Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа № 3

По дисциплине «Основы алгоритмизации и программирования»

На тему «Работа с файлами на языке С++»

Выполнил:

Студент 1 курса 8 группы

Статько Герман Вячеславович

Преподаватель: асс. Андронова М.В.

Минск, 2024

В соответствии со своим вариантом разработать программы для работы с файлами на языке С++.

Для первой программы необходимо предварительно создать текстовый файл FILE1 из нескольких строк и записать в него данные.

Во второй программе ввод информации с клавиатуры и вывод в консольное окно осуществить в главной функции, а запись в файл и чтение из файла - в функциях пользователя.

**Вариант 12**

****

#include <iostream>

#include <fstream>

#include <sstream>

using namespace std;

int main()

{

setlocale(LC\_CTYPE, "rus");

ifstream fin("FILE1.txt"); // Открываем файл FILE1.txt для чтения

string firstWord, secondWord, line; // Объявляем переменные для хранения слов

if (!fin.is\_open())

{

cout << "Файл не может быть открыт!\n"; // Выводим сообщение об ошибке

}

ofstream fout("FILE2.txt"); // Открываем файл FILE2.txt для записи

if (!fout.is\_open())

{

cout << "Файл не может быть открыт!\n"; // Выводим сообщение об ошибке

}

int count = 0; // Инициализируем переменную для подсчета количества букв в совпадающих словах

while (fin >> firstWord && fin >> secondWord) // Читаем по два слова из файла

{

if (firstWord == secondWord) // Если слова совпадают

{

fout << firstWord << " " << secondWord << "\n"; // Записываем оба слова в файл FILE2.txt

count += size(firstWord) + size(secondWord); // Увеличиваем счетчик на длину обоих слов

}

}

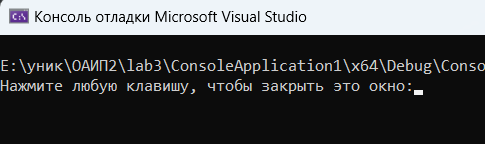
fout << "Количество букв: " << count; // Записываем общее количество букв в совпадающих словах в файл

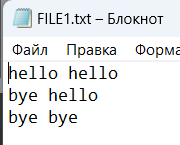
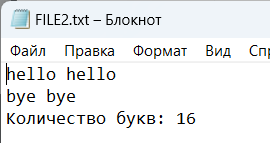
fin.close(); // Закрываем файл FILE1.txt

fout.close(); // Закрываем файл FILE2.txt

return 0;

}

****

**** ****



#include <iostream>

#include <fstream>

#include <stdio.h>

#include <cstdlib>

using namespace std;

void fout();

void scan(char\*);

int main()

{

setlocale(LC\_CTYPE, "rus");

fout();

ifstream fin("File.txt"); // Открываем файл для чтения

char line[50];

fin.getline(line, 50); // Считываем строку из файла

scan(line);

fin.close(); // Закрываем файл

return 0;

}

void fout()

{

ofstream fout("File.txt"); // Открываем файл для записи

if (!fout.is\_open())

{

cout << "Файл не может быть открыт! " << endl; // Выводим сообщение об ошибке, если файл не открылся

}

char buffer[50];

cout << "Введите строку символов, состоящую из цифр и слов, разделенных пробелами: ";

cin.getline(buffer, 50); // Считываем строку с клавиатуры

fout << buffer; // Записываем строку в файл

fout.close(); // Закрываем файл

}

void scan(char \*line)

{

int num;

for (int i = 0; i < 50; i++)

{

if (line[i] >= '0' && line[i] <= '9') // Проверяем, является ли символ цифрой

{

num = line[i] - '0'; // Преобразуем символ числа в само число

if (num % 2 != 0) // Проверяем, является ли число нечетным

{

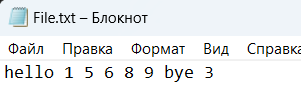
cout << num << " "; // Выводим нечетное число

}

}

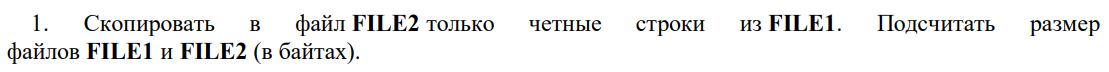
}

}****

****

**Доп. задания**

**Вариант 1**

****

#include <iostream>

#include <fstream>

#include <stdio.h>

using namespace std;

int main()

{

setlocale(LC\_CTYPE, "rus");

ifstream fin("FILE1.txt"); // Открытие файла FILE1.txt для чтения

if (!fin.is\_open()) // Проверка успешного открытия файла

{

cout << "Файл не может быть открыт!\n"; // Вывод сообщения об ошибке

return -1;

}

ofstream fout("FILE2.txt"); // Открытие файла FILE2.txt для записи

if (!fout.is\_open()) // Проверка успешного открытия файла

{

cout << "Файл не может быть открыт!\n"; // Вывод сообщения об ошибке

return -1;

}

char buff[50]; // Объявление массива символов для временного хранения строки

int count = 0; // Инициализация счетчика строк

while (fin.getline(buff, 50)) // Чтение строк из файла в цикле

{

count++; // Увеличение счетчика строк

if (count % 2 == 0) // Проверка на четность номера строки

{

fout << buff << "\n"; // Запись строки в файл FILE2.txt

}

}

fin.close(); // Закрытие файла FILE1.txt

fout.close(); // Закрытие файла FILE2.txt

FILE\* f1, \* f2;

int f1Size, f2Size;

fopen\_s(&f1, "FILE1.txt", "rb"); // Открытие файла FILE1.txt для определения размера

if (f1 == NULL) // Проверка успешного открытия файла

{

cout << "Ошибка при попытке подсчета размера файла FILE1"; // Вывод сообщения об ошибке

return -1;

}

fseek(f1, 0, SEEK\_END); // Перемещение указателя в конец файла

f1Size = ftell(f1); // Получение размера файла FILE1.txt

cout << "Размер FILE1 = " << f1Size << " байт" << endl; // Вывод размера файла FILE1.txt

fclose(f1); // Закрытие файла FILE1.txt

fopen\_s(&f2, "FILE2.txt", "rb"); // Открытие файла FILE2.txt для определения размера

if (f2 == NULL) // Проверка успешного открытия файла

{

cout << "Ошибка при попытке подсчета размера файла FILE2"; // Вывод сообщения об ошибке

return -1;

}

fseek(f2, 0, SEEK\_END); // Перемещение указателя в конец файла

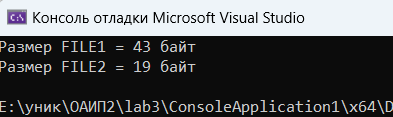
f2Size = ftell(f2); // Получение размера файла FILE2.txt

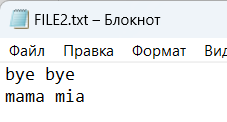
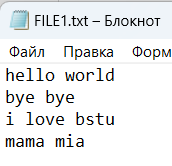
cout << "Размер FILE2 = " << f2Size << " байт" << endl; // Вывод размера файла FILE2.txt

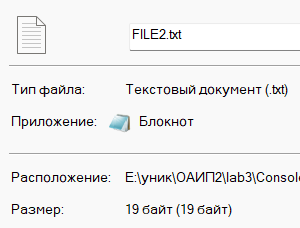
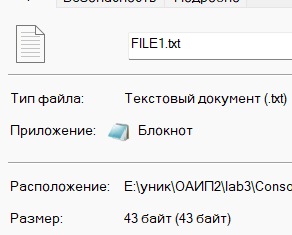
fclose(f2); // Закрытие файла FILE2.txt

return 0;

}

****

****

****

****

#include <iostream>

#include <fstream>

#include <stdio.h>

#include <cstdlib>

using namespace std;

void fout();

void scan(char\*);

int main()

{

setlocale(LC\_CTYPE, "rus");

fout();

ifstream fin("File.txt"); // Открываем файл для чтения

char line[50];

fin.getline(line, 50); // Считываем строку из файла

scan(line);

fin.close(); // Закрываем файл

return 0;

}

void fout()

{

ofstream fout("File.txt"); // Открываем файл для записи

if (!fout.is\_open())

{

cout << "Файл не может быть открыт! " << endl; // Выводим сообщение об ошибке, если файл не открылся

}

char buffer[50];

cout << "Введите строку символов, состоящую из цифр и слов, разделенных пробелами: ";

cin.getline(buffer, 50); // Считываем строку с клавиатуры

fout << buffer; // Записываем строку в файл

fout.close(); // Закрываем файл

}

void scan(char \*line)

{

int num;

for (int i = 0; i < 50; i++)

{

if (line[i] >= '0' && line[i] <= '9') // Проверяем, является ли символ цифрой

{

num = line[i] - '0'; // Преобразуем символ числа в само число

if (num % 2 != 0) // Проверяем, является ли число нечетным

{

cout << num << " "; // Выводим нечетное число

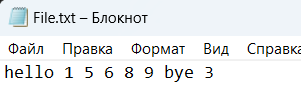
}

}

}

}

****

****

**Вариант 2**

****

#include <iostream>

#include <fstream>

#include <stdio.h>

using namespace std;

int main()

{

setlocale(LC\_CTYPE, "rus");

// Открываем файл FILE1.txt для чтения

ifstream fin("FILE1.txt");

// Проверяем успешность открытия файла

if (!fin.is\_open())

{

cout << "Ошибка открытия файла FILE1!\n";

return -1;

}

// Открываем файл FILE2.txt для записи

ofstream fout("FILE2.txt");

// Проверяем успешность открытия файла

if (!fout.is\_open())

{

cout << "Ошибка открытия файла FILE2!\n";

return -1;

}

char buffer[50];

int count = 1;

// Считываем строки из FILE1.txt и обрабатываем их

while (fin.getline(buffer, 50))

{

// Если первая буква строки - 'A', записываем строку в FILE2.txt

if (buffer[0] == 'A')

{

fout << buffer << "\n";

}

// Подсчитываем количество слов в строке

for (int k = 0; k < sizeof(buffer); k++)

{

if (buffer[k] == ' ')

{

count++;

}

}

}

// Выводим количество слов в FILE2.txt

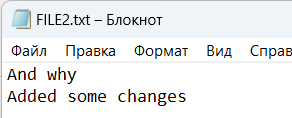
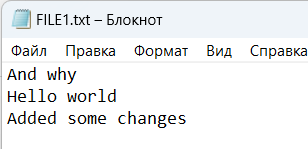
cout << "Количество слов в FILE2: " << count << endl;

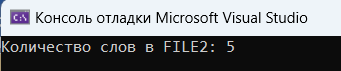
// Закрываем файлы

fin.close();

fout.close();

}

****

****

****

#include <iostream>

#include <fstream>

#include <climits>

using namespace std;

// Функция для поиска самой короткой группы

void findShortestGroup(const char\* groups) {

int start = 0;

int end = 0;

int minGroupStart = 0;

int minGroupEnd = INT\_MAX;

// Проходим по строке и ищем группы

for (int i = 0; i <= strlen(groups); i++) {

if (groups[i] == ' ' || groups[i] == '\0') {

end = i;

// Сравниваем длины текущей группы с минимальной длиной

if (end - start < minGroupEnd - minGroupStart) {

minGroupStart = start;

minGroupEnd = end;

}

start = i + 1; // Переходим к началу следующей группы

}

}

// Выводим самую короткую группу

for (int i = minGroupStart; i < minGroupEnd; i++) {

cout << groups[i];

}

}

int main() {

setlocale(LC\_CTYPE, "rus");

ofstream fout("file.txt");

// Проверяем успешное открытие файла для записи

if (!fout.is\_open()) {

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

return -1;

}

char buffer[50];

cout << "Введите строку символов, состоящую из групп цифр и нулей: ";

cin.getline(buffer, 50);

fout << buffer; // Записываем введенную строку в файл

fout.close(); // Закрываем файл после записи

ifstream fin("file.txt");

// Проверяем успешное открытие файла для чтения

if (!fin.is\_open()) {

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

return -1;

}

char groups[50];

fin.getline(groups, 50); // Читаем данные из файла

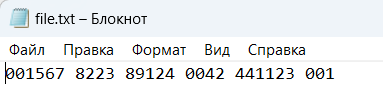
// Вызываем дополнительную функцию для поиска и вывода самой короткой группы

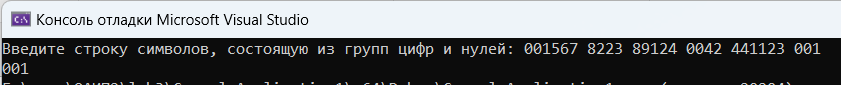
findShortestGroup(groups);

fin.close(); // Закрываем файл после чтения

return 0;

}

****

****

**Вариант 13**

****

#include <iostream>

#include <fstream>

#include <stdio.h>

using namespace std;

int main()

{

setlocale(LC\_CTYPE, "rus");

ifstream fin("FILE1.txt"); // Открытие файла FILE1 для чтения

char word[20], buffer[50];

if (!fin.is\_open())

{

cout << "Ошибка открытия файла FILE1!\n"; // Вывод сообщения об ошибке открытия файла FILE1

return -1;

}

ofstream fout("FILE2.txt"); // Открытие файла FILE2 для записи

if (!fout.is\_open())

{

cout << "Ошибка открытия файла FILE2!\n"; // Вывод сообщения об ошибке открытия файла FILE2

return -1;

}

int count = 0, maxcount = 0, wordcount = 1, maxglcount;

while (fin.getline(buffer, 50)) // Чтение строки из файла

{

for (int i = 0; i < sizeof(buffer); i++) // Перебор символов в строке

{

if (buffer[i] == ' ') // Если символ - пробел, сбросить счетчик гласных и увеличить счетчик слов

{

count = 0;

wordcount++;

}

if (buffer[i] == 'A' || buffer[i] == 'a' || buffer[i] == 'E' || buffer[i] == 'e' || buffer[i] == 'I' || buffer[i] == 'i' || buffer[i] == 'O' || buffer[i] == 'o' || buffer[i] == 'U' || buffer[i] == 'u' || buffer[i] == 'Y' || buffer[i] == 'y') // Проверка на гласные буквы

{

count++;

if (count > maxcount)

{

maxcount = count;

maxglcount = wordcount;

}

}

}

if (wordcount > 2) // Если номер слова больше 2, записать строку в файл FILE2

{

fout << buffer << "\n";

}

cout << "Номер слова с максимальным количеством гласных в строке: " << maxglcount << endl; // Вывод номера слова с максимальным количеством гласных в строке

count = 0;

maxcount = 0;

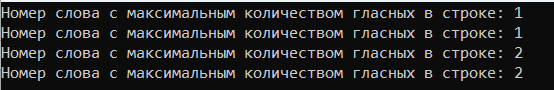
wordcount = 1;

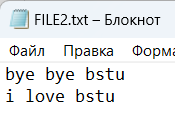
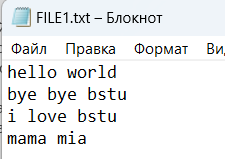
}

fin.close(); // Закрытие файла FILE1

fout.close(); // Закрытие файла FILE2

}

****

****

****

#include <iostream>

#include <fstream>

#include <conio.h>

using namespace std;

int findSkobki(char\* buffer);

int main()

{

setlocale(LC\_CTYPE, "rus");

ofstream fout("file.txt"); // Открытие файла для записи

char buffer[50]; // Объявление буфера для ввода строки

cout << "Введите строку символов, состоящую из цифр и скобок: ";

cin.getline(buffer, 100); // Ввод строки с клавиатуры

fout << buffer; // Запись строки в файл

cout << "Количество скобок разного вида: " << findSkobki(buffer) << "\n"; // Вывод количества скобок разного вида

fout.close(); // Закрытие файла

}

int findSkobki(char\* buffer)

{

bool fstfound = false, sndfound = false, trdfound = false; // Флаги для обозначения найденных скобок

int count = 0; // Счетчик количества скобок

for (int i = 0; i < 50; i++) // Цикл по символам в буфере

{

if (buffer[i] == '(' || buffer[i] == ')' || buffer[i] == '{' || buffer[i] == '}' || buffer[i] == '[' || buffer[i] == ']') // Проверка на скобки

{

if (!fstfound && (buffer[i] == '(' || buffer[i] == ')')) // Проверка на скобки типа ()

{

count++; // Увеличение счетчика

fstfound = true; // Установка флага

}

if (!sndfound && (buffer[i] == '{' || buffer[i] == '}')) // Проверка на скобки типа {}

{

count++; // Увеличение счетчика

sndfound = true; // Установка флага

}

if (!trdfound && (buffer[i] == '[' || buffer[i] == ']')) // Проверка на скобки типа []

{

count++; // Увеличение счетчика

trdfound = true; // Установка флага

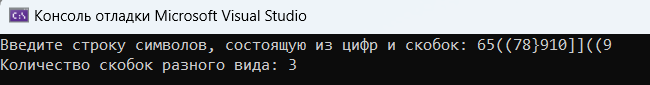
}

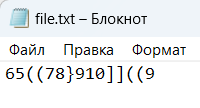
}

}

return count; // Возвращаем количество найденных скобок

}

****

****