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. 9303 _____ Низовцов Р. С.

Преподаватель _____ Филатов А. Ю.

Санкт-Петербург

2020

Цель работы.

Ознакомиться с понятием «рекурсия», изучить ее особенности и реализовать программу, решающую поставленную задачу с помощью рекурсии.

Задание.

Вариант 17

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

Ф(γ)β, если α = β/γ и текст β не содержит вхождений символа «/»,

Ф(α) = α, если в α нет вхождений символа «/».

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

Основные теоретические положения.

В программировании рекурсия — вызов функции (процедуры) из неё же самой, непосредственно (простая рекурсия) или через другие функции (*сложная* или *косвенная рекурсия*), например, функция A вызывает функцию B, а функция B — функцию A. Количество вложенных вызовов функции или процедуры называется глубиной рекурсии. Рекурсивная программа позволяет описать повторяющееся или даже потенциально бесконечное вычисление, причём без явных повторений частей программы и использования циклов.

Структурно рекурсивная функция на верхнем уровне всегда представляет собой команду ветвления (выбор одной из двух или более альтернатив в зависимости от условия (условий), которое в данном случае уместно назвать «условием прекращения рекурсии»), имеющую две или более альтернативные ветви, из которых хотя бы одна является *рекурсивной* и хотя бы одна —

терминальной. Рекурсивная ветвь выполняется, когда условие прекращения рекурсии ложно, и содержит хотя бы один рекурсивный вызов — прямой или опосредованный вызов функцией самой себя. Терминальная ветвь выполняется, когда условие прекращения рекурсии истинно; она возвращает некоторое значение, не выполняя рекурсивного вызова. Правильно написанная рекурсивная функция должна гарантировать, что через конечное число рекурсивных вызовов будет достигнуто выполнение условия прекращения рекурсии, в результате чего цепочка последовательных рекурсивных вызовов прервётся и выполнится возврат.

Выполнение работы.

Для выполнения программы была реализована функция bool functionF(ifstream& inFile, ofstream& outFile).

В функции int main() запрашивается путь к файлу с данными для обработки, проверяется её корректность. Потом запускается цикл и работает, пока functionF не вернет true. После этого информирует об окончании работы, закрывает все файлы.

В функции bool functionF(ifstream& inFile, ofstream& outFile) проводится посимвольное чтение из inFile до знака „/” , знака конца строки или конца файла. Далее проводится проверка на завершение строки, и, если она не завершена, функция вызывает себя. Функция возвращает проверку на достижения конца файла и этот флаг обрабатывается в main функции.

Выводы.

Ознакомился с понятием «рекурсия», изучил ее особенности и реализовал программу, решающую поставленную задачу с помощью рекурсии.

Была реализована программа, включающая в себя рекурсивную функцию functionF для обработки строк, хранящихся в файле. Программа выполняет чтение, запись результата в файл, а также рекурсивно производит проверку и обработку считанного текста.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

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

using namespace std;

bool functionF(ifstream& inFile, ofstream& outFile);

int main()
{
   string file_name;
   cout << "Enter the name of an input file: " << endl;
   cin >> file_name;
   ifstream inFile(file_name);
   if (!inFile.is_open()){
      cout << "Permission denied or wrong path";
      return 0;
   }
   cout << "Result was saved in result.txt" << endl;
   ofstream outFile("/home/rostislav/result.txt");
   int rowCount = 1;
   while(!functionF(inFile, outFile)){
      outFile << endl;
      cout << "Reading " << rowCount << "row" << endl;
      rowCount++;
   }
   cout << "End of work" << endl;
```

```
    inFile.close();
    outFile.close();
    return 0;
}

bool functionF(ifstream &inFile, ofstream &outFile){
    bool isEOF = false;
    string part = "";
    char currentChar = '\0';
    currentChar = inFile.get();
    while(currentChar != EOF && currentChar != '/' && currentChar != '\n'){
        part = part + currentChar;
        currentChar = inFile.get();
    }
    if(currentChar == '/'){
        isEOF = functionF(inFile, outFile);
    }
    outFile << part;
    if(isEOF || (currentChar == EOF))
        return true;
    return false;
}
```

ПРИЛОЖЕНИЕ Б

ТЕСТИРОВАНИЕ

Таблица Б.1 — Примеры тестовых случаев

№ п/п	Входные данные	Выходные данные	Комментарий
1.	ла/ска б/ру/с ца/ри/ца ум/ри/ва/к/а	скала сруб царица аквариум	Программа работает корректно
2.	///получилось, Ник? да, Майк!///	получилось, Ник? Да, Майк!	Программа работает корректно
3.			Программа работает корректно