УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе № 6

по предмету «Архитектура компьютерной техники и операционных систем»

Вариант 12

Выполнил:

Крутько А.А.

гр. 251004

Проверил:

Леванцевич В.А.

Минск 2023

Задание № 5

Написать программу, находящую в заданном каталоге и всех его подкаталогах все файлы, заданного размера. Имя каталога задаётся пользователем в качестве первого аргумента командной строки. Диапазон от ***N1*** до ***N2*** задается в аргументах командной строки. Программа выводит результаты поиска в файл (четвертый аргумент командной строки) в виде полный путь, имя файла, его размер. На консоль выводится общее число просмотренных файлов.

Код программы:

#include <stdio.h>

#include <fcntl.h>

#include <sys/stat.h>

#include <sys/types.h>

#include <dirent.h>

#include <sys/time.h>

#include <stdlib.h>

#include <string.h>

typedef struct stat STAT;

typedef struct dirent DIRENT;

int searchFiles(const char\* catalogPath, int minSize, int maxSize, FILE\* outputFile) {

int totalFiles = 0;

DIRENT\* entry;

DIR\* currentDir = opendir(catalogPath);

if (currentDir == NULL) {

fprintf(outputFile, "Error opening directory %s\n\n", catalogPath);

return totalFiles;

}

while (entry = readdir(currentDir)) {

if (strcmp(entry->d\_name, ".") != 0 && strcmp(entry->d\_name, "..") != 0) {

char\* path = (char\*)malloc(strlen(catalogPath) + NAME\_MAX + 2); // +2 для '/' и '\0'

strcpy(path, catalogPath);

strcat(path, "/");

strcat(path, entry->d\_name);

STAT buf;

if (stat(path, &buf) == -1) {

fprintf(outputFile, "Error reading file information %s\n\n", path);

}

if (S\_ISDIR(buf.st\_mode) != 0)

totalFiles += searchFiles(path, minSize, maxSize, outputFile);

else if (S\_ISREG(buf.st\_mode)) {

off\_t fileSize = buf.st\_size;

if (fileSize >= minSize && fileSize <= maxSize) {

fprintf(outputFile, "Path: %s,\nName: %s,\nSize: %ld\n\n", path, entry->d\_name, (long)fileSize);

totalFiles++;

}

}

free(path);

}

}

closedir(currentDir);

return totalFiles;

}

int main(int argc, char\* argv[]) {

if (argc != 5) {

fprintf(stderr, "Usage: %s <directory> <min\_size> <max\_size> <output\_file>\n", argv[0]);

return 1;

}

const char\* catalogPath = argv[1];

int minSize = atoi(argv[2]);

int maxSize = atoi(argv[3]);

const char\* outputFilePath = argv[4];

FILE \*outputFile = fopen(outputFilePath, "w");

if (outputFile == NULL) {

fprintf(stderr, "Error opening outputFile");

exit(1);

}

chmod(outputFilePath, 0666);

printf("Total files found: %d\n", searchFiles(catalogPath, minSize, maxSize, outputFile));

fclose(outputFile);

return 0;

}

Задание №3:

Найти совпадающие по содержимому файлы в двух заданных каталогах (аргументы 1 и 2 командной строки) и всех их подкаталогах. Вывести на консоль и в файл (аргумент 3 командной строки) полный путь, размер, дату создания, права доступа, номер индексного дескриптора.

Код программы:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <fcntl.h>

#include <sys/stat.h>

#include <sys/types.h>

#include <dirent.h>

#include <unistd.h>

#include <stdbool.h>

#include <time.h>

typedef struct stat STAT;

typedef struct dirent DIRENT;

int compareFiles(const char\* path1, const char\* path2) {

FILE\* file1 = fopen(path1, "r");

FILE\* file2 = fopen(path2, "r");

if (file1 == NULL || file2 == NULL) {

if (file1) fclose(file1);

if (file2) fclose(file2);

return 0;

}

int equal = 1;

while (true) {

int byte1 = fgetc(file1);

int byte2 = fgetc(file2);

if (byte1 != byte2) {

equal = 0;

break;

}

if (byte1 == EOF || byte2 == EOF) {

break;

}

}

fclose(file1);

fclose(file2);

return equal;

}

void searchAndPrintDuplicates(const char\* path1, const char\* path2, FILE\* outputFile) {

DIRENT\* entry1, \*entry2;

DIR\* dir1 = opendir(path1);

DIR\* dir2 = opendir(path2);

if (dir1 == NULL || dir2 == NULL) {

if (dir1) closedir(dir1);

if (dir2) closedir(dir2);

return;

}

while ((entry1 = readdir(dir1)) != NULL) {

if (strcmp(entry1->d\_name, ".") != 0 && strcmp(entry1->d\_name, "..") != 0) {

char pathInDir1[PATH\_MAX];

snprintf(pathInDir1, sizeof(pathInDir1), "%s/%s", path1, entry1->d\_name);

while ((entry2 = readdir(dir2)) != NULL) {

if (strcmp(entry2->d\_name, ".") != 0 && strcmp(entry2->d\_name, "..") != 0) {

char pathInDir2[PATH\_MAX];

snprintf(pathInDir2, sizeof(pathInDir2), "%s/%s", path2, entry2->d\_name);

STAT buf1, buf2;

if (stat(pathInDir1, &buf1) != -1 && stat(pathInDir2, &buf2) != -1) {

if (S\_ISDIR(buf1.st\_mode) && S\_ISDIR(buf2.st\_mode)) {

searchAndPrintDuplicates(pathInDir1, pathInDir2, outputFile);

} else if (S\_ISREG(buf1.st\_mode) && S\_ISREG(buf2.st\_mode)) {

if (compareFiles(pathInDir1, pathInDir2)) {

fprintf(outputFile, "Path 1: %s\n", pathInDir1);

fprintf(outputFile, "Path 2: %s\n", pathInDir2);

fprintf(outputFile, "Size: %ld bytes\n", (long)buf1.st\_size);

struct tm\* creationTime = localtime(&buf1.st\_ctime);

char formattedTime[80];

strftime(formattedTime, sizeof(formattedTime), "%Y-%m-%d %H:%M:%S", creationTime);

fprintf(outputFile, "Creation Time: %s\n", formattedTime);

fprintf(outputFile, "Permissions: %o\n", buf1.st\_mode & 0777);

fprintf(outputFile, "Inode Number: %ld\n\n", (long)buf1.st\_ino);

}

}

}

}

}

rewinddir(dir2); // Вернуться в начало каталога 2

}

}

closedir(dir1);

closedir(dir2);

}

int main(int argc, char\* argv[]) {

if (argc != 4) {

fprintf(stderr, "Usage: %s <directory1> <directory2> <output\_file>\n", argv[0]);

return 1;

}

const char\* path1 = argv[1];

const char\* path2 = argv[2];

const char\* outputFilePath = argv[3];

FILE\* outputFile = fopen(outputFilePath, "w");

if (outputFile == NULL) {

fprintf(stderr, "Error opening output file: %s\n", outputFilePath);

return 1;

}

chmod(outputFilePath, 0666);

searchAndPrintDuplicates(path1, path2, outputFile);

fclose(outputFile);

return 0;

}