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: Обход файловой системы

Студент гр. 8303 _____ Гришин К. И.

Преподаватель _____ Чайка К. В.

Санкт-Петербург
2019

Цель работы

Ознакомиться с библиотекой «dirent.h», предназначенной для просмотра характеристик файлов или директорий.

Задание

Дана некоторая корневая директория, в которой может находиться некоторое количество папок, в том числе вложенных. В этих папках хранятся некоторые текстовые файлы, имеющие имя вида *<filename>.txt*

В каждом текстовом файле хранится одна строка, начинающаяся с числа вида: *<число><пробел><латинские буквы, цифры, знаки препинания> ("124 string example!")*

Требуется написать программу, которая, будучи запущенной в корневой директории, выведет строки из файлов всех поддиректорий в порядке возрастания числа, с которого строки начинаются.

Ход выполнения работы

Для нахождения содержимого файлов в системе директорий создана рекурсивная функция *void* ***content****(char* path, char ***cont, int *count)*, на вход которой подается путь директории, содержимое файлов в которой требуется рассмотреть, ссылку на массив строк в который будет накапливаться содержимое файлов и ссылку на целое число, в которое будет записано количество строк массива *cont.*

Сначала функция открывает сообщенную ей директорию, затем рассматривает каждый файл в ней исходя из значения *d_type* структуры DIR, которая описывает прочитанный файл, если тот оказался типа *<file>.txt*, то он открывается и его содержимое записывается в массив *cont*, если же очередной файл оказался директорией то к ней применяется функция *content*, в которую сообщается уже новый путь (*начальный путь/название текущей директории).*

После применения функции *content*, в распоряжении программы есть массив, в котором хранятся первые строки каждого прочитанного файла. Т.к. задача требует отсортированный вывод, применяется функция *qsort* стандартной библиотеки *«stdlib.h»*, для которой написан компаратор *int* ***cmp****(const void* a, const void *b)*, который сравнивает числа вначале каждой строки путем использования функции *sscanf*, несмотря на то, что для данной цели использование такой функции нецелесообразно, в задании гарантируется, что вначале каждой строки находится число, а значит ошибок сортировки не будет.

После сортировки полученный массив записывается в файл *«result.txt»*.

Освобождается память, занимаемая массивом *cont*.

Код программы находится в приложении А к лабораторной работе.

Вывод

В ходе лабораторной работы была изучена библиотека *«dirent.h»*, которая позволяет сделать обход файловой системы, а также изучен метод написания рекурсивных функций. Написана программа, которая позволяет вывести содержимое файлов в заданной директории, а также вложенных в нее.

ПРИЛОЖЕНИЕ А

```
#include <stdlib.h>
#include <stdio.h>
#include <dirent.h>
#include <string.h>


void content(char* path, char ***cont, int *count);
int cmp(const void *a, const void *b);

int main(){
   char **cont;
   int count = 0;
   char path[] = "./";
   content(path, &cont, &count);

   for(int i = 0; i < count; i++)
           printf("%s", cont[i]);

   qsort(cont, count, sizeof(char*), cmp);
   printf("\nSorted\n");

   for(int i = 0; i < count; i++)
        printf("%s", cont[i]);

   FILE *result = fopen("./result.txt", "wb");
   for(int i = 0; i < count; i++)
           if (cont[i][strlen(cont[i])-2] != '\n')
                   fprintf(result, "%s", cont[i]);
   fclose(result);

   if(count){
           for(int i = 0; i < count; i++) free(cont[i]);
           free(cont);
   }
   return 0;
}


int cmp(const void* a, const void *b){
   char *strA = *(char**)a;
   char *strB = *(char**)b;
   long int numA;
   long int numB;
   sscanf(strA, "%ld", &numA);
   sscanf(strB, "%ld", &numB);
   if (numA < numB) return -1;
   if (numA == numB) return 0;
   return 1;
}


void content(char *path, char ***cont, int *count){
   DIR *dir = opendir(path);
   if (dir){
           struct dirent *de = readdir(dir);
           while(de){
                   if (de->d_type == 4 && strcmp(de->d_name, "..") && strcmp(de->d_name, ".")){
                           char nested_path[100] = "";
                           strcpy(nested_path, path);
                           strcat(nested_path, "/");
                           strcat(nested_path, de->d_name);
                           content(nested_path, cont, count);
                   }
                   else if (de->d_type == 8 && strcmp(de->d_name+strlen(de->d_name)-4, ".txt")==0){

                           char* sent = (char*)calloc(102, sizeof(char));
                           char txtpath[100] = "";
                           strcpy(txtpath, path);
                           strcat(txtpath, "/");
                           strcat(txtpath, de->d_name);
                           FILE *txt = fopen(txtpath, "r");
                           fgets(sent, 100, txt);
                           fclose(txt);
                           if (!(*count)) *cont = (char**)malloc(sizeof(char*));
                               else{
                                       char **buffer = (char**)realloc(*cont, ((*count)+1)*sizeof(char*));
                                       if (buffer) *cont = buffer;
                                       else exit(1);
                               }
                           (*cont)[*count] = sent;
                           (*count)++;
                   }
                   de = readdir(dir);
           }
   }
   closedir(dir);
}
```