Лабораторная работа 3

По дисциплине «Основы алгоритмизации и программирования»

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

***Задание #1***



Macros.h

#include <iostream>

#include <vector>

#include <ctime>

#include <cmath>

#include <iomanip>

#include <stack>

#include <map>

#include <string>

#include <chrono>

#include <fstream>

#include <Windows.h>

#define vi cout <<

#define vv cin >>

#define nl cout << "\n";

#define nw cout << "\t";

#define SCOCP1251 SetConsoleOutputCP(1251);

#define SCCP1251 SetConsoleCP(1251);

#define ret return

#define pause system("pause");

using namespace :: std;

.cpp

#include "Macros.h"

// проверка на наличие цифр

bool StringContainsDigits(const string& str);

// обработка содержимого файла и записи в другой

void ReadAndProcessFile(const string& fileName, const string& outputFileName);

// подсчет строк, начинающихся с символа с

int CountStringInFileStartsWith(const string& fileName, const char& c);

int main() {

SCOCP1251;

ReadAndProcessFile("FILE1.txt", "filee22.txt"); // обработка "FILE1.txt"

vi CountStringInFileStartsWith("filee22.txt", 'A'); // подсчет строк начинающихся с А

ret 0;

}

bool StringContainsDigits(const string& str) {

for (const char& c : str) { // итератор для прохода по символам строки

if ('0' <= c && c <= '9') { // если цифра

return true;

}

}

ret false;

}

void ReadAndProcessFile(const string& inputFileName, const string& outputFileName) {

ifstream inputFile(inputFileName);

ofstream outputFile(outputFileName);

// проверка на открытие

if (!inputFile.is\_open() || inputFile.fail()) {

cerr << "Ошибка открытия " + inputFileName + " для чтения"; nl

ret;

}

// проверка на открытие

if (!outputFile.is\_open() || outputFile.fail()) {

cerr << "Ошибка открытия " + outputFileName + " для записи"; nl

return;

}

string line; // строка

while (getline(inputFile, line)) { // получение строки файла

if (!StringContainsDigits(line)) { // нет цифр?

outputFile << line + '\n'; // перезапись в другой файл

}

}

inputFile.close();

outputFile.close();

}

int CountStringInFileStartsWith(const string& fileName, const char& c) {

ifstream file(fileName);

if (!file.is\_open() || file.fail()) {

cerr << "ошибка открытия " + fileName + " для чтения"; nl

ret -1;

}

string line;

int count = 0;

while (getline(file, line)) {

if (line[0] == c) { // проверка при помощи индекса на равность указанному

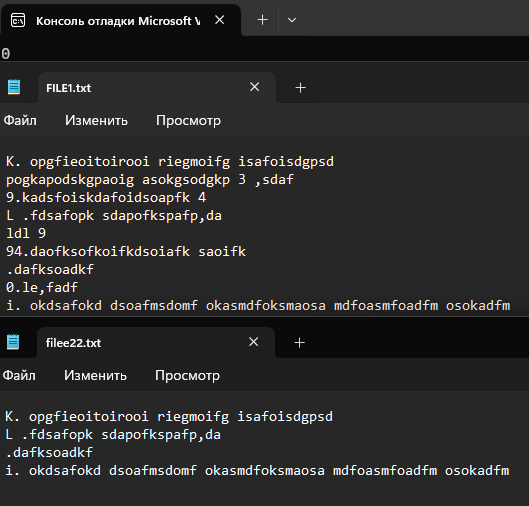
count++;

}

}

ret count;

}



***Задание #2***



.cpp

#include "Macros.h"

// функция записи строки в файл

void WriteStringIntoFile(const string& outputFile, string str);

// функция чтения и нахождения минимальной длины

string ReadAndProcess(const string& inputFile);

bool containsDigits(string& str);

int main() {

SCOCP1251

string fileName = "myfile.txt";

string input;

// ввод строки

getline(cin, input);

// запись

WriteStringIntoFile(fileName, input);

// нахождение

string x = ReadAndProcess(fileName);

if (x == "") { // ошибка или не было слов

ret -1;

}

vi x + "\n" << x.length(); nl

ret 0;

pause

}

void WriteStringIntoFile(const string& outputFile, string str) {

ofstream file(outputFile);

if (!file.is\_open() || file.fail()) {

cerr << "Файл записи не существует";

ret;

}

file << str;

file.close();

}

string ReadAndProcess(const string& inputFile) {

ifstream file(inputFile);

if (!file.is\_open() || file.fail()) {

cerr << "Файл чтения не существует";

ret ""; // возврат "", для ошибки

}

string str; // строка для группы символов

string maxWord = ""; // наидлиннейшее слово

int maxLength = 0;

while (file >> str) { // считывание групп разделенных пробелами

if (!containsDigits(str) && maxLength < str.length()) { // нет цифр - слово

maxLength = str.length(); // сохранение длины длиннейшего слова

maxWord = str; // сохранение длиннейших слов

}

}

file.close();

ret maxWord; // возврат длиннейшего

}

// функция проверки на наличие цифр

bool containsDigits(string& str) {

for (const char& c : str) { // итератор по строке

if ('0' <= c && c <= '9') { // цифра - возврат true

ret true;

}

}

ret false;

}

