Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

ПНИПУ

**Лабораторная работа  
“Поиски”**

Выполнил:   
студент группы РИС-23-1б   
Агзамов Артур Альферович

Проверила:   
доцент кафедры ИТАС   
О.А. Полякова

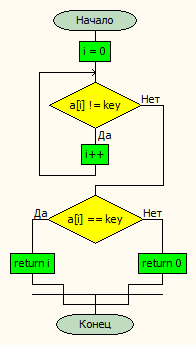
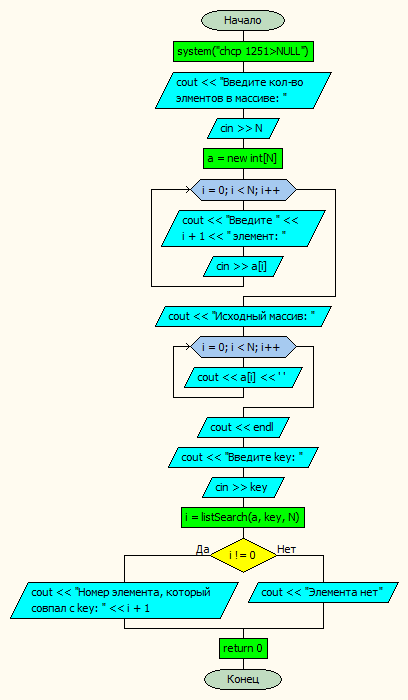
Пермь, 2024 г.

**Анализ задачи:**

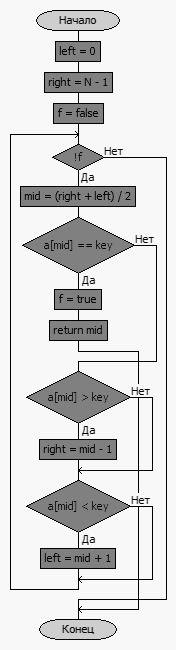
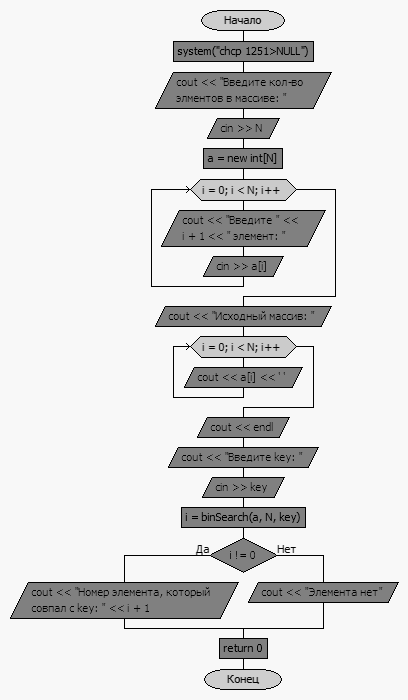
Изучить и применить определённый способ поиска.

**Блок-Схема:**

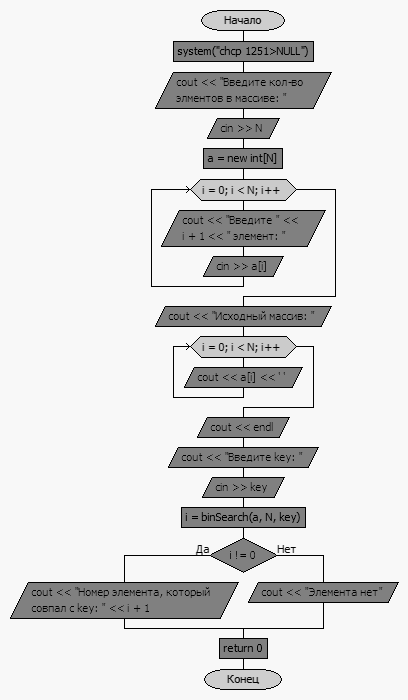
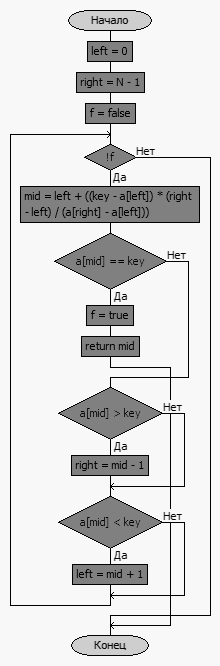
**1) Линейный поиск:**



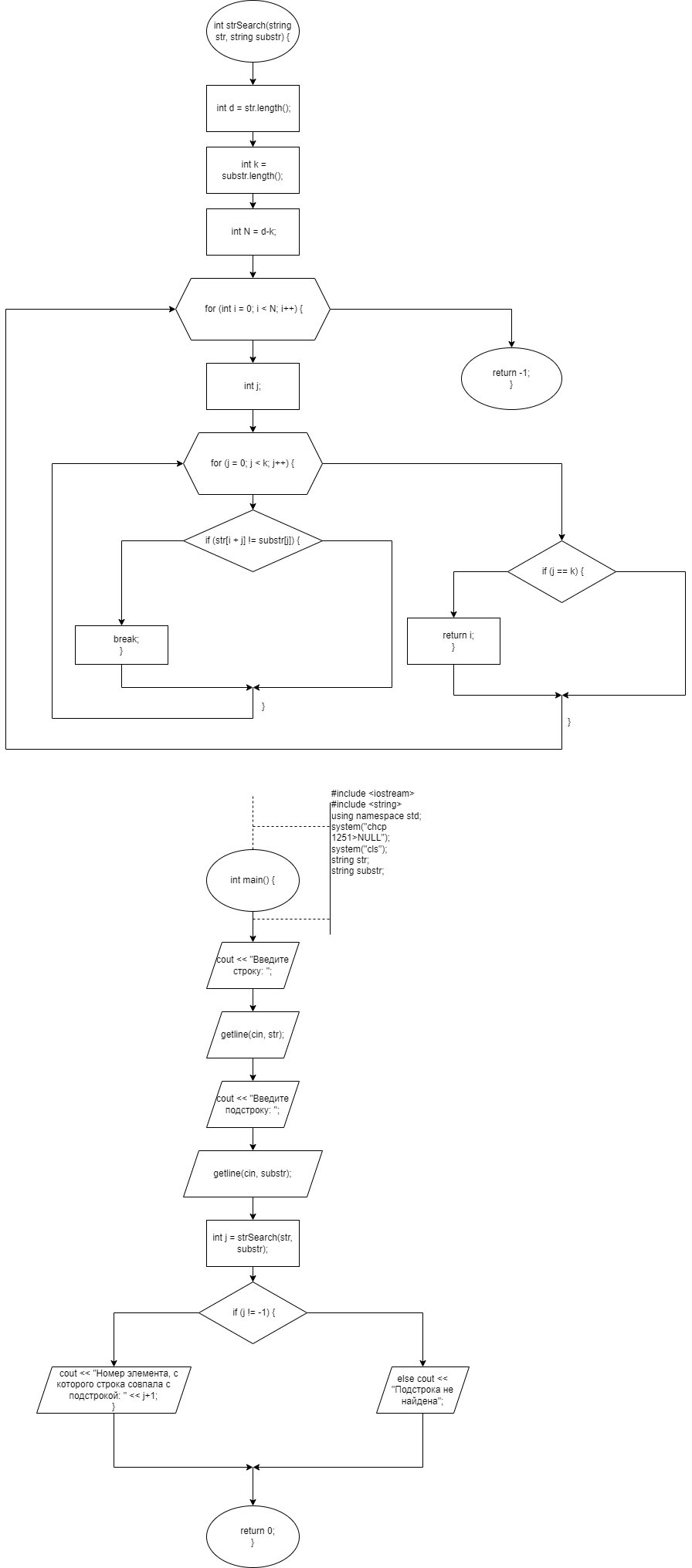
**2) Бинарный поиск:**

****

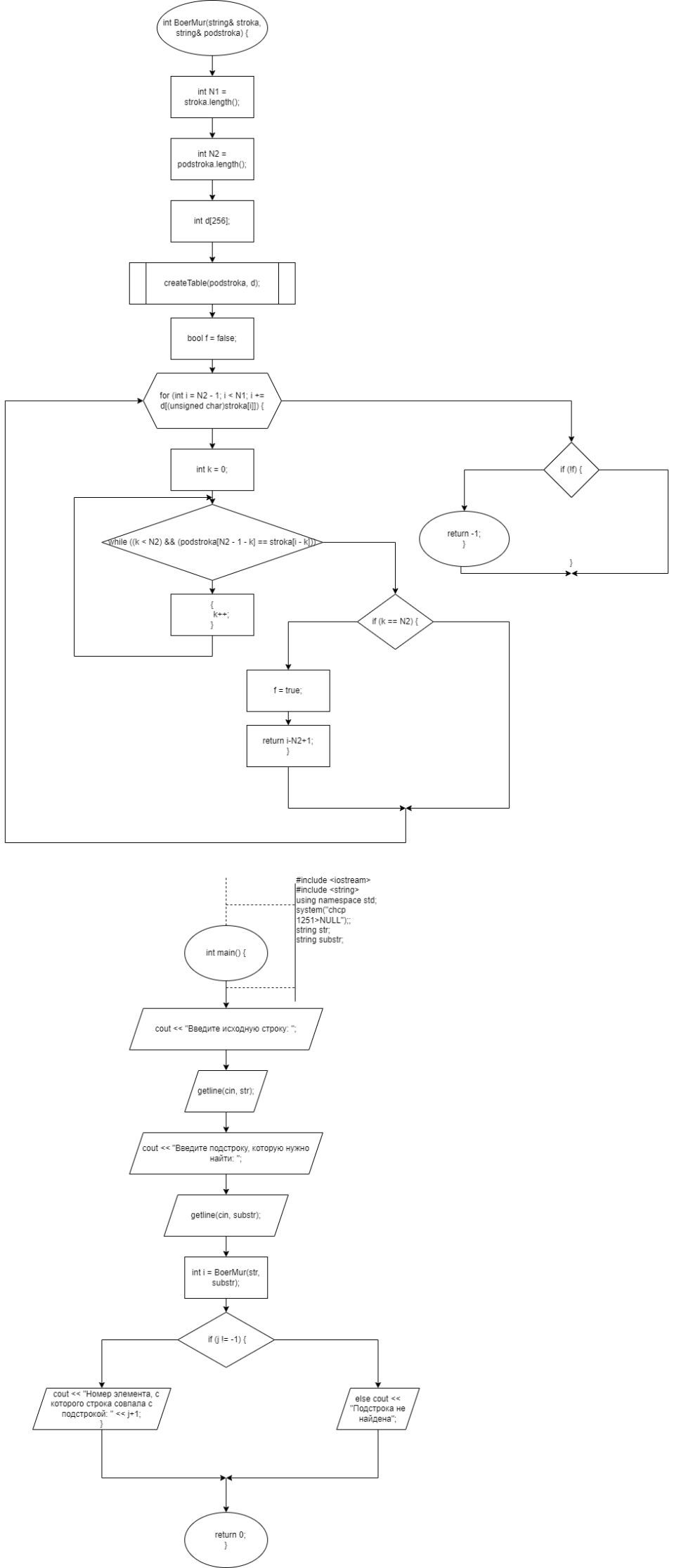
**3)Интерполяционный поиск:**

****

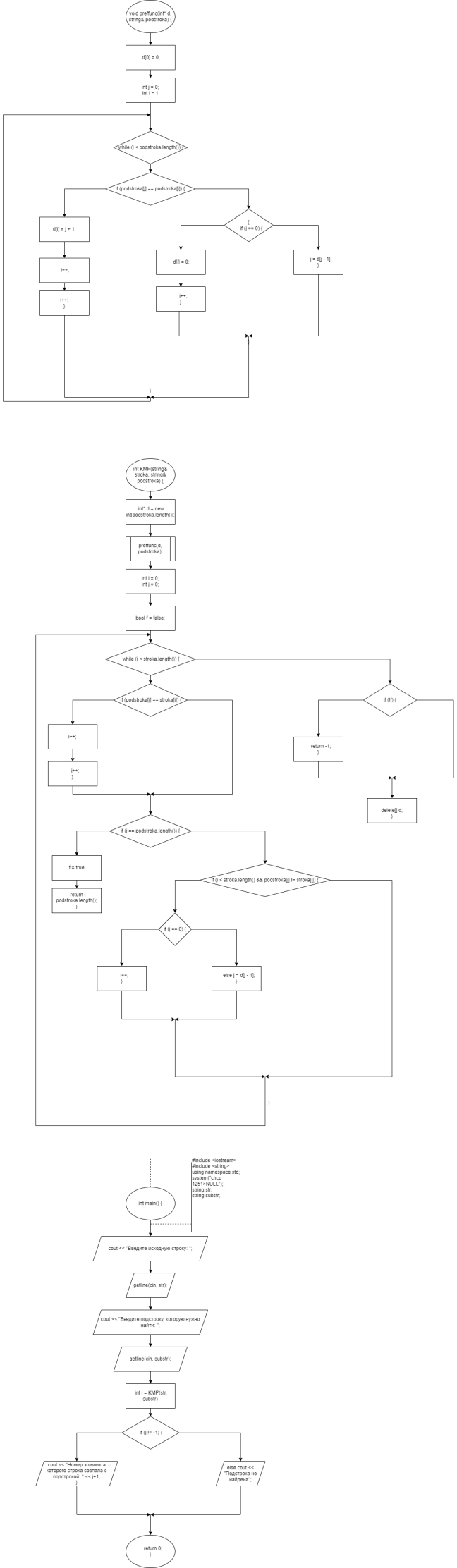
**4) Поиск подстроки в поиск:**

****

**5) Алгоритм Бойера — Мура:**

****

**6)Алгоритм Кнута-Морриса-Пратта:**

****

**Код на языке C++:**

**1) Линейный поиск:**

#include <iostream>

using namespace std;

int listSearch(int\* a, int key, int N){

int i = 0;

while (a[i] != key) {

i++;

}

if (a[i] == key) {

return i;

}

else return 0;

}

int main() {

system("chcp 1251>NULL");

int N;

cout << "Введите кол-во элментов в массиве: ";

cin >> N;

int\* a = new int[N];

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

cout << "Введите " << i + 1 << " элемент: ";

cin >> a[i];

}

cout << "Исходный массив: ";

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

cout << a[i] << ' ';

}

cout << endl;

int key;

cout << "Введите key: ";

cin >> key;

int i =listSearch(a, key, N);

if (i != 0) {

cout << "Номер элемента, который совпал с key: " << i + 1;

}

else cout << "Элемента нет";

return 0;

}

**2) Бинарный поиск:**

#include <iostream>

using namespace std;

int binSearch(int\* a, int N, int key) {

int left = 0;

int right = N - 1;

bool f = false;

while (!f) {

int mid = (left+right)/2;

if (a[mid] == key) {

f = true;

return mid;

}

if (a[mid] > key) {

right = mid - 1;

}

if (a[mid] < key) {

left = mid +1;

}

}

}

int main() {

system("chcp 1251>NULL");

int N;

cout << "Введите кол-во элментов в массиве: ";

cin >> N;

int\* a = new int[N];

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

cout << "Введите " << i + 1 << " элемент: ";

cin >> a[i];

}

cout << "Исходный массив: ";

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

cout << a[i] << ' ';

}

cout << endl;

int key;

cout << "Введите key: ";

cin >> key;

int i = binSearch(a, N, key);

if (i != 0) {

cout << "Номер элемента, который совпал с key: " << i + 1;

}

else cout << "Элемента нет";

return 0;

}

**3) Интерполяционный поиск:**

#include <iostream>

using namespace std;

int binSearch(int\* a, int N, int key) {

int left = 0;

int right = N - 1;

bool f = false;

while (!f) {

int mid = left + ((key - a[left]) \* (right - left) / (a[right] - a[left]));

if (a[mid] == key) {

f = true;

return mid;

}

if (a[mid] > key) {

right = mid - 1;

}

if (a[mid] < key) {

left = mid +1;

}

}

}

int main() {

system("chcp 1251>NULL");

int N;

cout << "Введите кол-во элментов в массиве: ";

cin >> N;

int\* a = new int[N];

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

cout << "Введите " << i + 1 << " элемент: ";

cin >> a[i];

}

cout << "Исходный массив: ";

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

cout << a[i] << ' ';

}

cout << endl;

int key;

cout << "Введите key: ";

cin >> key;

int i = binSearch(a, N, key);

if (i != 0) {

cout << "Номер элемента, который совпал с key: " << i + 1;

}

else cout << "Элемента нет";

return 0;

}

**4) Поиск подстроки в поиск:**

#include <iostream>

#include <string>

using namespace std;

int strSearch(string str, string substr) {

int d = str.length();

int k = substr.length();

int N = d-k;

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

int j;

for (j = 0; j < k; j++) {

if (str[i + j] != substr[j]) {

break;

}

}

if (j == k) {

return i;

}

}

return -1;

}

int main() {

system("chcp 1251>NULL");

system("cls");

string str;

string substr;

cout << "Введите строку: ";

getline(cin, str);

cout << "Введите подстроку: ";

getline(cin, substr);

int j= strSearch(str, substr);

if (j != -1) {

cout << "Номер элемента, с которого строка совпала с подстрокой: " << j+1;

}

else cout << "Подстрока не найдена";

return 0;

}

**5) Алгоритм Бойера — Мура:**

#include <iostream>

#include <string>

using namespace std;

void createTable(string& podstroka, int d[]) {

int Q = podstroka.length();

for (int i = 0; i < 256; i++) {

d[i] = Q;

}

for (int i = 0; i < Q - 1; i++) {

d[(unsigned char)podstroka[i]] = Q - i - 1;

}

}

int BoerMur(string& stroka, string& podstroka) {

int N1 = stroka.length();

int N2 = podstroka.length();

int d[256];

createTable(podstroka, d);

bool f = false;

for (int i = N2 - 1; i < N1; i += d[(unsigned char)stroka[i]]) {

int k = 0;

while ((k < N2) && (podstroka[N2 - 1 - k] == stroka[i - k]))

{

k++;

}

if (k == N2) {

f = true;

return i-N2+1;

}

}

if (!f) {

return -1;

}

}

int main() {

system("chcp 1251>NULL");

string str;

string substr;

cout << "Введите исходную строку: ";

getline(cin, str);

cout << "Введите подстроку, которую нужно найти: ";

getline(cin, substr);

int i = BoerMur(str, substr);

if (i != -1) {

cout << "Образ подстроки в строке найдет на элементе с номером : " << i+1;

}

else cout << "Образ не найден!";

return 0;

}

**6)Алгоритм Кнута-Морриса-Пратта:**

#include <iostream>

#include <string>

using namespace std;

void preffunc(int\* d, string& podstroka) {

d[0] = 0;

int j = 0;

int i = 1;

while (i < podstroka.length()) {

if (podstroka[j] == podstroka[i]) {

d[i] = j + 1;

i++;

j++;

}

else {

if (j == 0) {

d[i] = 0;

i++;

}

else {

j = d[j - 1];

}

}

}

}

int KMP(string& stroka, string& podstroka) {

int\* d = new int[podstroka.length()];

preffunc(d, podstroka);

int i = 0;

int j = 0;

bool f = false;

while (i < stroka.length()) {

if (podstroka[j] == stroka[i]) {

i++;

j++;

}

if (j == podstroka.length()) {

f = true;

return i - podstroka.length();

}

else if (i < stroka.length() && podstroka[j] != stroka[i]) {

if (j == 0) {

i++;

}

else j = d[j - 1];

}

}

if (!f) {

return -1;

}

delete[] d;

}

int main() {

system("chcp 1251>NULL");

string str;

string substr;

cout << "Введите исходную строку: ";

getline(cin, str);

cout << "Введите подстроку, котору нужно найти: ";

getline(cin, substr);

int i = KMP(str, substr);

if (i != -1) {

cout << "Образ подстроки в строке найдет на индексе: " << i;

}

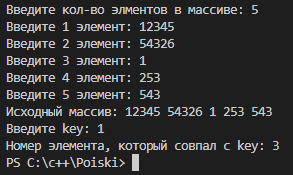
else cout << "Образ не найден!";

return 0;

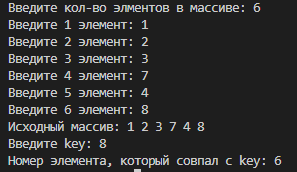
}

**Работа программы:**

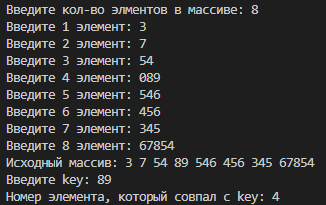
**1) Линейный поиск:**



**2) Бинарный поиск:**



**3) Интерполяционный поиск:**



**4) Поиск подстроки в поиск:**



**5) Алгоритм Бойера — Мура:**

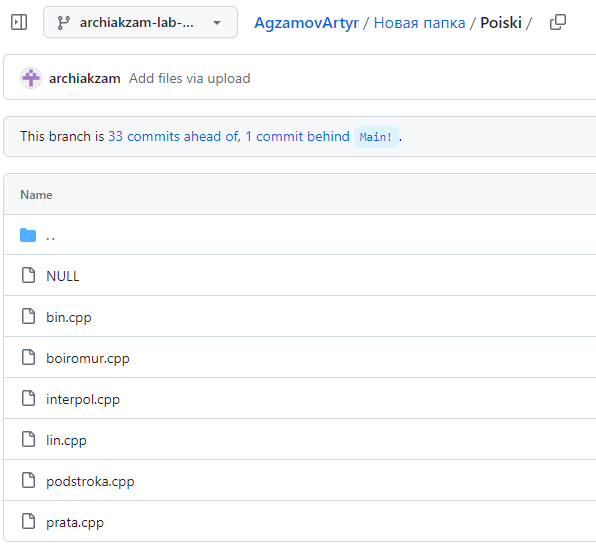


**6)Алгоритм Кнута-Морриса-Пратта:**





**Cкрины из гита:**



**Вывод:** Задача была выполнена. Всё получилось.