Implementacje monitorów

Monitor jest strukturalnym narzędziem synchronizacji wątków. W językach obiektowych (c++,java) implementuje się go jako klasę. Wszystkie metody monitora muszą być wykonywane z zachowaniem wzajemnego wykluczania. Oznacza to, że w klasie reprezentującej monitor musi istnieć jeden klucz dostępu do wszystkich metod. Korzystając z biblioteki pthread można użyć w tym celu wspólnego mutexu dla ochrony wszystkich metod.

ćwiczenie 1
a) Zapisać w pseudokodzie implementacje semafora przy pomocy monitora. Monitor powinien implementować metody wait i signal
b) Zaimplementować ten monitor w języku C++ wykorzystując funkcje z biblioteki pthread. Sprawdzić jego działanie na przykładzie programu *p1.c*.
c) Zaimplementować ten monitor w języku Java. Sprawdzić jego działanie na przykładzie programu p1.java.

ćwiczenie 2
Bariera jest narzędziem, które służy do wstrzymania grupy wątków w określonym miejscu programu współbieżnego. Wątki zostają wstrzymane w momencie wywołania operacji *waitb* na barierze do czasu, aż liczba wstrzymanych wątków nie osiągnie pewnej ustalonej wartości progowej. Wartość progowa jest parametrem charakterystycznym dla danej bariery. Barierę wykorzystuje się w obliczeniach wieloetapowych, gdzie wszystkie wątki z grupy muszą rozpoczynać każdy nowy etap obliczeń w tym samym czasie. Przykładowy program zapisany w pseudokodzie ilustruje wykorzystanie bariery.

Bariera bar ← 2
a,b,x,y,u,w

p1: a ← 2	*q1: b ← 1*
waitb (bar)	***waitb (bar)***
p2: x ← a+b	*q2: y ← a-b*
waitb (bar)	***waitb (bar)***
*p3: u ← x*y*	*q3: w ← x*y*

a) Zapisać w pseudokodzie implementacje bariery przy pomocy monitora. Monitor powinien i definiować operację *waitb*. Wątek wywołujący metodę *waitb* ma zostać wstrzymany w przypadku, gdy liczba wstrzymanych wątków będzie mniejsza od pewnej wartości progowej zdefiniowanej jako próg bariery. Gdy liczba wątków wywołujących metodę czeka osiągnie próg bariery wtedy ostatni z tych wątków nie będzie wstrzymany tylko spowoduje wznowienie pozostałych wątków.
b) Zaimplementować ten monitor w języku C++ wykorzystując funkcje z biblioteki pthread.

c) Wykorzystać ten monitor do synchronizacji programu w przykładzie *p2.c*.
Poniżej jest przedstawiony pseudokod analogicznego programu:

tablica t

p1:powtarzaj	*q1:powatrzaj*	*r1:powatrzaj*
p2: t[0] ← rand	*q2: t[1] ← rand*	*r2: wypisz (t[0]+t[1])/2*

Zapisać w pseudokodzie synchronizację tego programu.

Zadania dodatkowe

1. Napisać w języku C++ implementacje monitora producent-konsument na podstawie pseudokodu w pliku „*wykład3.pdf*” wykorzystując narzędzia synchronizacji z biblioteki pthread.