Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №9**

Дисциплина: Основы алгоритмизации и программирования

Тема: Строковый ввод-вывод

Вариант 15

Выполнил:

Студент гр. ИВТ-20-2б

Чувашев Максим Алексеевич

Проверил:

Доцент кафедры ИТАС  
Полякова О.А

Пермь, 2021

**Цель работы**

Работа с текстовыми файлами, ввод-вывод текстовой информации и ее хранение на внешних носителях.

**Постановка задачи**

1. Скопировать из файла F1 в файл F2 все строки,

заканчивающиеся на букву «А» и расположенные между

строками с номерами N1 и N2.

1. Определить номер той строки, в которой больше всего

букв «А», файла F2.

**Анализ задачи**

1. Типы данных, которые будут использованы:
   1. Тип integer:

* int count; // Переменная, которая хранит количество строк.
* int N1, N2; // Переменные, которые хранят номера строк, между которыми будет осуществляется поиск.
* int i; // Счетчик цикла, который проходится по строкам.
* int countS1; // Переменная, которая хранит количество букв “a” и “A” в текущей строке файла F2.
* int countS2; // Переменная, которая хранит количество символов “a” и “A” в предыдущей строке файла F2.
* int numb; // Переменная, которая хранит номер строки, содержащей наибольшее количество символов “a” и “A” в файле F2.
* int size; // Переменная, хранящая количество символов в строке.
* int DataF(int count, string path\_F1) // Функция, которая будет запрашивать у пользователя количество строк в файле F1.
* int cpy\_F2(string str, string path\_F2) // Функция, которая будет открывать файл F2 для записи, и будет записывать строку str в этот файл, но только при условии того, что последним символом является символ ‘a’ или ‘A’.
* int FillN(int \*N1, int \*N2, int count, bool \*f) // Функция, которая будет запрашивать у пользователя ввод для переменных N1 и N2.
* int countSTR\_F1(string path\_F1, string path\_F2, bool f, int count, int N1, int N2) // Функция, которая открывает файл F1 для чтения, проходится по всем строчкам файла и будет вызывать и передавать в функцию cpy\_F2() только те строчки, которые находятся между строчкой с номеров N1 и строчкой с номером N2.
* int STR\_F2(string path\_F2, bool f, int count) // Функция, которая открывает файл F2 для чтения. Если открыть файл удалось, то будет проходиться по всем строкам файла и будет считать все встретившиеся в каждой строке символы ‘a’ или ‘A’ и запоминать номер строки в которой больше всего этих символов, а так же количество этих символов в этой строке.
* int main() // Главная функция программы, которая отвечает за вызов практически всех функций.
  1. Тип string:
* string str; // Переменная, хранящая строку, с которой мы работаем в данном цикле.
* string path\_F1; // Переменная, хранящая полное название первого файла.
* string path\_F2; // Переменная, хранящая полное название второго файла.
  1. Тип bool:
* bool f; // Флаг, который завершает выполнение программы, при возникновении ошибки.
  1. Тип char:
* char symb; // Переменная, которая будет хранить текущий символ при прохождению по строке в файле. Данная переменная будет сравниваться с символом “a” и “A”.
  1. Тип void:
* void fill\_F1(string path\_F1, bool \*f, int count) // Функция, которая открывает файл F1 для записи и просит пользователя заполнить строки которые записываются в файл F1.
  1. Класс ifstream:
* ifstream fin; // Создаем объект класса ifstream, который позволит нам взаимодействовать с файловыми потоками, но только для чтения.
  1. Класс ofstream:
* ofstream fout; // Создаем объект класса ofstream, который позволит нам взаимодействовать с файловыми потоками, но только для записи.

1. Какие с этими данными надо выполнить действия:
   1. int count;

* Программа просит пользователя присвоить данной переменной значение. Важно, чтобы введенное число было не меньше 3, так как в противном случае мы не сможем найти строки, которые находятся между строками N1 и N2.

cout << "Введите количество строк находящихся в файле " << path\_F1 << " :\t";

cin >> count;

while (count < 3)

{

cout << "Минимальное количество строк в файле должно быть не меньше 3!" << endl;

cout << "Введите количество строк находящихся в файле " << path\_F1 << " :\t";

cin >> count;

}

* Эта переменная используется как условие выхода из цикла в функции fill\_F1().

for (int i = 1; i <= count; i++) {

...

}

* Так же эта переменная используется в функции FillN(), в данной функции эта переменная является одним из условий ввода переменных N1 и N2. Оно является их максимально возможным значением.

cout << "Между какими строчками хотите искать совпадения?" << endl;

cout << "Введите номер первой строки: ";

cin >> \*N1;

while (\*N1 < 1 || \*N1 >= count)

{

cout << "Введен некорректный номер!" << endl;

cout << "Введите номер первой строки: ";

cin >> \*N1;

}

cout << "Введите номер второй строки: ";

cin >> \*N2;

while (\*N2 < 1 || \*N2 > count || ((\*N2 - \*N1) <= 1))

{

if ((\*N2 - \*N1) == 1)

{

cout << "Между данными строками нет строк!" << endl;

\*f = 1;

return \*f;

}

cout << "Введен некорректный номер!" << endl;

cout << "Введите номер второй строки: ";

cin >> \*N2;

}

* Так же данная переменная в функции countSTR\_F1() будет отвечать за номер текущей строки считанной из файла F1.

while (!fin.eof())

{

str = "";

getline(fin, str);

count++;

if (count > N1 && count < N2)

{

cpy\_F2(str, path\_F2);

}

}

* В функции STR\_F2() данная переменная будет хранить номер текущей считанной строки из файла F2.

while (!fin.eof())

{

count++;

countS1 = 0;

str = "";

getline(fin, str);

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

{

symb = str[i];

if (symb == 'a' || symb == 'A')

{

countS1++;

}

}

if (countS2 < countS1)

{

countS2 = countS1;

numb = count;

}

}

* 1. int N1, N2;
* Программа будет запрашивать у пользователя значение двух этих переменных в функции FillN().

cout << "Между какими строчками хотите искать совпадения?" << endl;

cout << "Введите номер первой строки: ";

cin >> \*N1;

while (\*N1 < 1 || \*N1 >= count)

{

cout << "Введен некорректный номер!" << endl;

cout << "Введите номер первой строки: ";

cin >> \*N1;

}

cout << "Введите номер второй строки: ";

cin >> \*N2;

while (\*N2 < 1 || \*N2 > count || ((\*N2 - \*N1) <= 1))

{

if ((\*N2 - \*N1) == 1)

{

cout << "Между данными строками нет строк!" << endl;

\*f = 1;

return \*f;

}

cout << "Введен некорректный номер!" << endl;

cout << "Введите номер второй строки: ";

cin >> \*N2;

}

* Полученные значения N1 и N2 передаются в функцию int countSTR\_F1() в данной функции эти переменные будут выступать в качестве условия при соблюдении которых будет вызываться функция cpy\_F2().

while (!fin.eof())

{

str = "";

getline(fin, str);

count++;

if (count > N1 && count < N2)

{

cpy\_F2(str, path\_F2);

}

}

* 1. int countS1, countS2;
* Данные переменные отвечают за запоминание количества символов встретившихся в строках из файла F2 ‘a’ и ‘A’.

while (!fin.eof())

{

count++;

countS1 = 0;

str = "";

getline(fin, str);

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

{

symb = str[i];

if (symb == 'a' || symb == 'A')

{

countS1++;

}

}

if (countS2 < countS1)

{

countS2 = countS1;

numb = count;

}

}

* 1. int DataF(int count, string path\_F1)
* Функция, которая будет запрашивать у пользователя количество строк в файле F1. Количество строк при этом не должно быть меньше 3, так как в противном случае между строками с номерами N1 и N2 не найдется ни одной строки.

int DataF(int count, string path\_F1)

{

cout << "Введите количество строк находящихся в файле " << path\_F1 << " :\t";

cin >> count;

while (count < 3)

{

cout << "Минимальное количество строк в файле должно быть не меньше 3!" << endl;

cout << "Введите количество строк находящихся в файле " << path\_F1 << " :\t";

cin >> count;

}

return count;

}

* 1. int cpy\_F2(string str, string path\_F2)
* Данная функция принимает строку str в качестве параметра открывает файл F2 для записи, находит длину строки str, далее проходится во всей строке str, находит и считает количество встретившихся символов ‘a’ и ‘A’.

int cpy\_F2(string str, string path\_F2)

{

ofstream fout;

fout.open(path\_F2, ofstream::app);

if (!fout.is\_open())

{

cout << "Ошибка открытия файла!" << endl;

return 0;

}

else

{

int size;

size = str.length();

cout << "Файл " << path\_F2 << " был успешно открыт!\n";

if (str[size - 1] == 'a' || str[size - 1] == 'A')

{

fout << str << '\n';

}

}

fout.close();

return 0;

}

* 1. int FillN(int \*N1, int \*N2, int count, bool \*f)
* Данная функция запрашивает у пользователя между какими строками в файле F1, алгоритм будет искать строки, которые оканчиваются на символы ‘a’ и ‘A’.

int FillN(int \*N1, int \*N2, int count, bool \*f)

{

cout << "Между какими строчками хотите искать совпадения?" << endl;

cout << "Введите номер первой строки: ";

cin >> \*N1;

while (\*N1 < 1 || \*N1 >= count)

{

cout << "Введен некорректный номер!" << endl;

cout << "Введите номер первой строки: ";

cin >> \*N1;

}

cout << "Введите номер второй строки: ";

cin >> \*N2;

while (\*N2 < 1 || \*N2 > count || ((\*N2 - \*N1) <= 1))

{

if ((\*N2 - \*N1) == 1)

{

cout << "Между данными строками нет строк!" << endl;

\*f = 1;

return \*f;

}

cout << "Введен некорректный номер!" << endl;

cout << "Введите номер второй строки: ";

cin >> \*N2;

}

return 0;

}

* 1. int countSTR\_F1(string path\_F1, string path\_F2, bool f, int count, int N1, int N2)
* Данная функция открывает файл F1 для чтения, проверяет на то открылся ли файл, если открылся, то проходится по каждой строке, записывает каждую строку по очереди в переменную str и если номер записанной строки удовлетворяет условию (count > N1 && count < N2) , то вызывается функция cpy\_F2(str, path\_F2), и туда передается текущая записанная строка. В конце данной программы файловый поток закрывается.

int countSTR\_F1(string path\_F1, string path\_F2, bool f, int count, int N1, int N2)

{

string str;

ifstream fin;

cout << "Количество строк = " << count - 1 << endl;

count = 0;

fin.open(path\_F1);

if (!fin.is\_open())

{

cout << "Ошибка открытия файла!" << endl;

f = 1;

return 0;

}

else

{

cout << "Файл " << path\_F1 << " открыт для чтения!" << endl;

while (!fin.eof())

{

str = "";

getline(fin, str);

count++;

if (count > N1 && count < N2)

{

cpy\_F2(str, path\_F2);

}

}

}

fin.close();

return 0;

}

* 1. int STR\_F2(string path\_F2, bool f, int count)
* Данная функция открывает файл F2 для чтения, Проверяет открылся ли файл, и в случае если файл открылся, то функция проходится по всем строкам находящихся в файле F2. И в каждой строке подсчитывает количество встретившихся символов ‘a’ и ‘A’. И Записывает количество ‘a’ и ‘A’ в текущей строке в переменную countS1, а количество этих символов в предыдущей строке в переменную countS2. Алгоритм ищет строку с наибольшим количеством этих символов, запоминает номер этой строки в переменную numb. В конце данной функции закрывает файловый поток и выводит на консоль номер строки с наибольшим количеством символов ‘a’ и ‘A’ и соответственно из количество.

int STR\_F2(string path\_F2, bool f, int count)

{

count = 0;

int countS1;

int countS2 = 0;

char symb;

int numb;

ifstream fin;

fin.open(path\_F2);

if (!fin.is\_open())

{

cout << "Ошибка открытия файла!" << endl;

f = 1;

return 0;

}

else

{

cout << "Файл " << path\_F2 << " открыт для чтения!" << endl;

string str;

while (!fin.eof())

{

count++;

countS1 = 0;

str = "";

getline(fin, str);

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

{

symb = str[i];

if (symb == 'a' || symb == 'A')

{

countS1++;

}

}

if (countS2 < countS1)

{

countS2 = countS1;

numb = count;

}

}

}

fin.close();

cout << "Строка под номером " << numb << " в файле " << path\_F2;

cout << " с наибольшим количеством символов 'А' и 'а' ";

cout << "а именно " << countS2 << endl;

return 0;

}

* 1. void fill\_F1(string path\_F1, bool \*f, int count)
* Данная функция открывает файл F1 для записи, проверяет открылся ли данный файл и в случае если данный файл открыт, то функция просит пользователя заполнить строки, которые будут записаны в файл F1.

void fill\_F1(string path\_F1, bool \*f, int count)

{

ofstream fout;

fout.open(path\_F1);

if (!fout.is\_open())

{

cout << "Ошибка открытия файла!" << endl;

\*f = 1;

}

else

{

string str;

cout << "Файл " << path\_F1 << " был успешно открыт!\n";

for (int i = 1; i <= count; i++) {

cout << "Введите " << i << " строку в файле " << path\_F1 << endl;

cin.ignore();

getline(cin, str);

fout << str << '\n';

}

}

fout.close();

}

* 1. int main()
* В функции main, сначала функция удаляет файлы F1 и F2 для корректной работы программы с помощью функции remove(). Если в корне с проектом данные файлы не будут найдены, ничего не произойдет и алгоритм продолжит свою работу.

remove("F1.txt");

remove("F2.txt");

* Далее данная функция добавляет русский язык. Создает переменные содержащие названия файлов, для того чтобы потом передавать не полные названия файлов, а имена переменной отвечающей за имена данных файлов.

setlocale(LC\_ALL, "Ru");

string path\_F1 = "F1.txt";

string path\_F2 = "F2.txt";

* Создается переменная count, переменная N1 и N2. Переменной count присваивается возвращаемое значение функции DataF(count, path\_F1), создается флаг f, вызывается функция FillN(), действия которой описаны в пункте 2.6. Далее проверяется изменилось ли значение флага и в случае если f = 1, то программа завершает свою работу. Далее вызывается функция fill\_F1(path\_F1, &f, count), действие которой описаны в пункте 2.9. Снова проверяется значение флага. Далее вызывается функция countSTR\_F1(path\_F1, path\_F2, f, count, N1, N2); Действие которой описано в пункте 2.7. Вызывается функция STR\_F2(path\_F2, f, count), действие которой описано в пункте 2.8.

int main()

{

remove("F1.txt");

remove("F2.txt");

setlocale(LC\_ALL, "Ru");

string path\_F1 = "F1.txt";

string path\_F2 = "F2.txt";

int count = 0;

int N1;

int N2;

count = DataF(count, path\_F1);

bool f = 0;

FillN(&N1, &N2, count, &f);

if (f == true)

{

return 0;

}

fill\_F1(path\_F1, &f, count);

if (f == true) {

return 0;

}

countSTR\_F1(path\_F1, path\_F2, &f, count, N1, N2);

if (f == true) {

cout << "Ни в одной строке не встретилось ни одного символа Aa";

return 0;

}

STR\_F2(path\_F2, &f, count);

if (f == true) {

cout << "Ни в одной строке не встретилось ни одного символа Aa";

return 0;

}

return 0;

}

1. Какими операторами реализован ввод и вывод.
   1. Ввод:

* С помощью оператора cin, например:

cin >> count;

* С помощью функции getline():

getline(cin, str);

* 1. Вывод:
* С помощью оператора cout, например:

cout << "Ошибка открытия файла!" << endl;

1. Какими действиями реализованы поставленные задачи:
   1. Цикл for:

* Используется для перемещения между строками в файле F1.

for (int i = 1; i <= count; i++) {

cout << "Введите " << i << " строку в файле " << path\_F1 << endl;

cin.ignore();

getline(cin, str);

fout << str << '\n';

}

* Используется для передвижения по всем символам строки str.

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

{

symb = str[i];

if (symb == 'a' || symb == 'A')

{

countS1++;

}

* 1. Цикл while:
* Используется для наложения ограничения на ввод количества строк в файле F1.

while (count < 3)

{

cout << "Минимальное количество строк в файле должно быть не меньше 3!" << endl;

cout << "Введите количество строк находящихся в файле " << path\_F1 << " :\t";

cin >> count;

}

* Используется для наложения ограничения на ввод значений N1 и N2.

while (\*N1 < 1 || \*N1 >= count)

{

cout << "Введен некорректный номер!" << endl;

cout << "Введите номер первой строки: ";

cin >> \*N1;

}

while (\*N2 < 1 || \*N2 > count || ((\*N2 - \*N1) <= 1))

{

if ((\*N2 - \*N1) == 1)

{

cout << "Между данными строками нет строк!" << endl;

\*f = 1;

return \*f;

}

* Используется для прохождения по всем строкам файла F1, до того, пока цикл не достигнет конца файла.

while (!fin.eof())

{

str = "";

getline(fin, str);

count++;

if (count > N1 && count < N2)

{

cpy\_F2(str, path\_F2);

}

}

* Используется для прохождения по всем строкам файла F2, до того, пока цикл не достигнет конца файла.

while (!fin.eof())

{

count++;

countS1 = 0;

str = "";

getline(fin, str);

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

{

symb = str[i];

if (symb == 'a' || symb == 'A')

{

countS1++;

}

}

if (countS2 < countS1)

{

countS2 = countS1;

numb = count;

}

}

**Полный код**

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

int DataF(int count, string path\_F1)

{

cout << "Введите количество строк находящихся в файле " << path\_F1 << " :\t";

cin >> count;

while (count < 3)

{

cout << "Минимальное количество строк в файле должно быть не меньше 3!" << endl;

cout << "Введите количество строк находящихся в файле " << path\_F1 << " :\t";

cin >> count;

}

return count;

}

void fill\_F1(string path\_F1, bool \*f, int count)

{

ofstream fout;

fout.open(path\_F1);

if (!fout.is\_open())

{

cout << "Ошибка открытия файла!" << endl;

\*f = true;

}

else

{

string str;

cout << "Файл " << path\_F1 << " был успешно открыт!\n";

for (int i = 1; i <= count; i++) {

cout << "Введите " << i << " строку в файле " << path\_F1 << endl;

if (i == 1) {

cin.ignore();

}

getline(cin, str);

fout << str << '\n';

}

}

fout.close();

}

int cpy\_F2(string str, string path\_F2)

{

ofstream fout;

fout.open(path\_F2, ofstream::app);

if (!fout.is\_open())

{

cout << "Ошибка открытия файла!" << endl;

return 0;

}

else

{

int size;

size = str.length();

if (str[size - 1] == 'a' || str[size - 1] == 'A')

{

fout << str << '\n';

}

}

fout.close();

return 0;

}

int FillN(int \*N1, int \*N2, int count, bool \*f)

{

cout << "Между какими строчками хотите искать совпадения?" << endl;

cout << "Введите номер первой строки: ";

cin >> \*N1;

while (\*N1 < 1 || \*N1 >= count)

{

cout << "Введен некорректный номер!" << endl;

cout << "Введите номер первой строки: ";

cin >> \*N1;

}

cout << "Введите номер второй строки: ";

cin >> \*N2;

while (\*N2 < 1 || \*N2 > count || ((\*N2 - \*N1) <= 1))

{

if ((\*N2 - \*N1) == 1)

{

cout << "Между данными строками нет строк!" << endl;

\*f = true;

return \*f;

}

cout << "Введен некорректный номер!" << endl;

cout << "Введите номер второй строки: ";

cin >> \*N2;

}

return 0;

}

int countSTR\_F1(string path\_F1, string path\_F2, bool \*f, int count, int N1, int N2)

{

string str;

ifstream fin;

count = 0;

fin.open(path\_F1);

if (!fin.is\_open())

{

cout << "Ошибка открытия файла!" << endl;

\*f = true;

return 0;

}

else

{

cout << "Файл " << path\_F1 << " открыт для чтения!" << endl;

while (!fin.eof())

{

str = "";

getline(fin, str);

count++;

if (count > N1 && count < N2)

{

cpy\_F2(str, path\_F2);

}

}

}

fin.close();

return 0;

}

int STR\_F2(string path\_F2, bool \*f, int count)

{

count = 0;

int countS1;

int countS2 = 0;

char symb;

int numb = 0;

ifstream fin;

fin.open(path\_F2);

if (!fin.is\_open())

{

cout << "Ошибка открытия файла!" << endl;

\*f = true;

return 0;

}

else

{

cout << "Файл " << path\_F2 << " открыт для чтения!" << endl;

string str;

while (!fin.eof())

{

count++;

countS1 = 0;

str = "";

getline(fin, str);

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

{

symb = str[i];

if (symb == 'a' || symb == 'A')

{

countS1++;

}

}

if (countS2 < countS1)

{

countS2 = countS1;

numb = count;

}

}

}

fin.close();

if (numb == 0) {

\*f = true;

return 0;

}

cout << "Строка под номером " << numb << " в файле " << path\_F2;

cout << " с наибольшим количеством символов 'А' и 'а' ";

cout << "а именно " << countS2 << endl;

return 0;

}

int main()

{

remove("F1.txt");

remove("F2.txt");

setlocale(LC\_ALL, "Ru");

string path\_F1 = "F1.txt";

string path\_F2 = "F2.txt";

int count = 0;

int N1;

int N2;

count = DataF(count, path\_F1);

bool f = 0;

FillN(&N1, &N2, count, &f);

if (f == true)

{

return 0;

}

fill\_F1(path\_F1, &f, count);

if (f == true) {

return 0;

}

countSTR\_F1(path\_F1, path\_F2, &f, count, N1, N2);

if (f == true) {

cout << "Ни в одной строке не встретилось ни одного символа Aa";

return 0;

}

STR\_F2(path\_F2, &f, count);

if (f == true) {

cout << "Ни в одной строке не встретилось ни одного символа Aa";

return 0;

}

return 0;

}

**Блок-Схема**











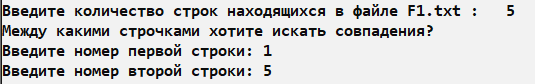


**Скриншоты результатов работы программы**

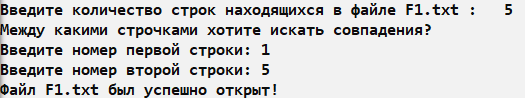
1. **Нормальный ввод:**
   1. Программа запрашивает ввод от пользователя количества строк, которые будут находиться в файле F1(В примере вводится 5).



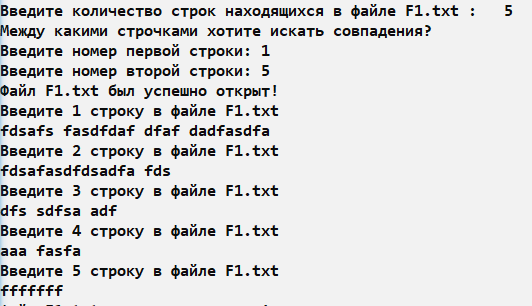
* 1. Программа спрашивает у пользователя между какими строчками мы хотим искать строчки которые заканчиваются на ‘a’ и ‘A’. ( На примере ввод 1 и 5, то есть алгоритм будет искать среди строчек 2-4).



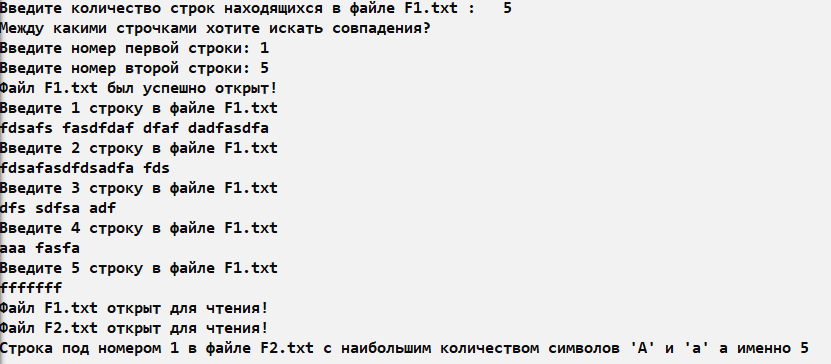
* 1. В случае если программе удалось открыть файл F1, то программа выводит сообщение об этом.(В случае если такого файла нет, а его нет, так как программа удаляет в начале программы файл F1 и F2, и так как не указан путь этих файлов, то алгоритм создает оба этих файла в папке с проектом!).



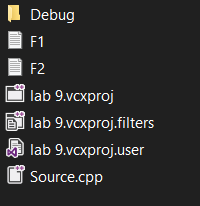
* 1. Программа просит ввести строки, которые будут переданы в файл F1.(В примере программа сделает 5 запросов ввода)



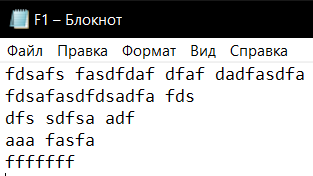
* 1. Программа ищет между строками N1 и N2(не включая их), указанными пользователем в файле F1, те строки, которые заканчиваются на символы ‘a’ и ‘A’, копирует их и записывает в файл F2. А затем, если в файле F2 есть строчки, он ищет среди них строку с наибольшим количеством этих символов, и выводит номер этой строки и количество этих символов в строке.



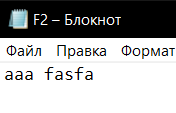
* 1. В результате программа создала 2 файла в папке с проектом.



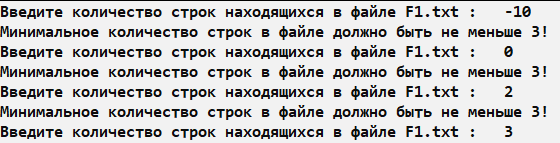
* 1. Открыв файл F1, можно увидеть, что там находятся введенные нами 5 строк.



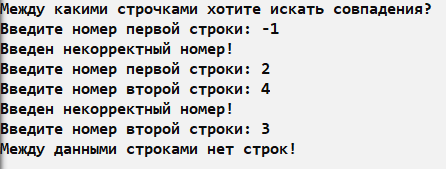
* 1. Открыв файл F2, мы увидим, что программа скопировала те строки, которые удовлетворяют нашим условиям.



1. Неправильный ввод:
   1. Пробуем ввести количество строк, которое меньше 3. Программа не дает этого сделать, так как в противном случае между строчками N1 и N2 не окажется строчек. (В примере программа приняла число 3)



* 1. Далее попробуем ввести отрицательное число в первый номер строки отрицательное число. Программа не даст нам этого сделать. Тогда введем 2, программа примет данный ввод. Далее программа запросит ввод номер второй строки, пробуем ввести число, которое больше количества строк. Программа не даст нам этого сделать, вводим 3. Программа завершит свою работу с сообщением, что между строчками 2 и 3 строчек нет.



* 1. Заново запускаем программу, доходим до предыдущего пункта, но вводим уже 1 и 3, теперь у нас есть 1 строка которая находится между, что если в ней не окажется букв ‘a’ и ‘A’. Программа завершит свое действие, с сообщением, что таких символов в строчках не встретилось.

