1. Лабораторная работа № 5

«Программирование операций над строками и файлами»

* 1. Цель работы

Изучение основных операций над строками и файлами, программирование операций обработки строк текстовых файлов, исследование свойств файловых указателей.

## Постановка задачи

Приобрести навыки работы с файлами с помощью средств языков программирования C/C++.

* 1. Ход выполнения работы

Вариант выполнения задания – 18.

* + 1. Описание алгоритма решения задачи
       1. Определение входных и выходных данных

На вход поступают 2 строки. Первая – путь к файлу, с которого будут считываться предложения. Вторая – путь к файлу, в который будут записываться предложения в обратном порядке.

* + - 1. Структурная схема алгоритма

Структурные схемы реализации алгоритма представлены в Приложении А. На Рисунке А.1 представлена схема алгоритма посимвольного считывания предложений из входного файла. На Рисунках А.2 и А.3 представлены структурные схемы построчного считывания предложений из файла, каждое предложение хранится в оперативной памяти до завершения выполнения программы.

* + 1. Текст программы (ЯП C, операционная система GNU/Linux Ubuntu, компилятор GCC)

#include <stdio.h>

#include <stdlib.h>

void printHelp();

int main (int argc, char \*\*argv)

{

// Если было передано не два аргумента

// то вывести справку и выйти

if (argc != 3) { printHelp(); return 0; }

// Открытие файлов

printf("Открытие файлов... ");

FILE \*inputFile = fopen(argv[1], "r"), \*outputFile = fopen(argv[2], "w");

char c;

// Проверка файлов на возможность работы с ними

// Возвращает ненулевое число

if (!inputFile)

{

printf("\nОшибка: невозможно открыть \"%s\" для чтения.\n", argv[1]);

return 1;

}

if (!outputFile)

{

printf("\nОшибка: невозможно отрыть \"%s\" для записи.\n", argv[2]);

return 2;

}

printf("ОТКРЫТО!\n"); printf("Начало перемещения предложений...\n");

fseek(inputFile, 0, SEEK\_END);

int sentStart = ftell(inputFile) - 1, sentEnd = 0;

while (ftell(inputFile) > 0)

{

c = fgetc(inputFile);

if (c == '.' || c == '!' || c == '?')

{

printf("Обнаружено предложение. Начало перемещения... ");

sentEnd = sentStart; sentStart = ftell(inputFile);

for (int i = sentStart; i <= sentEnd; i++)

{ char ch = fgetc(inputFile); fwrite(&ch, 1, 1, outputFile); }

printf("ЗАВЕРШЕНО!\n"); fseek(inputFile, sentStart, SEEK\_SET);

}

fseek(inputFile, -2, SEEK\_CUR);

}

if (sentEnd)

{

sentEnd = sentStart; sentStart = 0;

fseek(inputFile, sentStart, SEEK\_SET);

for (int i = sentStart; i <= sentEnd; i++)

{ char ch = fgetc(inputFile); fwrite(&ch, 1, 1, outputFile); }

fwrite("\n", 1, 1, outputFile);

}

// Закрытие файлов и выход из программы

fclose(inputFile); fclose(outputFile);

printf("Все файлы закрыты. Программа зарвешила свою работу.\n");

return 0;

}

// Напечатать справку

// Не возвращает ничего

void printHelp()

{

printf("Использование: %s [ФАЙЛ 1] [ФАЙЛ 2]\n", PROG\_NAME);

printf("Берёт предложения из [ФАЙЛ 1] и выводит их же в [ФАЙЛ 2], но в обратном порядке.\n");

printf("\n");

printf("Параметры:\n");

printf(" -h, --help Показать эту справку и выйти\n");

fflush(stdout);

}

* + 1. Текст программы (ЯП C++, операционная система GNU/Linux Ubuntu, компилятор G++)

#include <iostream>

#include <fstream>

#include <cstring>

using namespace std;

#define BUFF\_SIZE 1024

const char DELIMITERS[] = ".?!";

void printHelp();

int getCharPosition(const char arr[], const char a);

int copyStr1ToStr2 (char \*str1, unsigned long a, unsigned long b, char \*str2, unsigned long offset);

int getStrLenght(char \*str);

int main (int argc, char \*\*argv)

{

char \*\*sentArray = (char \*\*) malloc(sizeof(char \*) \* 1024), tmpSent[BUFF\_SIZE];

int delType = -1, sentArray\_rows = 0, sentStart = 0, i = -1;

// Если было передано не два аргумента,

// то вывести справку и выйти

if (argc != 3) { printHelp(); return 0; }

cout << "Открытие файлов... ";

ifstream inFile(argv[1], ios::in);

ofstream outFile(argv[2], ios::out);

// Проверка, открыт ли файл для чтения

if (!inFile)

{

cout << "Ошибка: невозножно открыть для чтения '" << argv[1] << "'." << endl;

return 1;

}

// Проверка, открыт ли файл для записи

if (!outFile)

{

cout << "Ошибка: невозножно открыть для записи '" << argv[2] << "'." << endl;

return 2;

}

cout << "ОТКРЫТО" << endl;

cout << "Начало перемещения предложений..." << endl;

// Начало выполнения алгоритма

while(inFile)

{

inFile.read(tmpSent, BUFF\_SIZE);

\*(tmpSent + BUFF\_SIZE) = '\0';

while (\*(tmpSent + ++i))

{

delType = getCharPosition(DELIMITERS, \*(tmpSent + i));

if (delType + 1)

{

//realloc(sentArray, sizeof(char \*) \* (++sentArray\_rows));

cout << "Обнаружено предложение. Началось перемещение в файл... ";

++sentArray\_rows;

\*(sentArray + sentArray\_rows - 1) = (char \*) malloc(sizeof(char) \* (i - sentStart + 1));

sentArray[sentArray\_rows - 1][i - sentStart + 1] = '\0';

std::cout << "ЗАВЕРШЕНО. Нескопировано символов: " << copyStr1ToStr2(tmpSent, sentStart, i, sentArray[sentArray\_rows - 1], 0) << "." << std::endl;

sentStart = i + 1;

}

}

}

for (i = sentArray\_rows - 1; i >= 0; i--)

{

outFile.write(\*(sentArray + i), getStrLenght(\*(sentArray + i)));

}

// Закрытие файлов и выход из программы

inFile.close();

outFile.close();

cout << "Все файлы закрыты. Завершение работы программы." << endl;

return 0;

}

// Функция поиска первого вхождения символа a в строку arr

// Возвращет позицию первого вхождения. Иначе возвращает -1

int getCharPosition(const char arr[], const char a)

{

int i = -1;

while (arr[++i])

if (arr[i] == a)

return i;

return -1;

}

// Функция копирования символов str1[a; b] в str2, начиная с позиции offset

// Возвращает 0, если копирование прошло успешно, иначе возвращает количество символов,

// которое не удалось скопировать в строку str2

int copyStr1ToStr2 (char \*str1, unsigned long a, unsigned long b, char \*str2, unsigned long offset)

{

int i = a, j = offset;

for (; i <= b; i++, j++)

{

//if (\*(str2 + j) == '\0') break;

\*(str2 + j) = \*(str1 + i);

}

return b - i + 1;

}

// Напечатать справку

void printHelp()

{

cout << "Использование: " << PROG\_NAME << " [ФАЙЛ 1] [ФАЙЛ 2]" << endl;

cout << "Берёт предложения из [ФАЙЛ 1] и выводит их же в [ФАЙЛ 2], но в обратном порядке." << endl;

cout << endl;

cout << "Параметры:" << endl;

cout << " -h, --help Показать эту справку и выйти" << endl;

}

int getStrLenght(char \*str)

{

int l = 0;

while (\*(str++) != '\0') l++;

return l;

}

* + 1. Описание тестовых примеров

Тестовые примеры продемонстрированы в Приложении А на Рисунках А.4–А.13.

На Рисунке А.4 продемонстрировано поведение программы, если пользователь не передаст должное количество аргументов. На Рисунке А.5 – если пользователь передаст путь к входному файлу, по которому данного файла не существует.

Рисунки А.6–А.10 содержат тестовый пример 1. На Рисунке А.6 показаны исходные данные. На Рисунках А.7 и А.9 – работа программы, видная пользователю. На Рисунках А.8 и А.10 – содержимое выходного файла после выполнения каждой программы.

Рисунки А.11–А.13 содержат тестовый пример 2. На Рисунке А.11 показаны исходные данные. На Рисунках А.12 и А.13 – содержимое выходного файла после выполнения каждой программы.

## Вывод

В ходе выполнения данной лабораторной работы были получены навыки разработки приложений, умеющих работать с пользовательскими файлами; были выявлены особенности работы со строками и с файлами на языках программирования C и C++. Также закреплены навыки работы с двумерными массивами, с функциями. Полученные навыки помогут в будущем создавать более сложные программы обработки пользовательских файлов, обработки строк и предложений.