Министерство науки и высшего образования Российской Федерации

Новосибирский государственный технический университет

Управление ресурсами в вычислительных системах

Лабораторная работа №2

Факультет: прикладной математики и информатики

Группа: ПМ-63

Студенты: Кожекин М.В.

Утюганов Д.С.

Преподаватели: Стасышин В. М.

Хайленко Е. А.

Новосибирск

2019

1. Условие (Вариант №10)

Разработать программу, которая осуществляет просмотр текущего каталога и выводит на экран имена находящихся в нём каталогов, упорядочив их по числу файлов и каталогов, содержащихся в отображаемом каталоге. Для каждого такого каталога указывается число содержащихся в нём файлов и каталогов.

1. Анализ задачи

Если заданный каталог существует, то рекурсивно обходим все его поддиректории и считаем количество папок и файлов внутри. В конце выводим название каждой поддиректории и число содержащихся в ней файлов и папок.

1. Используемые программные средства

**Функции:**

DIR \*opendir(char \*dirname) – открытие папки dirname

struct dirent \*readdir(DIR \*dirptr) – чтение содержимого папки, где dirptr – дескриптор папки

int closedir(DIR \*ptr) – закрытие папки

char \*strcpy(char \* destination, const char \* source) – копирует строку source в строку destination

char \*strcat(char \* destination, const char \* source) – добавляет строку source в конец строкиdestination

int strcmp(const char \* string1, const char \* string2) – сравнивает строки string1 и string2

**Структуры:**

struct pair – число файлов и папок внутри директории

struct directory – название поддиректории, число файлов и число папок внутри неё

struct dirent – имя файлв и индексный дескриптор

1. Спецификация

Программа находится в папке ~/lab2

Чтобы собрать программу нужно ввести “make lab2”

Чтобы запустить программу, нужно использовать команду “./lab2 %dit%”, где %dir% - название директории.

В результате работы скрипта, будет выведен список всех поддиректорий и 2 числа: количество вложенных папок и число файлов.

Возможные ошибки предусмотрены и описаны при вводе команды “./lab2 –help”

1. Исходный код

**Текст программы**

|  |
| --- |
| #include <dirent.h> // Структура папки  #include <stdio.h> // Стандартный ввод/вывод  #include <stdlib.h> // Стандартная библиотека  #include <string.h> // Строки  #include <errno.h> // Проверка существования папки  #include <sys/types.h>  #include <sys/stat.h>  #include <unistd.h>  struct pair {  int foldersCount; // Число папок  int filesCount; // Число файлов  };  struct directory {  int foldersCount; // Число папок  int filesCount; // Число файлов  char \*name; // Название поддиректории  };  /\*\*  @function sum  \* Функция, находящая суммарное число файлов и папок  @param a,b  @return Суммарное число файлов и папок  \*/  struct pair sum(struct pair a, struct pair b) {  struct pair result;  result.foldersCount = a.foldersCount + b.foldersCount;  result.filesCount = a.filesCount + b.filesCount;  return result;  }  /\*\*  @function compare  \* Функция сравнения двух поддиректорий по количеству файлов и папок  @param a,b - элементы массива, которые будут сравниваться внутри qsort  @return позитивный результат, если в первой папке хранится больше файлов  \* и папок, чем во второй, а иначе наоборот  \*/  int compare(const void \*a, const void \*b) {  int l = ((struct directory \*)a)->foldersCount + ((struct directory \*)a)->filesCount;  int r = ((struct directory \*)b)->foldersCount + ((struct directory \*)b)->filesCount;  return (r-l);  }  /\*\*  @function qetQuantityOfFoldersAndFiles  \* Функция, которая определяет число файлов и папок внутри директории  @param path - текущая директория  @return пара: число папок, число файлов  \*/  struct pair getQuantityOfFoldersAndFiles(const char \*path) {  // Начальное число папок равно нулю  struct pair quantity = {0, 0};    DIR \*dir;  struct dirent \*entry;  struct stat buf;  // Открываем директорию  dir = opendir(path);  if (!dir) {  printf("Can't open directory %s\n", path);  return quantity;  }  // Выводит все файлы и поддерректории внутри папки  while ( (entry = readdir(dir)) != NULL) {  char \*newPath = (char\*) calloc ((strlen(path) + 1 + strlen(entry->d\_name) + 1), sizeof(char));  strcpy(newPath, path);  strcat(newPath, "/");  strcat(newPath, entry->d\_name);    if (stat(newPath, &buf) != 0) {  printf("Error in stat.\n");  printf("%s\n", newPath);  return quantity;  }    if (S\_ISREG(buf.st\_mode)) { // Файл  quantity.filesCount++;  } else if ( S\_ISDIR(buf.st\_mode) && strcmp(entry->d\_name, ".") != 0 &&  strcmp(entry->d\_name, "..") !=0 ) { // Папка, но не "." или ".."  quantity.foldersCount++;  quantity = sum(quantity, getQuantityOfFoldersAndFiles(newPath));  }  free(newPath);  }  closedir(dir);  return quantity;  }  int main(int argc, char \*argv[]) {  if (argc > 2) {  printf("Ошибка: слишком длинный список параметров. Параметр должен быть один\n");  return -1;    } else if (argc < 2) {  printf("Ошибка: слишком короткий список параметров. Параметр должен быть один.\n");  return -1;  } else if (strcmp(argv[1], "--help") == 0) {  printf("Инструкция:\n");  printf("Введите название каталога, для которого вы хотите вывести имена находящихся в нём каталогов, упорядочив их по числу файлов и каталогов, содержащихся в отображаемом каталоге\n");  printf("Если введённый вами каталог не существует, то будет выведено сообщение о соответсвующей ошибке.\n");  printf("Если введено неверное число параметров (не равно 1), то выводится сообщение об ошибке.\n");  return 0;  }  char \*path = argv[1];  int array\_len = 0;  struct dirent \*entry;  struct directory \*array = (struct directory\*) malloc ( 0 \* sizeof(struct directory));  struct stat buf;  // Открываем директорию  DIR \*dir = opendir(path);  if (ENOENT == errno) {  printf("Directory does not exist.\n");  return -1;  } else if (!dir) {  printf("Can't open directory %s\n", path);  return -1;  }  // Выводит все файлы и поддерректории внутри папки  while ( (entry = readdir(dir)) != NULL) {  char \*newPath = (char\*) calloc ((strlen(path) + 1 + strlen(entry->d\_name) + 1), sizeof(char));  strcpy(newPath, path);  strcat(newPath, "/");  strcat(newPath, entry->d\_name);    if (stat(newPath, &buf) != 0) {  printf("Error in stat.\n");  printf("%s\n", newPath);  return -1;  }  if ( S\_ISDIR(buf.st\_mode) && strcmp(entry->d\_name, ".") != 0 &&  strcmp(entry->d\_name, "..") !=0 ) { // Папка  struct pair p = getQuantityOfFoldersAndFiles(newPath);  array\_len++;  array = (struct directory\*)realloc(array, array\_len \* sizeof(struct directory));  array[array\_len-1].filesCount = p.filesCount;  array[array\_len-1].foldersCount = p.foldersCount;  array[array\_len-1].name = entry->d\_name;  }  free(newPath);  }  closedir(dir);    // Сортируем по убыванию  qsort(array, array\_len, sizeof(struct directory),compare);  // Выводим подкатологи и число файлов и папок содержащихся в них  printf("DIR\tf\td\n");  for (int i = 0; i<array\_len; ++i){  printf("%s\t%d\t%d\n", array[i].name, array[i].filesCount, array[i].foldersCount);  }  // Освобождаем память  free(array);  return 0;  } |

**Содержимое файла Makefile**

|  |
| --- |
| # Makefile for lab #2  lab2:  gcc lab2.c -std=gnu99 -o lab2  clean:  rm lab2 |

1. Набор тестов

**Тест 1**

Описание: пустая папка

|  |  |
| --- | --- |
|  | [pmi-b6307@students lab2]$ ./lab2 EmptyFolder  DIR f d  [pmi-b6307@students lab2]$ |

**Тест 2**

Описание: папка, содержащая файлы

|  |  |
| --- | --- |
|  | [pmi-b6307@students lab2]$ ./lab2 FolderWithfiles  DIR f d  [pmi-b6307@students lab2]$ |

**Тест 3**

Описание: бинарное дерево папок с глубиной 3

|  |  |
| --- | --- |
|  | [pmi-b6307@students lab2]$ ./lab2 f11  DIR f d  f21 1 3  f22 0 1  f23 0 0  [pmi-b6307@students lab2]$ |

**Тест 4**

Описание: домашняя директория

|  |  |
| --- | --- |
|  | [pmi-b6307@students ~]$ lab2/./lab2 ~  DIR f d  lab2 7 10  lab1 5 10  .config 3 3  .local 2 3  .cache 2 3  .mozilla 0 2  [pmi-b6307@students ~]$ |