

Zadanie 1.

- 1) Napisać program współbieżny symulujący działanie m producentów i n konsumentów komunikujących się przez ograniczony bufor cykliczny.
- 2) Działanie wątku producenta polega na wielokrotnym (zadana liczba powtórzeń) wykonywaniu po sobie ciągów instrukcji odpowiadających tzw. *produkcji danych* oraz synchronizowanego wstawienia ich do buforu. Produkcja danych polegać ma na wstrzymaniu wątku przez losowy czas w przedziale <1, 10> milisekund oraz wylosowaniu liczby całkowitej z przedziału <0, 99>. Dana jest typu string i ma następującą postać:

Dana=[*id producenta, nr powt., wartość*]

Dana=[P-1, 100, 88]

- 3) Działanie wątku konsumenta polega na wielokrotnym (zadana liczba powtórzeń) wykonywaniu po sobie synchronizowanego pobrania danych z buforu oraz tzw. *konsumpcji danych*. Konsumpcja danych polegać ma na wstrzymaniu wątku przez losowy czas w przedziale <2, 12> milisekund oraz wypisania stosownego komunikatu.
- 4) Komunikat powinien mieć postać:

[*id konsumenta, nr powt.*] >> Dana=[*id producenta, nr powt., wartość*]

[K-2, 33] >> Dana=[P-1, 100, 88]

- 5) W celu synchronizacji procesów wykorzystać mechanizm monitora procesów opierając się na zamkach jawnych (ReentrantLock) i zmiennych warunkowych z wykorzystaniem interfejsu Condition.
- 6) Działanie programu zademonstrować dla m=4, n=5.

Zadanie 2.

- 1) Napisać program współbieżny symulujący działanie m czytelników i n pisarzy korzystających ze współdzielonej czytelni.
- 2) Działanie wątku czytelnika oraz pisarza polega na wielokrotnym (zadana liczba powtórzeń) wykonywaniu po sobie ciągów instrukcji odpowiadających tzw. *sprawom własnym* oraz synchronizowanemu korzystaniu z czytelni. Sprawy własne polegać mają na wstrzymaniu wątku przez losowy czas w przedziale <5, 15> milisekund.
- 3) Synchronizowane korzystanie z czytelni dla wątków czytelników i pisarzy objawia się wstrzymaniem wątku przez losowy czas w przedziale <1, 5> milisekund oraz wypisaniem stosownych komunikatów.
- 4) Komunikaty powinny być wypisane na początku i końcu każdej z procedur synchronizacji korzystania z czytelni (po dwa komunikaty na rozpoczęcie i zakończenie korzystania z czytelni).
- 5) Komunikat powinien mieć postać (początkowe symbole oznaczają: „>>> (1)” „>>> (2)” – przed, „<<< (1)” „<<< (2)” – po):

>>> (1) [*id wątku, nr powt.*] :: [*licz_czyt=1, licz_czyt_pocz=1, licz_pis=0, licz_pis_pocz=0*]

>>> (1) [C-2, 33] :: [*licz_czyt=1, licz_czyt_pocz=1, licz_pis=0, licz_pis_pocz=0*]

- 6) Uwaga: znaki „>>>” „<<<” stosować dla czytelnika, natomiast znaki „==>” „<==” dla pisarza.
- 7) W celu synchronizacji procesów wykorzystać mechanizm monitora procesów opierając się na zamkach jawnych (ReentrantLock) i zmiennych warunkowych z wykorzystaniem interfejsu Condition.
- 8) Działanie programu zademonstrować dla m=4, n=2.

Zadanie 3.

- 1) Napisać program współbieżny demonstrujący rozwiązanie problemu uczujących (pięciu) filozofów.
- 2) Działanie wątku filozofa polega na wielokrotnym (zadana liczba powtórzeń) wykonywaniu po sobie ciągów instrukcji odpowiadających tzw. *rozmyślaniu* oraz synchronizowanemu korzystaniu z przypisanych widelców. Rozmyślanie polegać ma na wstrzymaniu wątku przez losowy czas w przedziale <5, 15> milisekund.
- 3) Synchronizowane korzystanie z widelców objawia się wstrzymaniem wątku przez losowy czas w przedziale <1, 5> milisekund oraz wypisaniem stosownych komunikatów przed rozpoczęciem i po zakończeniu posiłku.
- 4) Komunikaty powinny być wypisane na początku i końcu każdej z procedur synchronizacji korzystania z widelców (po dwa komunikaty na rozpoczęcie i zakończenie korzystania z widelców).

5) Komunikat powinien mieć postać (początkowe symbole oznaczają: „>>> (1)” „>>> (2)” – przed, „<<< (1)” „<<< (2)” – po):

>>> (1) [id wątku, nr powt.] :: [stan wid. 0, stan wid. 1, stan wid. 2, stan wid. 3, stan wid. 4] – licz. fil. przy stole

>>> (1) [F-1, 33] :: [0, 0, 1, 1, 1] – 1

6) W celu synchronizacji procesów wykorzystać mechanizm monitora procesów opierając się na zamkach jawnych (ReentrantLock) i zmiennych warunkowych z wykorzystaniem interfejsu Condition.