалізацію цих двох методів, та виводить час виконання пошуку. Спочатку потрібно ввести повний рядок , а потім рядок який потрібно знайти. Програма здійснить пошук та виведе час виконання пошуку. По цьому , можна побачити , який з даних методів більш ефективний. Далі на скріншоті ми зможемо розглянути результат виконання програми

*Скрін виводу консолі :*



*Висновок :* під час виконання даної лабораторної роботи ми розглянули два алгоритми пошуку : Рабіна-Карпа та КМП. Дізнались як за допомогою цих методів можна знайти знайти потрібний елемент серед цілого рядка значень. Також зробили порівняння і як виявилось алгоритм пошуку Рабіна-Карпа ефективніший .