МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

## Факультет информационных технологий и робототехники

Кафедра программного обеспечения информационных систем и технологий

**Отчет по лабораторной работе № 1**

по дисциплине:” Алгоритмы и структуры данных”

на тему:**”** **Поиск подстроки”**

Вариант 1

Выполнил**:** студент группы 10701322

Бородкин Д.В.

Приняла**:** пр. Мисякова В.А..

Минск 2024

# Лабораторная работа №1. Поиск подстроки.

# Цель работы: Закрепление теоретических знаний и получение практических навыков применения алгоритмов поиска подстрок.

### Задание:

1. Изобразить в виде таблиц строку и шаблон со всеми сдвигами при поиске для всех алгоритмов поиска подстроки.

2. Создать программу, выполняющую следующие действия.

2.1. Ввод строки, содержащей фамилию, имя и отчество студента.

2.2. Ввод шаблона для поиска в соответствии с вариантом.

2.3. Подсчет числа сравнений символов при прямом поиске, при БМХ, КМР поиске.

.

### Код программы

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Security.Cryptography;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace TrueLab1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private int[] MatchInSubstring(string s2, char ch)

{

int comp = 0;

for (int i = s2.Length - 1; i > -1; i--)

{

comp++;

if (ch == s2[i])

{

return new int[2] { i, comp };

}

}

return new int[2] { -1, comp };

}

private int[] MismatchInText(string s1, string s2, int currentPos)

{

int comp = 0;

for (int i = s2.Length - 1; i > -1; i--)

{

comp++;

if (s2[i] != s1[currentPos + i])

{

return new int[2] { currentPos + i , comp };

}

}

return new int[2] { -1, comp };

}

// O(N \* M)

private int[] StraightSearch(string s1, string s2)

{

int m = 0;

int comp = 0;

for (int i = 0; i < s1.Length - s2.Length; i++)

{

int j;

for (j = 0; j < s2.Length; j++)

{

comp++;

if (s1[i] == s2[j])

{ i++; continue; }

else break;

}

if (j == s2.Length)

{ m = i - s2.Length + 1; break; }

else m = 0;

}

return new int[2] { m - 1, comp };

}

// O(N + M)

private int[] BMHMethod(string s1, string s2)

{

List<int> positions = new List<int>();

int i = 0;

int comp = 0;

while (i <= s1.Length - s2.Length)

{

int[] result = MismatchInText(s1, s2, i);

comp += result[1];

int mismatchIndex = result[0];

if (mismatchIndex == -1)

{

positions.Add(i);

i++;

} else

{

int[] substrres = MatchInSubstring(s2, s1[mismatchIndex]);

int mismatchIndexSubstr = substrres[0];

i = mismatchIndex - mismatchIndexSubstr;

comp += substrres[1];

}

}

return new int[2] { positions[0], comp};

}

// O(N + M)

//https://www.youtube.com/watch?v=S2I0covkyMc

private int[] KMPMethod(string s1, string s2)

{

int comp = 0;

//forming pi array

//O(M)

if (s1.Length == 0 || s2.Length == 0 || s1.Length < s2.Length) return new int[] { 1, comp };

int i = 1;

int j = 0;

int[] pi = new int[s2.Length];

while (i < s2.Length)

{

comp++;

if (s2[j] == s2[i])

{

pi[i] = j + 1; i++;

j++;

}

else

{

if (j == 0)

{

pi[i] = 0;

i++;

}

else { j = pi[j - 1]; }

}

}

//O(N)

//searching substring

i = 0;

j = 0;

while (i < s1.Length)

{

comp++;

if (s1[i] == s2[j]) {

i++;

j++;

if (j == s2.Length) return new int[] { i - s2.Length, comp };

} else

{

if (j > 0)

{

j = pi[j - 1];

} else

{

i++;

}

}

}

return new int[] { 0, comp };

}

private void findButton\_Click(object sender, EventArgs e)

{

string s1 = stringTextBox.Text;

string s2 = substringTextBox.Text;

int[] res = StraightSearch(s1, s2);

if (straightMethodButton.Checked)

res = StraightSearch(s1, s2);

if (BMHButton.Checked)

res = BMHMethod(s1, s2);

if (KMPButton.Checked)

res = KMPMethod(s1, s2);

compTextBox.Text = res[1].ToString();

if (res[0] > 0)

resultTextBox.Text = "i = " + (res[0] + 1);

else

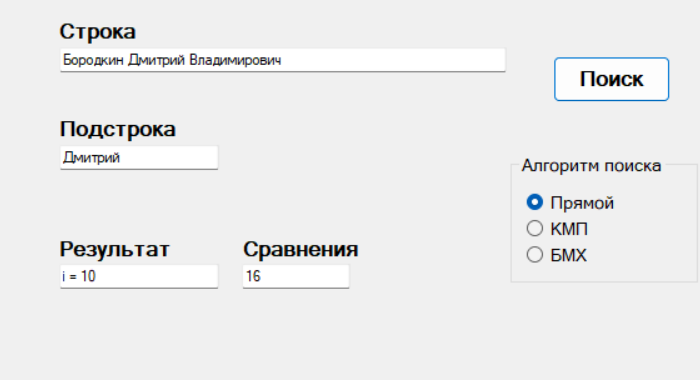
resultTextBox.Text = "Не найдено";

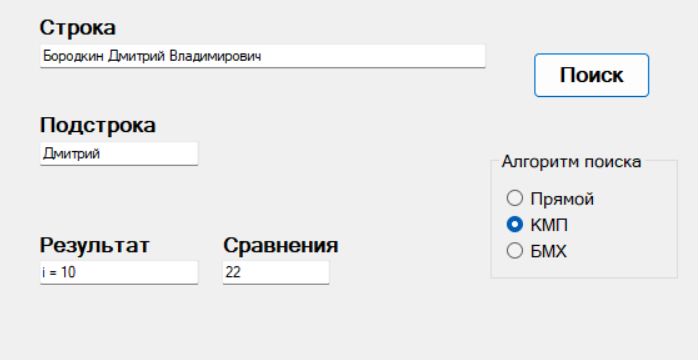
}

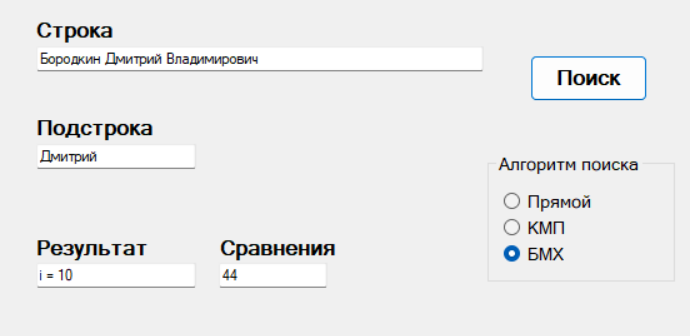
}

}

### Скриншот результата







### Результат определения функции f(N)

F(N) = O(n \* m)

## Контрольные вопросы

1. Прямой поиск текста, также известный как последовательный поиск, это простейший метод поиска подстроки в строке. Он работает путем сравнения каждого символа в исходной строке с первым символом искомой подстроки. [Если символы совпадают, алгоритм продолжает сравнивать последующие символы до тех пор, пока не найдет полное совпадение или несоответствие](https://habr.com/ru/articles/111449/).
2. Алгоритм Бойера-Мура-Хорспула (БМХ) - это эффективный метод поиска подстроки, который начинает сравнение с конца искомой подстроки, а не с начала. [Если символ в тексте не совпадает с соответствующим символом в подстроке, алгоритм смещает подстроку вперед на расстояние, определенное таблицей смещений](https://habr.com/ru/articles/111449/).
3. Таблица смещений для БМХ-поиска строится для каждого символа в искомой подстроке. [Если символ в тексте не совпадает с соответствующим символом в подстроке, подстрока смещается вперед на расстояние, указанное в таблице смещений для этого символа](https://habr.com/ru/articles/111449/).
4. Алгоритм Кнута-Морриса-Пратта (КМП) - это еще один эффективный метод поиска подстроки. Он использует префикс-функцию для определения, сколько символов можно пропустить при обнаружении несовпадения. [Это позволяет алгоритму пропускать проверку уже проверенных символов, что ускоряет процесс поиска](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%9A%D0%BD%D1%83%D1%82%D0%B0_%E2%80%94_%D0%9C%D0%BE%D1%80%D1%80%D0%B8%D1%81%D0%B0_%E2%80%94_%D0%9F%D1%80%D0%B0%D1%82%D1%82%D0%B0).
5. [Префикс-функция для строки и позиции в ней - это длина наибольшего собственного (не равного всей подстроке) префикса подстроки, который одновременно является суффиксом этой подстроки](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D1%84%D0%B8%D0%BA%D1%81-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F).

## Вывод: Я изучил различные методы текстового поиска такие, как прямой, БМХ, КМР и написал программу для поиска определенного шаблона в тексте на языке С++.