УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе № 7

по предмету «Архитектура компьютерной техники и операционных систем»

Вариант 4

Выполнил:

Сак К.А.

гр. 251002

Проверил:

Леванцевич В.А.

Минск 2023

**Условие:**

4. Для заданного каталога (аргумент 1 командной строки) и всех его подкаталогов вывести в заданный файл (аргумент 2 командной строки) и на консоль имена файлов, их размер и дату создания, удовлетворяющих заданным условиям: 1 – размер файла находится в заданных пределах от ***N1*** до ***N2*** (***N1,N2***задаются в аргументах командной строки), 2 – дата создания находится в заданных пределах от ***M1*** до ***M2*** (***M1***,***M2***задаются в аргументах командной строки).Процедура поиска для каждого подкаталога должна запускаться в отдельном процессе. Каждый процесс выводит на экран свой ***pid,*** полный путь, имя и размер просмотренного файла, общее число просмотренных файлов в подкаталоге. Число запущенных процессов в любой момент времени не должно превышать ***N*** (вводится пользователем).

**Код 7.1**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <string.h>**

**#include <dirent.h>**

**#include <sys/stat.h>**

**#include <sys/types.h>**

**#include <unistd.h>**

**#include <fcntl.h>**

**#include <time.h>**

**#define MAX\_PATH 1024**

**void process\_directory(char \*path, int minSize, int maxSize, time\_t minTime, time\_t maxTime,**

**FILE \*outputFile, int maxProcesses) {**

**DIR \*dir;**

**struct dirent \*entry;**

**struct stat fileStat;**

**char filePath[256];**

**int numProcesses = 0;**

**int filesNum = 0;**

**dir = opendir(path);**

**if (dir == NULL) {**

**printf("Error: could not open directory %s\n", path);**

**return;**

**}**

**while ((entry = readdir(dir)) != NULL) {**

**if (entry->d\_type == DT\_DIR)**

**{**

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

**{**

**snprintf(filePath, MAX\_PATH, "%s/%s", path, entry->d\_name);**

**pid\_t pid = fork();**

**if (pid == 0) {**

**process\_directory(filePath, minSize, maxSize, minTime, maxTime, outputFile, maxProcesses);**

**exit(0);**

**} else if (pid > 0) {**

**numProcesses++;**

**// printf("\n num processes before: %d in filePath: %s", numProcesses, filePath);**

**if (numProcesses >= maxProcesses) {**

**wait(NULL);**

**numProcesses--;**

**}**

**} else {**

**printf("Error: could not create child process\n");**

**}**

**}**

**}**

**else{**

**snprintf(filePath, MAX\_PATH, "%s/%s", path, entry->d\_name);**

**if (stat(filePath, &fileStat) == 0) {**

**filesNum++;**

**if(fileStat.st\_size >= minSize && fileStat.st\_size <= maxSize && fileStat.st\_ctime >= minTime && fileStat.st\_ctime <= maxTime){**

**fprintf(outputFile, "\nname: %s, size: %d, creation time: %s, pid: %d, path: %s, visited files: %d\n", entry->d\_name, fileStat.st\_size, ctime(&fileStat.st\_ctime), getpid(), path, filesNum);**

**printf("\nname: %s, size: %d, creation time: %s, pid: %d, path: %s, visited files: %d\n",entry->d\_name, fileStat.st\_size, ctime(&fileStat.st\_ctime), getpid(), path, filesNum);**

**}**

**} else{**

**printf("Error: could not get file information for %s\n", filePath);**

**}**

**}**

**}**

**closedir(dir);**

**}**

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

**if (argc != 4) {**

**printf("Usage: %s <directory> <outputFile> <maxProcesses>\n", argv[0]);**

**return 1;**

**}**

**char \*directory = argv[1];**

**char \*outputFileName = argv[2];**

**int maxProcesses = atoi(argv[3]);**

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

**if (outputFile == NULL) {**

**printf("Error: could not open output file\n");**

**return 1;**

**}**

**process\_directory(directory, 0, 1000000000, 0, time(NULL), outputFile,maxProcesses);**

**fclose(outputFile);**

**return 0;**

**}**

**Код 7.2**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <string.h>**

**#include <unistd.h>**

**#include <ctype.h>**

**#include <errno.h>**

**#include <time.h>**

**#include <linux/stat.h>**

**#include <linux/fcntl.h>**

**#include <sys/syscall.h>**

**#include <sys/types.h>**

**#include <sys/stat.h>**

**#include <sys/time.h>**

**#include <dirent.h>**

**#include <stdbool.h>**

**#include <pthread.h>**

**int min, max;**

**int maxThread, curThread = 1;**

**time\_t mint, maxt;**

**FILE\* outputFileName;**

**void\* getinfo(void\* param){**

**char filePath[256];**

**DIR\* dir;**

**struct dirent\* entry;**

**int filesNum = 0;**

**char curfile[256];**

**struct stat fileStat;**

**char newdir[256];**

**pthread\_t tid;**

**strcpy(filePath, (char\*) param);**

**dir = opendir(filePath);**

**if (dir == NULL) {**

**printf("Error: could not open directory %s\n", filePath);**

**return;**

**}**

**strcat(filePath, "/");**

**while ((entry = readdir(dir))!=NULL) {**

**memset(curfile, 0, sizeof(curfile));**

**strcpy(curfile, filePath);**

**strcat(curfile, entry->d\_name);**

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

**stat(curfile, &fileStat);**

**filesNum++;**

**if(fileStat.st\_size >= min && fileStat.st\_size <= max**

**&& fileStat.st\_ctime >= mint && fileStat.st\_ctime <= maxt){**

**printf("%lu:%s;\t",pthread\_self(), curfile);**

**fprintf(outputFileName,"%lu:%s;\t",pthread\_self(), curfile);**

**printf("%lu: size = %d bytes\t time: %s\n", pthread\_self(), (int)fileStat.st\_size, ctime(&fileStat.st\_ctime));**

**fprintf(outputFileName,"%lu: size = %d bytes\t time: %s\n", pthread\_self(), (int)fileStat.st\_size, ctime(&fileStat.st\_ctime));**

**}**

**}else if (entry->d\_type == DT\_DIR && entry->d\_name[0] != '.' && strcmp(entry->d\_name, "..") != 0){**

**if(curThread < maxThread){**

**strcpy(newdir, curfile);**

**curThread++;**

**pthread\_create(&tid, NULL, getinfo, (void\* )newdir);**

**printf("%lu: New thread, tid: %lu\n",pthread\_self(), tid);**

**fprintf(outputFileName,"%lu: Created new thread, tid: %lu\n",pthread\_self(), tid);**

**pthread\_join(tid, NULL);**

**}else{**

**getinfo(curfile);**

**}**

**}**

**}**

**printf("%lu: Number of files in %s : %d\n",pthread\_self(),filePath, filesNum);**

**fprintf(outputFileName,"%lu: Number of files in %s : %d\n",pthread\_self(),filePath, filesNum);**

**closedir(dir);**

**curThread--;**

**}**

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

**char \*directory = argv[1];**

**outputFileName = fopen(argv[2], "w");**

**min = atoi(argv[3]);**

**max = atoi(argv[4]);**

**mint = 0;**

**maxt = time(NULL);**

**maxThread = atoi(argv[5]);**

**getinfo(directory);**

**fclose(outputFileName);**

**return 0;**

**}**