БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

ИИТ БГУИР

Программное обеспечение информационных технологий.

Дисциплина: «Сетевые технологии и администрирование операционных систем»

Отчет

По лабораторной работе №4

«ПРОЦЕССЫ И ПОТОКИ В ОС LINUX»

Выполнил студент группы 481064 Сорока А.А.

Проверил преподаватель: Калитеня И. Л.

Минск 2016 г

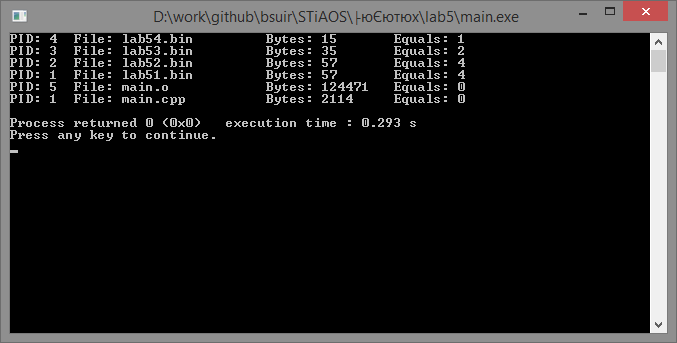
**Цель:**

1. Исследовать методы создания процессов в ОС Linux, основные функции создания и управления процессами, обмен данными между процессами.

**Задание:**

1. Написать программу поиска заданной пользователем комбинации из m байт (m < 255) во всех файлах текущего каталога. Пользователь задает имя каталога. Главный процесс открывает каталог и запускает для каждого файла каталога отдельный процесс поиска заданной комбинации из m байт. Каждый процесс выводит на экран свой pid, имя файла, число просмотренных байт и результаты поиска. Число запущенных процессов не должно превышать N (вводится пользователем).

**Скриншоты работающего приложения:**



**Код приложения:**

#include <stdio.h>

#include <vector>

#include <unistd.h>

#include <iostream>

#include <iomanip>

#include <fstream>

#include <string.h>

#include <sys/stat.h>

#include <sys/types.h>

#include <dirent.h>

#include <wait.h>

#include <map>

using namespace std;

bool ArrayEqual (char\* a, char\* b) {

for (int i = 0; i < sizeof a; i++)

{

if (a[i] != b[i]) return false;

}

return true;

}

void SeeBytes(string url) {

ifstream inputFile(url.c\_str(), ifstream::binary);

char arrayKey[4] = {'r','d','c','t'};

char arrayTemp[sizeof arrayKey];

inputFile.seekg(0,inputFile.end);

int length = inputFile.tellg();

int countEquals = 0;

for (int i = 0; i < length - sizeof arrayTemp + 1; i++) {

inputFile.seekg(i, inputFile.beg);

inputFile.read((char\*)&arrayTemp, sizeof arrayTemp);

countEquals += (int) ArrayEqual(arrayKey, arrayTemp);

}

inputFile.close();

cout << "PID: " << getpid() << '\t' << "File: " << url << '\t' << '\t' << '\t' << "Bytes: " << length << '\t' << "Equals: " << countEquals << endl;

}

int isFile(const char \*path)

{

struct stat path\_stat;

stat(path, &path\_stat);

return S\_ISREG(path\_stat.st\_mode);

}

void FF(int cur, vector<string> files, int deep) {

pid\_t pid;

int status;

if(cur < files.size()) {

if(deep == 0) {

SeeBytes(files[cur]);

} else {

switch (pid = fork()) {

case 0:

FF(cur + 1, files,deep - 1);

exit(42);

break;

default:

waitpid(pid, &status, WSTOPPED);

SeeBytes(files[cur]);

}

}

}

}

int main()

{

DIR \*dir;

struct dirent \*entry;

struct stat statbuf;

string url = "d:\\work\\github\\bsuir\\STiAOS\\Готовое\\lab4";

vector<string> files;

dir = opendir(url.c\_str());

int deepMax = 3;

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

if (isFile(entry->d\_name)) {

files.push\_back(entry->d\_name);

}

};

closedir(dir);

int current = 0;

int maxDeep = 2;

while(current < files.size()) {

FF(current, files, maxDeep);

current +=maxDeep + 1;

}

return 0;

}