

Homework 1

This assignment must be done individually

Date Due: Feb 6, 2023

The objectives of this assignment are the following: Acquire familiarity with using POSIX thread management primitives. How to write POSIX thread programs and communicate through shared memory. Read the manual pages for the LINUX primitives for creating shared memory between two LINUX processes. These functions are -- shmget and shmat. For sample programs on how to use the POSIX thread library and its shared memory management, you can use Google to search for "POSIX Thread Programming Examples".

Problem Statement:

In this assignment you will write a POSIX program. Consider a system comprising of three producer threads, one consumer thread, and a shared buffer of size 2. Each producer thread **randomly (over time)** creates a specific colored item: producer_red creates red colored items, producer_black creates black colored items, and producer_white creates white-colored items. Associated with each item created by a specific producer is a local LINUX timestamp in microseconds (use the gettimeofday() function) which records the time that item was placed into the buffer. Each producer deposits its item into the buffer, and the consumer retrieves the items from the buffer. The items produced by the producers will be of any of the three strings below:

“RED timestamp”, “BLACK timestamp”, “WHITE timestamp”

Each producer threads, after successfully depositing an item into the buffer, will write that item (essentially the “COLOR timestamp” string) into its log file called Producer_COLOR.txt (e.g. Producer_RED.txt). The consumer thread will then retrieve the items from this shared buffer. When it retrieves an item, the consumer thread will write that item (essentially the “COLOR timestamp” string) into a file called Consumer.txt.

Requirements for the program:

1. Each producer thread will produce its respective colored item (represented as a string “COLOR timestamp”). If the producer thread finds the shared buffer has space, it will first create a timestamp for that item. The producer thread will then write the item into its log file (Producer_COLOR.txt) and also deposit the item into the buffer. If the buffer does not have space, the producer will have to wait. Each string (or item) written into the log file will be on a new line.
2. The consumer thread will take an item from the buffer, and write its contents to Consumer.txt file. Each string should be written in the order it was read from the buffer,

and in a new line. If the buffer was initially full, the consumer thread should signal any one of the producer threads.

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

3. Each producer thread should produce 1000 items. Hence at the end of running the program, the consumer should have written 3000 items into its log file.

Extra Guidelines:

- One of the objectives of this assignment is to implement counting semaphores using POSIX mutex and condition variables. Hence, you *cannot* use **semaphore.h** (which is part of the 3RT library) or any other counting semaphore library.

You can only use the following .h files:

```
#include<pthread.h>
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include<unistd.h>
#include <sys/time.h> //for gettimeofday()
```

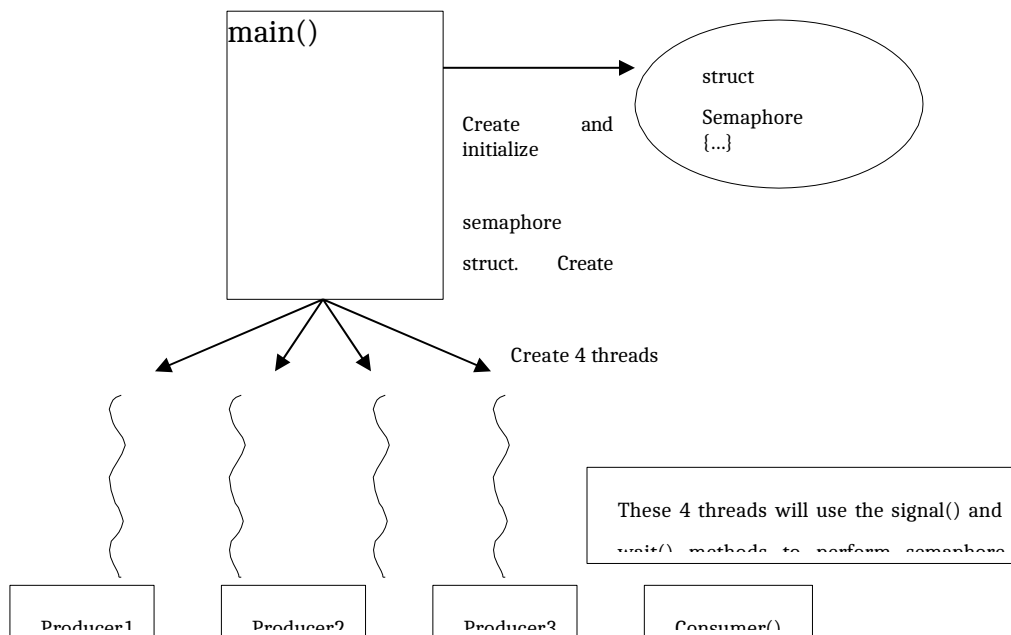
To do tasks: Write the code for this system as a single LINUX process containing four POSIX threads. The functions of each producer and consumer should be performed by one separate POSIX thread.

Assignment Project Exam Help

<https://tutorcs.com>

Recipe:

WeChat: cstutorcs



- 1.) In your program, define a semaphore struct (which would obviously have a counter, a pthread_mutex, and a pthread_condition variable).
- 2.) Define signal() and wait() methods which will be used on the semaphores.
- 3.) Your main() program can create and initialize the required semaphores. It will then spawn 4 threads, which will perform their functions as specified in the homework. The threads will obviously have to use the implemented signal() and wait() methods to perform semaphore operations.

Items to be submitted in a LINUX tar file:

- (1) Code for your program. This should be in a file named prod_cons.c.
- (2) A makefile used to compile your code
- (3) A Readme.txt file with instructions on how to run your program

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs