**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №3**

**по дисциплине «Программирование»**

Тема: Строки. Рекурсия, циклы, обход дерева

|  |  |  |
| --- | --- | --- |
| Студентка гр. 3341 |  | Чинаева М.Р. |
| Преподаватель |  | Глазунов С.А. |

Санкт-Петербург

2024

## Цель работы

Цель работы заключается в разработке программы на языке программирования, которая осуществляет рекурсивный обход иерархии папок и файлов в заданной структуре, анализирует содержимое текстовых файлов, выполняет математические операции в соответствии с правилами задания и выводит на экран итоговый результат вычислений, основанный на содержимом файлов и вложенных папок.

## Задание

Вариант 2

Задана иерархия папок и файлов по следующим правилам:

название папок может быть только "add" или "mul"

В папках могут находиться другие вложенные папки и/или текстовые файлы

Текстовые файлы имеют произвольное имя с расширением .txt

Содержимое текстовых файлов представляет собой строку, в которой через пробел записано некоторое количество целых чисел

Требуется написать программу, которая, запускается в корневой директории, содержащей одну папку с именем "add" или "mul" и вычисляет и выводит на экран результат выражения состоящего из чисел в поддиректориях по следующим правилам:

Если в папке находится один или несколько текстовых файлов, то математическая операция, определяемая названием папки (add = сложение, mul = умножение) применяется ко всем числам всех файлов в этой папке

Если в папке находится еще одна или несколько папок, то сначала вычисляются значения выражений, определяемые ими, а после используются уже эти значения

Ваше решение должно находиться в директории /home/box, файл с решением должен называться solution.c. Результат работы программы должен быть записан в файл result.txt. Ваша программа должна обрабатывать директорию, которая называется tmp.

## Основные теоретические положения

Рекурсия – это вызов функции в ней самой же [7]. То есть если внутри

выполнения функции A происходит вызов функции А, то это рекурсия.

Функции для работы с файлами определены в заголовочном файле stdio.h. Для работы с файлами используется файловый поток, реализованный структурой FILE. Напрямую работа с данной структурой не производится, поэтому содержимое структуры рассмотрено не будет. Основными функциями для работы с файлами являются:

• FILE \* fopen(const char \* filename, const char \* mode) – открывает файл с названием filename в режиме mode и возвращает указатель на файловый поток FILE;

• int fclose (FILE \* stream) – закрывает файловый поток stream, полученный из fopen().

Основными режимами открытия файла являются “r” и “w”: первый режим открывает файл на чтение, второй – на запись.

Определение структур и функций для работы с директориями находятся в заголовочном файле dirent.h. Для работы с директориями используется поток директории, реализованный структурой DIR (по аналогии с файловым потоком FILE), которая используется в качестве аргумента для функций. Так как напрямую работа с данной структурой не производится, то ее структура не будет рассмотрена. Основными функциями для работы с директориями являются:

• DIR \*opendir(const char \*dirname) – открывает директорию dirname и возвращает указатель на поток директории DIR. Если не удалось открыть директорию, то возвращается NULL;

• int closedir (DIR \*dir) – закрывает поток директории dir, который был получен из opendir();

• struct dirent \*readdir (DIR \*dirstream) – считывает следующий элемент из потока директории dirstream и возвращает указатель на прочитанный элемент. Если в потоке больше не осталось элементов, то возвращается NULL.

## Выполнение работы

Подключаются библиотеки <stdio.h>, <dirent.h> и <string.h> дляя работы с файлами, директориями и строками

В функции main создается строка с названием начальной директории, параметру result присваивается значение, которое возвращает функция operation. Далее создается или открывается файл, в который должен быть записан результат, записывается результат и файл закрывается.

Функции:

1. int operation(const char\* name\_dir, int check\_operation)

Получает на вход имя директории и численное значение, обозначающее, какую операцию надо выполнить.

Открывается нужная директория, считывается ее первый элемент. Если операция сложение – result равен 0, если умножение, то единице. Далее с помощью цикла while совершается обход данной директории. Если элемент – директория и ее название не является «.» или «..» (ссылка на текущую и родительскую директорию), то проверяется является ли эта папка со сложением или с умножением, если является то функция operation вызывается рекурсивно и в нее уже передается название этой директории. Если же элемент файл, вызывается функция для обработки файлов согласно условиям. Стоит обратить внимание на переменную check\_iteration, которая обнуляется в начале каждого цикла и становится равной 1, если элемент подходит под условия. Это связано с тем, что в директориях могут содержатся элементы, не соответствующие условиям. Далее если эта переменная равна 1, то в соответствие с кодом операции выполнятся нужное действие. Считывается следующий элемент директории. После завершения цикла директория закрывается.

Функция возвращает результат обхода текущей директории.

2. int operation\_file(char\* file\_name, int check\_operation)

На вход получает путь к файлу и числовое значение, обозначающее какую операцию надо выполнить.

Открывается файл. Числа из файла считываются последовательно с помощью цикла while и или прибавляются, или перемножаются между собой. Файл закрывается.

Функция возвращает результат операции.

Разработанный программный код см. в приложении А.

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные в файле result.txt | Комментарии |
|  | add:  >file.txt:1  >file1.txt:2 3  >add:  >>file2.txt:1 4 | 11 | Тест для сложения чисел |
|  | mul:  >file.txt:1  >file1.txt:2 3  >mul:  >>file2.txt:1 4 | 24 | Тест для умножения чисел |
|  | mul:  >file.txt:0  >file1.txt:5 7  >add:  >>file2.txt:4 6 7 4 | 0 | Тест крайнего случая для умножения |
|  | file.txt: 1  file1.txt: 1  file2.txt: 2 2  file3.txt: 7  file4.txt: 1 2 3  file5.txt: 3 -1 | 226 | Тест с e.moevm |

## Выводы

В ходе выполнения данной работы были приобретены навыки эффективного использования рекурсивных методов для обхода файловых систем, анализа содержимого текстовых файлов и выполнения математических операций в соответствии с заданными правилами. Разработка программы, способной автоматически обрабатывать информацию из различных файлов и директорий, позволила улучшить навыки программирования и решения сложных задач.

# Приложение А Исходный код программы

Название файла: solution.c

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <dirent.h>

#include <string.h>

#define MUL 1

#define ADD 2

#define SIZE\_NAME\_DIR 1000

#define LINK\_PARRENT\_DIR "."

#define LINK\_CURRENT\_DIR ".."

#define ADD\_DIR "add"

#define MUL\_DIR "mul"

int operation\_file(char\* file\_name, int check\_operation) {

FILE\* file = fopen(file\_name, "r");

int number = 0;

int result = 0;

if (check\_operation == ADD) {

while (fscanf(file, "%d", &number) == 1) {

result += number;

}

}

else {

result = 1;

while (fscanf(file, "%d", &number) == 1) {

result \*= number;

}

}

fclose(file);

return result;

}

int operation(const char\* name\_dir, int check\_operation) {

DIR\* dir = opendir(name\_dir);

struct dirent\* de = readdir(dir);

int result = 0;

if (check\_operation == MUL) {

result = 1;

}

int result\_iteration = 0;

int check\_iteration;

while (de) {

check\_iteration = 0;

if (de->d\_type == DT\_DIR && strcmp(de->d\_name, LINK\_PARRENT\_DIR) != 0 && strcmp(de->d\_name, LINK\_CURRENT\_DIR) != 0) {

char subdir\_name[SIZE\_NAME\_DIR];

sprintf(subdir\_name, "%s/%s", name\_dir, de->d\_name);

if (strcmp(de->d\_name, ADD\_DIR) == 0) {

result\_iteration = operation(subdir\_name, ADD);

check\_iteration = 1;

}

if (strcmp(de->d\_name, MUL\_DIR) == 0) {

result\_iteration = operation(subdir\_name, MUL);

check\_iteration = 1;

}

}

if (de->d\_type == DT\_REG) {

char file\_name[1000];

sprintf(file\_name, "%s/%s", name\_dir, de->d\_name);

result\_iteration = operation\_file(file\_name, check\_operation);

check\_iteration = 1;

}

if (check\_iteration) {

if (check\_operation == 0) {

return result\_iteration;

}

if (check\_operation == ADD) {

result += result\_iteration;

}

else {

result = result \* result\_iteration;

}

}

de = readdir(dir);

}

closedir(dir);

return result;

}

void result\_to\_file(int result) {

FILE\* result\_file = fopen("/home/box/result.txt", "w");

fprintf(result\_file, "%d", result);

fclose(result\_file);

}

int main() {

const char\* name\_first\_dir = "./tmp";

int result = 0;

result = operation(name\_first\_dir, 0);

result\_to\_file(result);

return 0;

}