**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Алгоритмы и структуры данных»**

Тема: Рекурсия**.**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 9304 |  | Паутова Ю.В. |
| Преподаватель |  | Фиалковский М.С. |

Санкт-Петербург

2020

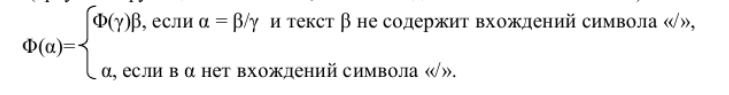
## Цель работы.

## Ознакомиться с основными понятиями и приёмами рекурсивного программирования, получить навыки программирования рекурсивных процедур и функций на языке программирования С++.

## Задание.

## Вариант 17

Функция Ф преобразования текста определяется следующим образом (аргумент функции – это текст, т. Ею последовательность символов):



Например: Ф(«ла/ска»)=«скала», Ф(«б/ру/с»)=«сруб», Ф(«ца/ри/ца»)= «царица», Ф(«ум/ри/ва/к/а»)= «аквариум». Реализовать функцию Ф рекурсивно.

## Выполнение работы.

Программе с помощью аргументов командной строки подаются два файла: файл1, содержащий строку-аргумент и файл2, в который будет записан результат преобразования строки-аргумента. Строка-аргумент записывается в объект класса string, после чего проверяется на корректность. Если строка-аргумент корректна, вызывается рекурсивная функция преобразования строки. Результат работы рекурсивной функции записывается в файл2.

Входные данные представлены в виде строки, которая может содержать вхождения символа «/», а может и не содержать их. Выходные данные представлены в виде строки, которая является преобразованием введенной строки, и строк, которые отображают глубину рекурсии.

Функции программы:

* bool isCorrect() – принимает ссылку на объект класса string, и проверяет корректность строки-аргумента. Строка-аргумент считается некорректной и функция возвращает false, если начинается или заканчивается символом «/», или в ней содержатся подряд идущие несколько символов «/», или строка-аргумент пустая. Иначе функция возвращает true.
* void Files\_read\_write() – принимает ссылки на объекты классов ifstream: in, ofstream: out, string: argument и result, vector: logs, и ссылку на переменную типа int: deep. Строка-аргумент считывается с помощью getline() из файла in. Если строка-аргумент некорректна, выводится сообщение об ошибке и функция завершается. Иначе вызывается рекурсивная функция RF(), затем результат и глубина рекурсии записываются в файл out.
* void RF() – принимает ссылки на объекты классов string: argument и result, и vector: logs и ссылку на переменную типа int: deep. Сначала с помощью метода empty() строка-аргумент проверяется на пустоту. Затем значение deep увеличивается на 1, в строке-аргументе находится индекс символа «/». Если такого символа нет, то в начало result записывается argument, в конец logs вставляется строка, состоящая из deep-1 раз повторяющихся «.», argument и переноса строки, argument очищается, и функция завершается. Если символ найден, в переменную типа size\_t: size записывается индекс найденного символа; в начало result записываются size символов из argument; в конец logs вставляется строка, состоящая из deep-1 раз повторяющихся «.», size символов из argument и переноса строки; из начала argument удаляются size+1 символ; вызывается функция RF().
* int main() – принимает переменную типа int: argc и массив строк: argv. Создаются объекты классов string: argument и result, vector с типом элементов string: logs и переменная типа int: deep = 0. Если argc меньше 3, выводится сообщение о том, что программа была запущенна без необходимого количества аргументов. Иначе происходит открытие файла на чтение. Если открыть файл не вышло, выводится сообщение об ошибке. Затем открывается файл на запись. Если открыть файл не вышло, также выводится сообщение об ошибке. Если оба файла успешно открылись, вызывается функция Files\_read\_write(), затем оба файла закрываются и программа завершается.

## Разработанный программный код см. в приложении А.

Результаты тестирования см. в приложении B.

## Выводы.

Произошло ознакомление с основными понятиями и приёмами рекурсивного программирования, были получены навыки программирования рекурсивных процедур и функций на языке программирования С++.

Была разработана программа для преобразования текста согласно определению функции Ф. Функция Ф реализована рекурсивно. Также реализованы функция для проверки аргумента функции на корректность( bool isCorrect()) и функция считывания из файла аргумента и записи результат в файл(void Files\_read\_write()). Произведено тестирование правильности работы программы.

# Приложение А Исходный код программы

Название файла: **source/lb1.cpp**

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <locale>

using namespace std;

bool isCorrect(string& argument);

void Files\_read\_write(ifstream& in, ofstream& out, string& argument, string& result, vector<string> &logs, int& deep);

void RF(string& argument, string& result, vector<string> &logs, int& deep);

int main(int argc, char\*\* argv){

    setlocale(LC\_ALL,"ru");

    string argument;

    string result;

    vector<string> logs;

    int deep = 0;

    if (argc < 3){

        std::cout << "Команда для запуска программы: lab1 /path/to/input /path/to/output" << endl;

        return 1;

    }

    else{

        ifstream in(argv[1]);

        if (!in.is\_open()){

            std::cout << "Не удалось открыть файл " << argv[1] << endl;

            return 1;

        }

        ofstream out(argv[2]);

        if (!out.is\_open()){

            std::cout << "Не удалось открыть файл " << argv[2] << endl;

            return 1;

        }

        Files\_read\_write(in, out, argument, result, logs, deep);

        in.close();

        out.close();

    }

    return 0;

}

bool isCorrect(string& argument){

    if (argument[argument.length()-1] == '/' || argument[0] == '/' || argument.empty())

        return false;

    size\_t index = 0;

    while (argument.find("/", index) != string::npos){

        index = argument.find("/", index);

        if (argument[++index] == '/')

            return false;

    }

    return true;

}

void Files\_read\_write(ifstream& in, ofstream& out, string& argument, string& result, vector<string> &logs, int& deep){

    getline(in, argument);

    if (!isCorrect(argument)){

        cout << "Некорректно введен аргумент." <<endl;

        return;

    }

    RF(argument, result, logs, deep);

    out << "result = " << result << "\nГлубина:\n";

    for (const auto &str : logs)

        out << str;

}

void RF(string& argument, string& result, vector<string> &logs, int& deep){

    if (argument.empty()){

        return;

    }

    deep++;

    if (argument.find("/") != string::npos){

        size\_t size = argument.find("/");

        result.insert(0, argument, 0 ,size);

        logs.emplace\_back(string(deep-1,'.') + move(string(argument, 0, size)) + "\n");

        argument.erase(0, size+1);

        RF(argument, result, logs, deep);

    }

    else{

        result.insert(0, argument);

        logs.emplace\_back(string(deep-1,'.') + argument + "\n");

        argument.clear();

        return;

    }

}

Название файла: **Makefile**

lab1: asd.cpp

    g++ asd.cpp -o lab1

run\_tests: lab1

    chmod +x ./myscript

Название файла: **./myscript**

#!/bin/bash

arg1=$(cat Tests/test1.txt)

echo "Test 1"

echo "argument = $arg1"

./lab1 Tests/test1.txt result1

cat result1

echo -e "\nTest 2"

arg2=$(cat Tests/test2.txt)

echo "argument = $arg2"

./lab1 Tests/test2.txt result2

cat result2

echo -e "\nTest 3"

arg3=$(cat Tests/test3.txt)

echo "argument = $arg3"

./lab1 Tests/test3.txt result3

cat result3

echo -e "\nTest 4\nВеденная команда: ./lab1"

./lab1

echo -e "\nTest 5"

arg5=$(cat Tests/test5.txt)

echo "argument = $arg5"

./lab1 Tests/test5.txt result5

res5=$(cat result5)

if [$res5 = ""]

then

rm result5

fi

echo -e "\nTest 6"

arg6=$(cat Tests/test6.txt)

echo "argument = $arg6"

./lab1 Tests/test6.txt result6

res6=$(cat result6)

if [$res6 = ""]

then

rm result6

fi

echo -e "\nTest 7"

./lab1 Tests/test7.txt result7

# Приложение B тестирование

Таблица B.1 – Результаты тестирования

|  |  |  |
| --- | --- | --- |
| № | Входные данные | Выходные данные |
| 1 | ta/ki/Ni |  |
| 2 | ум/ри/ва/к/а |  |
| 3 | October |  |
| 4 | ./lab1 | Команда для запуска программы:  lab1 /path/to/input /path/to/output |
| 5 | A//S/D | Некорректно введен аргумент. |
| 6 | /И/Т/Э/Л | Некорректно введен аргумент. |
| 7 | ./lab1 Tests/test7.txt result7 | Не удалось открыть файл Tests/test7.txt |