МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Алгоритмы и структуры данных»

Тема: Рекурсия

Студент гр. 8304 _____ Мешков М.А.

Преподаватель _____ Фирсов М.А.

Санкт-Петербург

2019

Цель работы.

Изучить методы решения задач с помощью рекурсии и основы составления эффективных алгоритмов.

Постановка задачи.

1. Разработать программу, использующую рекурсию.

2. Сопоставить рекурсивное решение с итеративным решением задачи.

3. Сделать вывод о целесообразности и эффективности рекурсивного решения задачи.

Вариант 17.

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

Ф(a) = Ф(c)b, если a = b/c и текст b не содержит вхождений символа "/".

Ф(a) = a, если в a нет вхождений символа "/".

Описание алгоритма.

Для решения поставленной задачи была написана рекурсивная функция f, которая принимает строку и целочисленное число, предназначенное для выравнивания вывода при отладке. В строке ищется первое вхождение символа "/", дальнейшие действия зависят от того был ли найден этот символ. Если символ найден не был, то функция просто возвращает переданную ей строку. Если символ был найден, функция работает с двумя частями данной строки (до символа "/" и после), она возвращает результат вызова самой себя с правой частью данной строки, добавляя в конец левую часть.

Описание основных структур данных и функций.

Функция f реализует алгоритм, требуемый в задании и описанный выше.

Функция launchTest выполняет тестирование программы, с помощью тестов заданных в файле Tests/test, в нечетных строках этого файла заданы

входные данные для программы, в четных строках заданы ожидаемые выходные данные для предыдущей строки.

Функция main выполняет взаимодействие с пользователем и позволяет ему выбрать между запуском алгоритма с некоторыми входными данными и запуском тестирования программы (для тестирования программу нужно запустить с флагом --test).

Глобальная переменная isDebug контролирует нужно ли выводить отладочную информацию.

Тестирование.

Программа была протестирована на следующих данных

Ввод	Вывод
ла/ска	скала
ца/ри/ца	царица
ум/ри/ва/к/а	аквариум
а	а
/	
аа//бб	ббаа

Выводы.

В ходе выполнения работы был реализован рекурсивный алгоритм решения данной задачи. Итеративное решение можно сделать более эффективным как по скорости работы, так и по памяти, так как при рекурсивном решении происходит множественные вызовы функцией самой себя, на что необходимо время, также растет расход памяти при каждом рекурсивном вызове.

ПРИЛОЖЕНИЕ А

Исходный код

```
#include <iostream>
#include <string>
#include <cstring>
#include <fstream>

using namespace std;

static bool isDebug = true;

string f(const string &a, unsigned indent = 0)
{
    auto getIndent = [indent] {
        string indentStr;
        for (unsigned i = 0; i < indent; i++)
            indentStr += " │  ";
        return indentStr;
    };
    auto showResult = [getIndent](string result) {
        cout << getIndent() << " └ = " << result << endl;
    };

    if (isDebug)
        cout << getIndent() << "f(" << a << "):" << endl;

    auto p = a.find("/");
    auto left = a.substr(0, p);

    if (p == string::npos) {
```

```
        if (isDebug) {
            cout << getIndent() << " │ " << left << endl;
            showResult(a);
        }
        return a;
    }
    else {
        auto right = a.substr(p + 1);
        auto result = f(right, indent + 1) + left;
        if (isDebug) {
            cout << getIndent() << " │ +" << endl;
            cout << getIndent() << " │ " << left << endl;
            showResult(result);
        }
        return result;
    }
}

void launchTests() {
    auto oldIsDebug = isDebug;
    isDebug = false;
    ifstream fin("Tests/test");
    if (fin.is_open()) {
        string in, out;
        int i = 1;
        while (true) {
            getline(fin, in);
            getline(fin, out);
            if (fin.eof())
                break;
```

```
            cout << "Test " << i++ << " is ";
            if (f(in) == out)
                cout << "ok";
            else
                cout << "FAILED";
            cout << endl;
        }
        fin.close();
    }
    else {
        cerr << "The test file is not found.";
    }
    isDebug = oldIsDebug;
}

int main(int argc, char** argv)
{
    auto showHelp = []() {
        cout << "Enter the string as a command-line argument." << endl;
        cout << "Use --test to launch the tests." << endl;
    };

    if (argc > 1) {
        if (strcmp(argv[1], "--test") == 0)
            launchTests();
        else {
            auto result = f(argv[1]);
            cout << "Result: " << result << endl;
        }
    }
```

```
    else {
        showHelp();
    }
    return 0;
}
```