**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САПР**

отчет

**по лабораторной работе №1**

**по дисциплине «Программирование»**

|  |  |  |
| --- | --- | --- |
| Студент гр. 1302 |  | Гордиенко М. Ю. |
| Преподаватель |  | Калмычков В.А. |

Санкт-Петербург

2022

Оглавление

[Исходная формулировка 3](#_Toc98185031)

[Контрольные примеры 3](#_Toc98185032)

[Формальная постановка задачи 3](#_Toc98185033)

[Формат хранения данных 3](#_Toc98185034)

[Ограничения, обусловленные выполнениям компьютера 3](#_Toc98185035)

[Макет ввода/вывода 3](#_Toc98185036)

[Средства обеспечения ввода/вывода 4](#_Toc98185037)

[Алгоритм решения 4](#_Toc98185038)

[Пользовательский тип данных 8](#_Toc98185039)

[Текст программы 8](#_Toc98185040)

[Результат работы программы 20](#_Toc98185041)

[Вывод 22](#_Toc98185042)

# Исходная формулировка

Проверить, находится ли в заданной строке заданная подстрока.

# Контрольные примеры

1.1 и 1.2

Ограничитель (для 1.1) – (.)

Лимит символов (для 1.2) – 9 и 8

Маркер – @

Строка – potatotar

Подстрока – potatota

Ответ: Строка содержит подстроку

2.1 и 2.2

Ограничитель (для 2.1) – (.)

Лимит символов (для 2.2) – 2 и 7

Строка – rurukru

Подстрока – uk

Ответ: Строка содержит подстроку

# Формальная постановка задачи

Дано: строка и подстрока

Найти: содержание в строке подстроки

Способ решение: сравнение элементов строки с элементами подстроки

# Формат хранения данных

|  |  |  |
| --- | --- | --- |
| Идентификатор типа | Тип | Назначение |
| f, f1, f2 | fstream | Файловые переменные |
| s, s1 | str | Пользовательский тип |
| stroka, podstroka | char | Строки |
| mark, ogr | char | Ограничители |
| i, j, i1, j1 | int | Счётчики |
| k, sum, r | int | Дополнительные переменные |
| l | int | Длинна строки |

# Ограничения, обусловленные выполнениям компьютера

Целый тип int имеет размер 4 байта (32 бита). Минимальное значение -2 147 483 648, максимальное значение 2 147 483 647.

Символьный тип char. Минимальное значение -128, максимальное значение 128.

# Макет ввода/вывода

|  |  |
| --- | --- |
| Приветствие | f2 << "Гордиенко Михаил" << endl << "Версия 1.1/1.2/2.1/2.2" << endl << "Группа 1302" << endl; |
| Некорректный размер | f2 << "некорректная длина, скорректирована в 0" << '\n';  f2 << "некорректная длина, скорректирована в N" << '\n'; |
| Наличие символов | f << " пустая строка";  f << " пустая подстрока"; |
| Пояснения | f2 << "Введённая строка" << endl;  f2 << endl << "Введённая подстрока" << endl;  f2 << "Результат обработки" << endl; |
| Результат работы | f2 << "В строке содержится подстрока";  f2 << "В строке не содержится подстрока"; |
| Ввод ограничителя | f >> ogr; |
| Ввод маркера | f >> mark; |
| Ввод длины строки | f >> l; |

# Средства обеспечения ввода/вывода

Ввод из файла:

f>>

f1>>

Вывод в консоль:

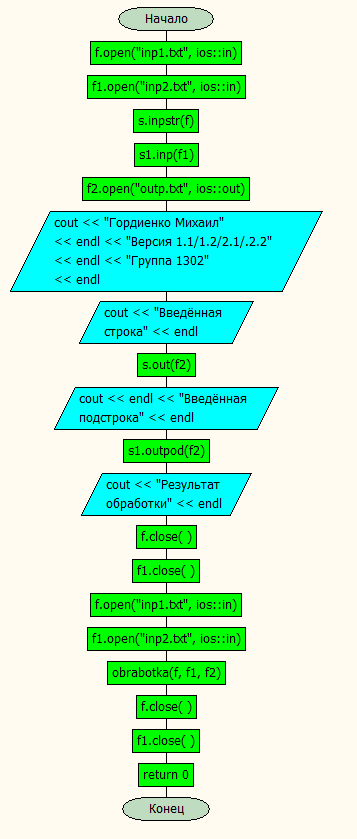
cout<<

Вывод в файл:

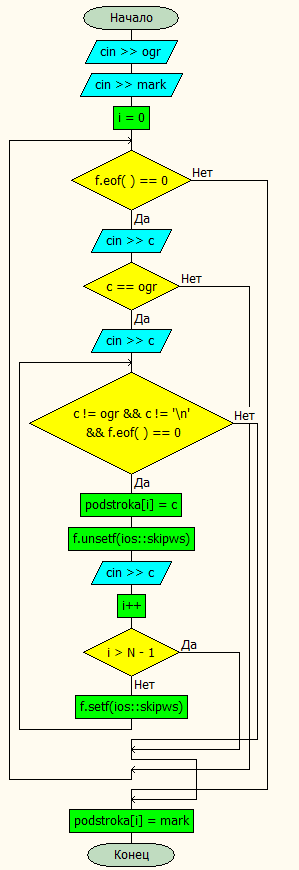
f2<<

# Алгоритм решения

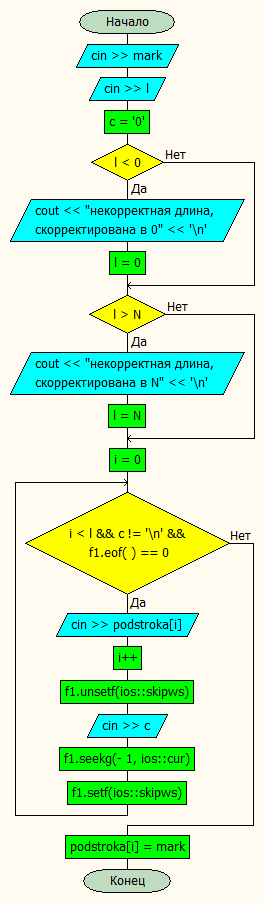
main () - во всех версиях



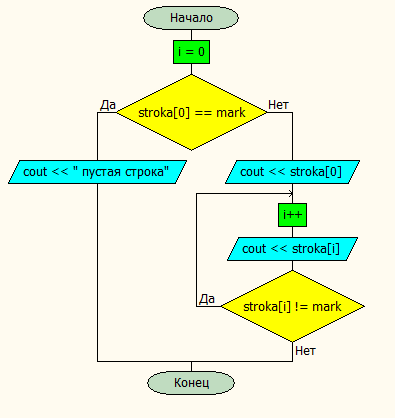
void inp(fstream& f) – для 1.1 и 2.1



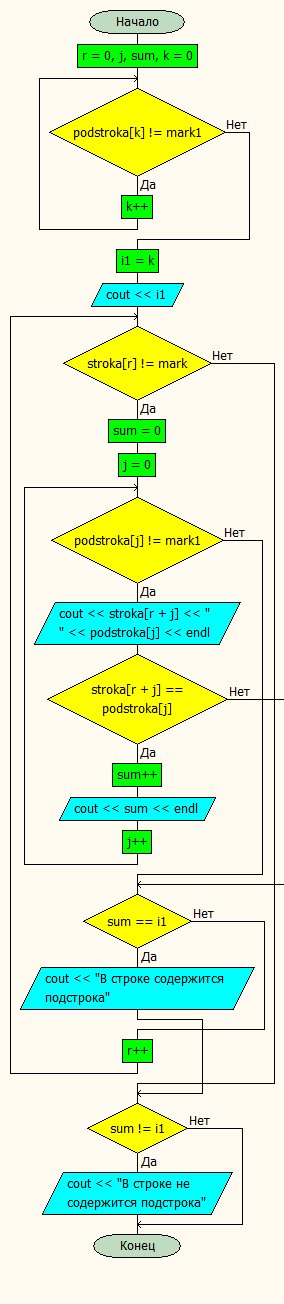
void inp(fstream& f, fstream& f2) – для 1.2 и 2.2



void out(fstream& f) – для всех версий



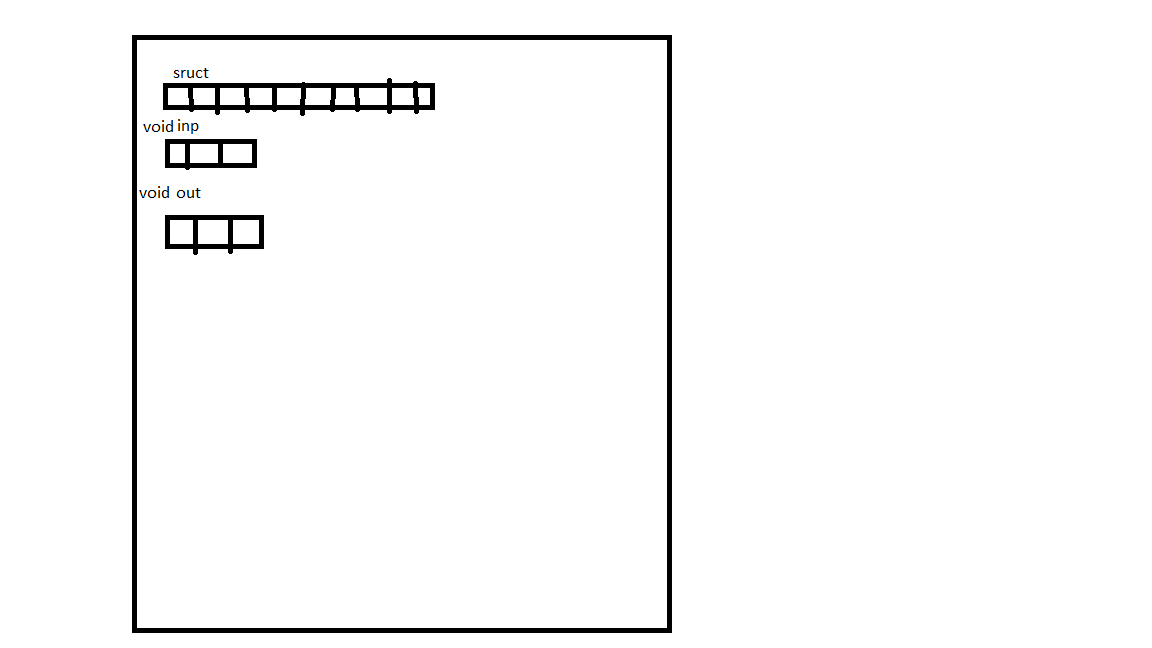
void obrabotka(fstream& f, fstream& f1, fstream& f2) – для всех версий



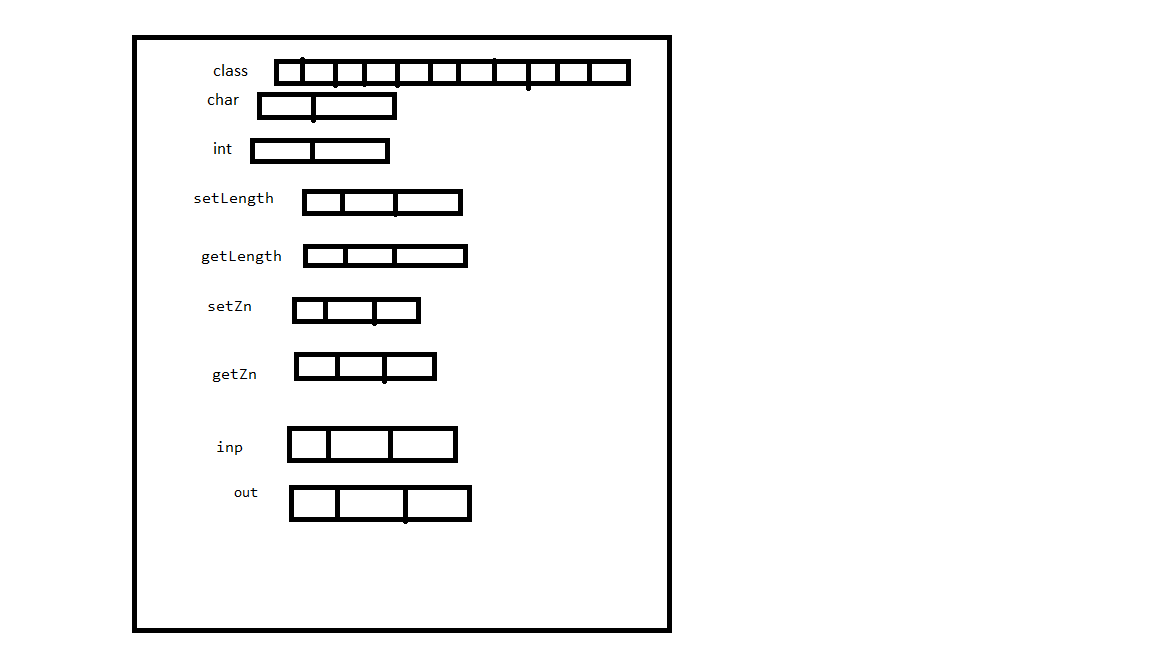
# Пользовательский тип данных

В программе используется 1 пользовательский тип данных str

Для 1.1 и 1.2:



Для 2.1 и 2.2:



# Текст программы

1.1

//Гордиенко Михаил;

//Группа 1302

//Версия: 1.1;

#include <fstream>

#include <iostream>

using namespace std;

const unsigned N = 25;

struct str

{

char stroka[N + 1], podstroka[N + 1];

char mark;

void inpstr(fstream& f)

{

char ogr;

f >> ogr;

f >> mark;

char c;

unsigned i = 0;

while (f.eof() == 0)

{

f >> c;

if (c == ogr)

{

f >> c;

while (c != ogr && c != '\n' && f.eof() == 0)

{

stroka[i] = c;

f.unsetf(ios::skipws);

f >> c;

i++;

if (i > N - 1)

break;

f.setf(ios::skipws);

}

break;

}

else

continue;

}

stroka[i] = mark;

}

void inp(fstream& f)

{

char ogr;

f >> ogr;

f >> mark;

char c;

int i = 0;

while (f.eof() == 0)

{

f >> c;

if (c == ogr)

{

f >> c;

while (c != ogr && c != '\n' && f.eof() == 0)

{

podstroka[i] = c;

f.unsetf(ios::skipws);

f >> c;

i++;

if (i > N - 1)

break;

f.setf(ios::skipws);

}

break;

}

else

continue;

}

podstroka[i] = mark;

}

void out(fstream& f)

{

unsigned i = 0;

if (stroka[0] == mark)

f << " пустая строка";

else

{

f << stroka[0];

do

{

i++;

f << stroka[i];

} while (stroka[i] != mark);

}

}

void outpod(fstream& f)

{

unsigned i = 0;

if (podstroka[0] == mark)

f << " пустая подстрока";

else

{

f << podstroka[0];

do

{

i++;

f << podstroka[i];

} while (podstroka[i] != mark);

}

}

};

void obrabotka(fstream& f, fstream& f1, fstream& f2)

{

char stroka[N + 1], mark;

char ogr;

f >> ogr;

f >> mark;

char c;

unsigned i = 0;

while (f.eof() == 0)

{

f >> c;

if (c == ogr)

{

f >> c;

while (c != ogr && c != '\n' && f.eof() == 0)

{

stroka[i] = c;

f.unsetf(ios::skipws);

f >> c;

i++;

if (i > N - 1)

break;

f.setf(ios::skipws);

}

break;

}

else

continue;

}

stroka[i] = mark;

char podstroka[N + 1], mark1;

char ogr1;

f1 >> ogr1;

f1 >> mark1;

char c1;

unsigned i1 = 0;

while (f1.eof() == 0)

{

f1 >> c1;

if (c1 == ogr1)

{

f1 >> c1;

while (c1 != ogr && c1 != '\n' && f1.eof() == 0)

{

podstroka[i1] = c1;

f1.unsetf(ios::skipws);

f1 >> c1;

i1++;

if (i1 > N - 1)

break;

f.setf(ios::skipws);

}

break;

}

else

continue;

}

podstroka[i1] = mark1;

int l = 0, j, sum, k=0;

while (podstroka[k] != mark1)

k++;

i1 = k;

cout << i1;

while (stroka[l] != mark)

{

sum = 0;

j = 0;

while (podstroka[j] != mark1)

{

cout << stroka[l + j] << " " << podstroka[j] << endl;

if (stroka[l + j] == podstroka[j]) sum++;

else

break;

cout << sum << endl;

j++;

}

if (sum == i1)

{

f2 << "В строке содержится подстрока";

break;

}

l++;

}

}

int main()

{

fstream f, f1, f2;

f.open("inp1.txt", ios::in);

f1.open("inp2.txt", ios::in);

str s, s1;

s.inpstr(f);

s1.inp(f1);

f2.open("outp.txt", ios::out);

f2 << "Гордиенко Михаил" << endl << "Версия 1.1" << endl << "Группа 1302" << endl;

f2 << "Введённая строка" << endl;

s.out(f2);

f2 << endl << "Введённая подстрока" << endl;

s1.outpod(f2);

f2 << '\n';

f2 << "Результат обработки" << endl;

f.close();

f1.close();

f.open("inp1.txt", ios::in);

f1.open("inp2.txt", ios::in);

obrabotka(f, f1, f2);

f.close();

f1.close();

return 0;

}

1.2

//Гордиенко Михаил;

//Группа 1302

//Версия: 1.2;

#include <fstream>

#include <iostream>

using namespace std;

const unsigned N = 25;

struct str

{

char stroka[N + 1], podstroka[N + 1];

char mark;

void inp2(fstream& f1, fstream& f2)

{

int l;

f1 >> mark;

f1 >> l;

char c = '0';

if (l < 0)

{

f2 << "некорректная длина, скорректирована в 0" << '\n';

l = 0;

}

if (l > N)

{

f2 << "некорректная длина, скорректирована в N" << '\n';

l = N;

}

int i = 0;

while (i < l && c != '\n' && f1.eof() == 0)

{

f1 >> podstroka[i];

i++;

f1.unsetf(ios::skipws);

f1 >> c;

f1.seekg(-1, ios::cur);

f1.setf(ios::skipws);

}

podstroka[i] = mark;

}

void inp1(fstream& f, fstream& f2)

{

int l;

f >> mark;

f >> l;

char c = '0';

if (l < 0)

{

f2 << "некорректная длина, скорректирована в 0" << '\n';

l = 0;

}

if (l > N)

{

f2 << "некорректная длина, скорректирована в N" << '\n';

l = N;

}

int i = 0;

while (i < l && c != '\n' && f.eof() == 0)

{

f >> stroka[i];

i++;

f.unsetf(ios::skipws);

f >> c;

f.seekg(-1, ios::cur);

f.setf(ios::skipws);

}

stroka[i] = mark;

}

void out(fstream& f)

{

unsigned i = 0;

if (stroka[0] == mark)

f << " пустая строка";

else

{

f << stroka[0];

do

{

i++;

f << stroka[i];

} while (stroka[i] != mark);

}

}

void outpod(fstream& f)

{

unsigned i = 0;

if (podstroka[0] == mark)

f << " пустая строка";

else

{

f << podstroka[0];

do

{

i++;

f << podstroka[i];

} while (podstroka[i] != mark);

}

}

};

void obrabotka(fstream& f, fstream& f1, fstream& f2)

{

char stroka[N + 1], mark;

int l;

f >> mark;

f >> l;

char c = '0';

if (l < 0)

{

l = 0;

}

if (l > N)

{

l = N;

}

int i = 0;

while (i < l && c != '\n' && f1.eof() == 0)

{

f >> stroka[i];

i++;

f.unsetf(ios::skipws);

f >> c;

f.seekg(-1, ios::cur);

f.setf(ios::skipws);

}

stroka[i] = mark;

char podstroka[N + 1], mark1;

int l1;

f1 >> mark1;

f1 >> l1;

char c1 = '0';

if (l1 < 0)

{

l1 = 0;

}

if (l1 > N)

{

l1 = N;

}

int i1 = 0;

while (i1 < l && c1 != '\n' && f1.eof() == 0)

{

f1 >> podstroka[i1];

i1++;

f1.unsetf(ios::skipws);

f1 >> c1;

f1.seekg(-1, ios::cur);

f1.setf(ios::skipws);

}

podstroka[i1] = mark1;

int r = 0, j, sum, k = 0;

while (podstroka[k] != mark1)

k++;

i1 = k;

cout << i1;

while (stroka[r] != mark)

{

sum = 0;

j = 0;

while (podstroka[j] != mark1)

{

cout << stroka[r + j] << " " << podstroka[j] << endl;

if (stroka[r + j] == podstroka[j]) sum++;

else

break;

cout << sum << endl;

j++;

}

if (sum == i1)

{

f2 << "В строке содержится подстрока";

break;

}

r++;

}

if (sum != i1)

{

f2 << "В строке не содержится подстрока";

}

}

int main()

{

fstream f, f1, f2;

f.open("inp1.txt", ios::in);

f1.open("inp2.txt", ios::in);

str s, s1;

s.inp1(f, f2);

s1.inp2(f1, f2);

f2.open("outp.txt", ios::out);

f2 << "Гордиенко Михаил" << endl << "Версия 1.2" << endl << "Группа 1302" << endl;

f2 << "Введённая строка" << endl;

s.out(f2);

f2 << endl << "Введённая подстрока" << endl;

s1.outpod(f2);

f2 << '\n';

f2 << "Результат обработки" << endl;

f.close();

f1.close();

f.open("inp1.txt", ios::in);

f1.open("inp2.txt", ios::in);

obrabotka(f, f1, f2);

f.close();

f1.close();

return 0;

}

2.1

//Гордиенко Михаил

//Группа 1302

//Версия: 2.1

#include<fstream>

#include<iostream>

using namespace std;

const unsigned N = 25;

class strL

{

char stroka[N], podstroka[N];

int length, length1;

public:

void setLength(int l)

{

length = l;

}

int getLength()

{

return length;

}

void setLength1(int l1)

{

length1 = l1;

}

int getLength1()

{

return length1;

}

void setZn(unsigned i, char c)

{

stroka[i] = c;

}

char getZn(unsigned i)

{

return stroka[i];

}

void setZn1(unsigned i1, char c1)

{

stroka[i1] = c1;

}

char getZn1(unsigned i1)

{

return stroka[i1];

}

void inp1(fstream& f);

void inp2(fstream& f1);

void out(fstream& f2);

void outpod(fstream& f2);

};

void strL::inp1(fstream& f)

{

char ogr;

f >> ogr;

char c;

unsigned i = 0;

while (f.eof() == 0)

{

f >> c;

if (c == ogr) {

f >> c;

while (c != ogr && c != '\n' && f.eof() == 0)

{

setZn(i, c);

f.unsetf(ios::skipws);

f >> c;

i++;

if (i > N)

break;

f.setf(ios::skipws);

}

setLength(i);

break;

}

else

continue;

}

}

void strL::inp2(fstream& f1)

{

char ogr1;

f1 >> ogr1;

char c1;

unsigned i1 = 0;

while (f1.eof() == 0)

{

f1 >> c1;

if (c1 == ogr1)

{

f1 >> c1;

while (c1 != ogr1 && c1 != '\n' && f1.eof() == 0)

{

setZn1(i1, c1);

f1.unsetf(ios::skipws);

f1 >> c1;

i1++;

if (i1 > N)

break;

f1.setf(ios::skipws);

}

setLength1(i1);

break;

}

else

continue;

}

}

void strL::out(fstream& f2)

{

if (getLength() == 0)

f2 << "пустая строка";

else

{

f2 << "длина строки: " << getLength() << endl;

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

{

f2 << getZn(i);

}

}

f2 << endl;

}

void strL::outpod(fstream& f2)

{

if (getLength() == 0)

f2 << "пустая строка";

else

{

f2 << "длина подстроки: " << getLength1() << endl;

for (int i1 = 0; i1 < getLength1(); i1++)

{

f2 << getZn1(i1);

}

}

f2 << endl;

}

void obrabotka(fstream& f, fstream& f1, fstream& f2)

{

char ogr, stroka[N], podstroka[N];

f >> ogr;

char c;

unsigned i = 0;

while (f.eof() == 0)

{

f >> c;

if (c == ogr) {

f >> c;

while (c != ogr && c != '\n' && f.eof() == 0)

{

stroka[i] = c;

f.unsetf(ios::skipws);

f >> c;

i++;

if (i > N)

break;

f.setf(ios::skipws);

}

break;

}

else

continue;

}

cout << i << endl;

char ogr1;

f1 >> ogr1;

char c1;

unsigned i1 = 0;

while (f1.eof() == 0)

{

f1 >> c1;

if (c1 == ogr1)

{

f1 >> c1;

while (c1 != ogr1 && c1 != '\n' && f1.eof() == 0)

{

podstroka[i1] = c1;

f1.unsetf(ios::skipws);

f1 >> c1;

i1++;

if (i1 > N)

break;

f1.setf(ios::skipws);

}

break;

}

else

continue;

}

cout << i1 << endl;

int a = i, b = i1, sum;

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

{

sum = 0;

for (int j = 0; j < b; j++)

{

cout << stroka[i + j] << " " << podstroka[j] << endl;

if (stroka[i + j] == podstroka[j]) sum++;

else

break;

cout << sum << endl;

}

if (sum == i1)

{

f2 << "В строке содержится подстрока";

break;

}

}

if (sum != i1)

{

f2 << "В строке не содержится подстрока";

}

}

int main()

{

fstream f, f1, f2;

f.open("inp1.txt", ios::in);

f1.open("inp2.txt", ios::in);

f2.open("outp.txt", ios::out);

strL stroka, podstroka;

stroka.inp1(f);

podstroka.inp2(f1);

f2 << "Гордиенко Михаил" << endl << "Версия: 2.1" << endl;

f2 << "введённая строка:" << endl;

stroka.out(f2);

f2 << "введённая подстрока:" << endl;

podstroka.outpod(f2);

f.close();

f1.close();

f.open("inp1.txt", ios::in);

f1.open("inp2.txt", ios::in);

f2 << "результат обработки:" << endl;

obrabotka(f, f1, f2);

f.close();

f1.close();

f2.close();

return 0;

}

2.2

//Гордиенко Михаил

//Группа 1302

//Версия: 2.2

#include<fstream>

#include<iostream>

using namespace std;

const unsigned N = 25;

class strL

{

char stroka[N], podstroka[N];

int length, length1;

public:

void setLength(int l)

{

length = l;

}

int getLength()

{

return length;

}

void setLength1(int l1)

{

length1 = l1;

}

int getLength1()

{

return length1;

}

void setZn(unsigned i, char c)

{

stroka[i] = c;

}

char getZn(unsigned i)

{

return stroka[i];

}

void setZn1(unsigned i1, char c1)

{

stroka[i1] = c1;

}

char getZn1(unsigned i1)

{

return stroka[i1];

}

void inp1(fstream& f, fstream& f2);

void inp2(fstream& f1, fstream& f2);

void out(fstream& f2);

void outpod(fstream& f2);

};

void strL::inp1(fstream& f, fstream& f2)

{

int l;

f >> l;

char c = '0', b;

if (l < 0)

{

f2 << "некорректная длина, скорректирована в 0" << '\n';

l = 0;

}

if (l > N)

{

f2 << "некорректная длина, скорректирована в N" << '\n';

l = N;

}

int i = 0;

while (i < l && c != '\n' && f.eof() == 0) {

f >> b;

setZn(i, b);

i++;

f.unsetf(ios::skipws);

f >> c;

f.seekg(-1, ios::cur);

f.setf(ios::skipws);

}

setLength(i);

}

void strL::inp2(fstream& f1, fstream& f2)

{

int l1;

f1 >> l1;

char c1 = '0', b1;

if (l1 < 0) {

f2 << "некорректная длина, скорректирована в 0" << '\n';

l1 = 0;

}

if (l1 > N) {

f2 << "некорректная длина, скорректирована в N" << '\n';

l1 = N;

}

int i1 = 0;

while (i1 < l1 && c1 != '\n' && f1.eof() == 0) {

f1 >> b1;

setZn1(i1, b1);

i1++;

f1.unsetf(ios::skipws);

f1 >> c1;

f1.seekg(-1, ios::cur);

f1.setf(ios::skipws);

}

setLength1(i1);

}

void strL::out(fstream& f2)

{

if (getLength() == 0)

f2 << "пустая строка";

else

{

f2 << "длина строки: " << getLength() << endl;

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

{

f2 << getZn(i);

}

}

f2 << endl;

}

void strL::outpod(fstream& f2)

{

if (getLength() == 0)

f2 << "пустая строка";

else

{

f2 << "длина подстроки: " << getLength1() << '\n';

for (int i1 = 0; i1 < getLength1(); i1++)

{

f2 << getZn1(i1);

}

}

f2 << '\n';

}

void obrabotka(fstream& f, fstream& f1, fstream& f2)

{

char stroka[N], podstroka[N];

int l;

f >> l;

char c = '0', b;

if (l < 0)

{

l = 0;

}

if (l > N)

{

l = N;

}

int i = 0;

while (i < l && c != '\n' && f.eof() == 0)

{

f >> b;

stroka[i] = b;

i++;

f.unsetf(ios::skipws);

f >> c;

f.seekg(-1, ios::cur);

f.setf(ios::skipws);

}

cout << i << endl;

int l1;

f1 >> l1;

char c1 = '0', b1;

if (l1 < 0)

{

l1 = 0;

}

if (l1 > N)

{

l1 = N;

}

int i1 = 0;

while (i1 < l1 && c1 != '\n' && f1.eof() == 0)

{

f1 >> b1;

podstroka[i1] = b1;

i1++;

f1.unsetf(ios::skipws);

f1 >> c1;

f1.seekg(-1, ios::cur);

f1.setf(ios::skipws);

}

cout << i1 << endl;

int sum;

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

{

sum = 0;

for (int j1 = 0; j1 < i1; j1++)

{

cout << stroka[j + j1] << " " << podstroka[j1] << endl;

if (stroka[j + j1] == podstroka[j1]) sum++;

else

break;

cout << sum << endl;

}

if (sum == i1)

{

f2 << "В строке содержится подстрока";

break;

}

}

if (sum != i1)

{

f2 << "В строке не содержится подстрока";

}

}

int main()

{

fstream f, f1, f2;

f.open("inp1.txt", ios::in);

f1.open("inp2.txt", ios::in);

f2.open("outp.txt", ios::out);

strL stroka, podstroka;

stroka.inp1(f, f2);

podstroka.inp2(f1, f2);

f2 << "Гордиенко Михаил" << endl << "Версия: 2.2" << endl;

f2 << "введённая строка:" << endl;

stroka.out(f2);

f2 << "введённая подстрока:" << endl;

podstroka.outpod(f2);

f.close();

f1.close();

f.open("inp1.txt", ios::in);

f1.open("inp2.txt", ios::in);

f2 << "результат обработки:" << endl;

obrabotka(f, f1, f2);

f.close();

f1.close();

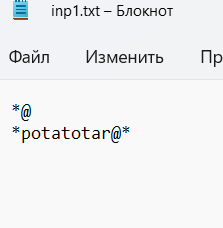
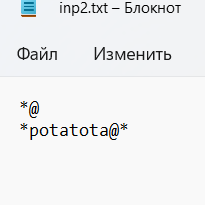
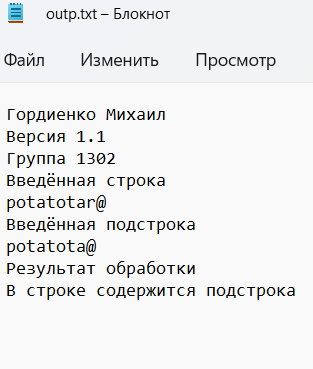
f2.close();

return 0;

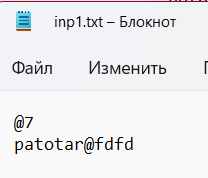
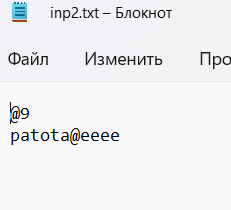
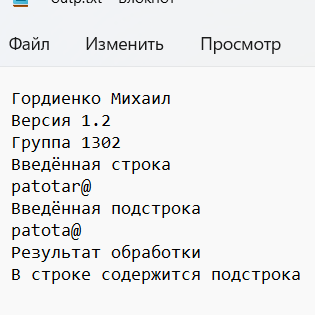
}

# Результат работы программы

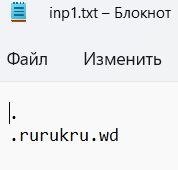
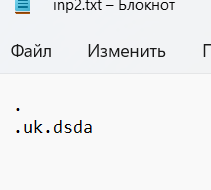
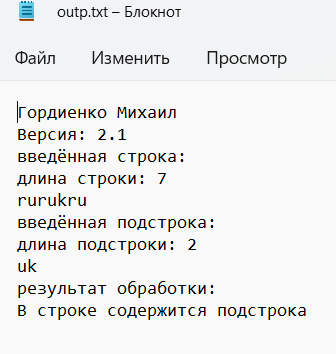
Версии 1.1

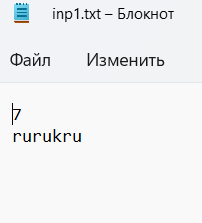
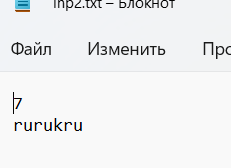
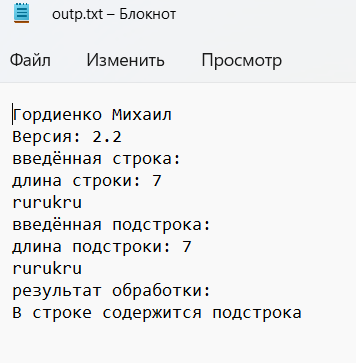
Версия 1.2

Версия 2.1

Версия 2.2

# Вывод

В процессе выполнения работы изучил классы и структуры.