Федеральное государственное бюджетное образовательное учреждение высшего образования

Национально исследовательский университет «Московский энергетический институт»

Кафедра ВМСС

Лабораторная работа №6

по курсу

«Программное обеспечение высокопроизводительных вычислительных систем»

Выполнили: студенты группы А-07м-23

Балашов С.А., Кретов Н.В., Михайлов С.А.

Проверил: Орлов Д.А., Хиль С.

Москва, 2024 г.

**Задание**

1. Переписать программу из ЛР1, работающую с разделяемой памятью и семафорами, используя POSIX Threads. Для синхронизации нитей использовать pthread\_mutex.

Код программы:

#include <pthread.h>  
#include <unistd.h>  
#include <fcntl.h>  
#include <cstring>  
#include <cstdio>  
#include <cstdlib>  
  
#define BLOCK\_SIZE 32  
  
pthread\_mutex\_t memGuard;  
char \*mem = nullptr;  
pthread\_cond\_t dataReady;  
bool isDataReady = false;  
  
void\* child(void\* arg){  
 while (true)  
 {  
 while (!isDataReady) // Wait until data is ready  
 {  
 pthread\_cond\_wait(&dataReady, &memGuard);  
 }  
 pthread\_mutex\_lock(&memGuard);  
 ssize\_t nbytes = mem[BLOCK\_SIZE\*sizeof(char)];  
 if (nbytes > 0)  
 {  
 write(STDOUT\_FILENO, mem, nbytes);  
 }  
 else  
 {  
 pthread\_mutex\_unlock(&memGuard);  
 break;  
 }  
 pthread\_mutex\_unlock(&memGuard);  
 usleep(11);  
 }  
 return nullptr;  
}  
  
void\* parent(void\* arg){  
 int file;  
 file = open("../file.txt", O\_RDONLY);  
 if (file == -1) {  
 perror("open");  
 exit(1);  
 }  
  
 while (true)  
 {  
 pthread\_mutex\_lock(&memGuard);  
 ssize\_t nbytes = read(file, mem, BLOCK\_SIZE);  
 mem[BLOCK\_SIZE\*sizeof(char)] = nbytes;  
 isDataReady = true;  
 pthread\_cond\_signal(&dataReady);  
 pthread\_mutex\_unlock(&memGuard);  
 usleep(10);  
  
 if (nbytes == 0)  
 {  
 close(file);  
 break;  
 }  
 }  
 return nullptr;  
}  
  
int main ()  
{  
 pthread\_t childThread, parentThread;  
 pthread\_mutex\_init(&memGuard, nullptr);  
 mem = (char\*)malloc(BLOCK\_SIZE);  
  
 pthread\_create(&childThread, nullptr, &child, nullptr);  
 pthread\_create(&parentThread, nullptr, &parent, nullptr);  
  
 pthread\_join(parentThread, nullptr);  
 pthread\_join(childThread, nullptr);  
  
 pthread\_mutex\_destroy(&memGuard);  
 free(mem);  
  
 return 0;  
}

1. Переписать программу из ЛР1, работающую с разделяемой памятью и семафорами, используя std::thread и std::mutex. Допускается использование атомарных переменных.

Код программы:

#include <thread>  
#include <mutex>  
#include <atomic>  
#include <vector>  
#include <string>  
#include <fcntl.h>  
#include <unistd.h>  
  
static constexpr int bufSize = 32;  
  
std::vector<char> buf(bufSize);  
std::atomic<bool> dataReady{false};  
std::mutex memGuard;  
  
void child() {  
 while (true) {  
 while (!dataReady) {  
 std::this\_thread::yield();  
 }  
 std::unique\_lock<std::mutex> lock(memGuard);  
 ssize\_t nbytes = buf[bufSize\*sizeof(char)];  
 if (nbytes > 0) {  
 write(1, buf.data(), nbytes);  
 } else {  
 lock.unlock();  
 break;  
 }  
 lock.unlock();  
 std::this\_thread::sleep\_for(std::chrono::nanoseconds(11));  
 }  
}  
  
void parent() {  
 int file;  
 file = open("../file.txt", O\_RDONLY);  
 if (file == -1) {  
 perror("open");  
 exit(1);  
 }  
  
 while (true) {  
 std::unique\_lock<std::mutex> lock(memGuard);  
 ssize\_t nbytes = read(file, buf.data(), bufSize);  
 buf[bufSize\*sizeof(char)] = nbytes;  
 dataReady = true;  
 lock.unlock();  
  
 if (nbytes == 0) {  
 close(file);  
 break;  
 }  
 std::this\_thread::sleep\_for(std::chrono::nanoseconds(10));  
 }  
}  
  
int main() {  
 std::thread childThread(child);  
 std::thread parentThread(parent);  
  
 parentThread.join();  
 childThread.join();  
  
 return 0;  
}