Systemy operacyjne

Pracownia programistyczna 2

Do każdego zadania należy dostarczyć test, który będzie badał poprawność implementacji. Zawsze implementuj najbardziej złośliwy test jaki jesteś w stanie wymyśleć.

Pełna dokumentacja dotycząca mechanizmów synchronizacji i komunikacji jest dostępna po zainstalowaniu pakietu `manpages-posix-dev`. Użyteczne strony podręcznika to:

- `unix(7)`
- `mq_overview(7)`, `sem_overview(7)`, `shm_overview(7)`
- `pthread_mutex_lock(P)`
- `pthread_cond_wait(P)`, `pthread_cond_signal(P)`
- `pthread_barrier_wait(P)`

Pożyteczne odnośniki:

1. Podstawy obsługi edytora Vim
2. Szybkie wprowadzenie do GNU Make
3. Krótki opis komend debuggera GDB

Zadanie 1

Napisz prosty komunikator dla dwóch osób. Użyj do tego celu nazwanych gniazdek domeny UNIX. Otwórz gniazdko z uprawnieniami pozwalającymi innemu użytkownikowi na podłączenie. Wysyłaj i odbieraj komunikaty po naciśnięciu klawisza `ENTER`.

Zadanie 1+

Wykorzystując bibliotekę `ncurses` podziel ekran na dwie części - górną, w której widać przebieg rozmowy, i dolną, w której jest to co aktualnie piszesz. Aby móc jednocześnie pisać i odbierać komunikaty od drugiego użytkownika użyj wywołania `poll` lub `select`.

Zadanie 2

Jednym z klasycznych problemów dot. synchronizacji jest problem czytelników i pisarzy. Rozwiąż ten problem używając semaforów binarnych dla wątków (`pthread_mutex`). Czy w Twoim rozwiązaniu proces może zostać zagłodzony?

Zadanie 3

Korzystając z semaforów `POSIX` zaimplementuj barierę dla procesów z trzema operacjami `init`, `wait` i `destroy`. Po przejściu procesów przez barierę musi się ona nadawać do ponownego użycia (tj. ma się zachowywać tak jak bezpośrednio po wywołaniu funkcji `init`).

Zadanie 4

Pokaż, że przekazywanie komunikatów i semafory wraz z pamiecią dzielona dostarczają podobnej mocy wyrazu. W tym celu zaimplementuj prosty interfejs:

- skrzynek pocztowych używając semaforów (Wskazówka: Użyj pamięci dzielonej jako bufora przechowującego N komunikatów o pewnej ustalonej długości)
- semaforów zliczających używając skrzynek pocztowych (Wskazówka: Użyj osobnego procesu jako gwaranta synchronizacji)

Zadanie 5

Korzystając z blokad (`pthread_mutex`) i zmiennych warunkowych (`pthread_cond`) zaimplementuj problem konsumentów i producentów dla ograniczonego bufora długości `N`. Czy da się zaimplementować w/w monitor na dwa sposoby tj. Hoare i Mesa przy pomocy mechanizmów `POSIX Threads`?

Zadanie 6

Wyobraź sobie restaurację ramen z pięcioma siedzeniami. Jeśli pojawisz się w restauracji, kiedy jedno siedzenie jest puste, możesz od razu je zająć. Jednakże jeśli wszystkich pięć siedzeń jest zajętych, musisz poczekać aż wszystkie pięć osób zje swoje ramen i opuści restaurację, aby zasiąść do stołu. Napisz program implementujący klientów restauracji ramen spełniający powyższe wymagania. Każdy klient musi być procesem. Użyj semaforów `POSIX`.

Zadanie 6+

Zaimplementuj problem restauracji ramen przy pomocy kolejek komunikatów `POSIX`.

Zadanie 7

Trzy rodzaje wątków dzielą dostęp do listy jednokierunkowej:

- przeszukujące: wyłącznie przeglądają listę, zatem każdy z nich może wykonywać tą operację współbieżnie względem innych przeszukujących,
- wkładające: dodają nowy element na koniec listy, zatem muszą unikać sytuacji, w której będą modyfikować ostatni element listy równocześnie; operacja dodawania elementu może wykonywać się bez przeszkód z współbieżnymi operacjami przeszukiwania,
- usuwające: mogą usunąć dowolny element listy, zatem należy wstrzymać pozostałe operacje na liście.

Zaimplementuj kontener listy jednokierunkowej wraz z trzema operacjami, które będą spełniać wyżej wymienione założenia.