Opis zadania:

Program uruchamia się z poziomu klasy Main i metody main. Program składa się z trzech wątków WatekA, WatekB, i WatekC oraz interfejsu Synchronizator. Klasy Main, WatekA, WatekB, WatekC oraz interfejsu Synchronizator nie wolno modyfikować. Synchronizator odpowiedzialny jest za obsługę sekcji krytycznej dla każdego z wątków, tak że dla każdego z wątków ma bezargumentowe metody startA, koniecA, startB, koniecB, startC i koniecC.

Wątki w nieskończonej pętli wykonują swoje zadania – patrz klasy implementujące WatekA, WatekB, WatekC.

Twoim zadaniem jest stworzenie implementacji interfejsy Synchronizator w postaci klasy SynchronizatorImpl, tak aby zapewnić, że zadanie związane z Watkiem A zostanie wykonane 3 razy, a wówczas pozostałe wątki nie mogą wykonać swojej sekcji krytycznej. Po trzykrotnym wykonaniu zadania realizowanego przez watekA, powinno nastąpić dwukrotne wykonanie wątkuB, tak, że pozostałe wątki nie wykonują swojej sekcji krytycznej, a na koniec wątekB powinien odblokować WątekC, który powinien zostać wykonany tylko raz. Pozostałe wątki powinny wówczas czekać. Koniec wątku C powinien odblokować wątek A znów trzykrotnie.

W rezultacie na ekranie powinno się wyświetlić:

Synchronizowany kod A
Synchronizowany kod A
Synchronizowany kod A
Synchronizowany kod B
Synchronizowany kod B
Synchronizowany kod C
Synchronizowany kod A
Synchronizowany kod A
Synchronizowany kod A
Synchronizowany kod B
Synchronizowany kod B
Synchronizowany kod C

UWAGA: Zadanie wykonaj z wykorzystaniem semaforów