Semafory

1. Napisać program, w którym, oprócz wątku głównego, działają dwa wątki robocze. Zadaniem pierwszego jest pobranie od użytkownika dwóch liczb całkowitych i umieszczenie ich w tablicy współdzielonej z drugim wątkiem. Gdy liczby zostaną podane, wątek *niezwłocznie powiadamia* drugi wątek. Drugi wątek po uruchomieniu *czeka na sygnał* od pierwszego wątku, a następnie wyświetla iloczyn podanych liczb.

   Do synchronizacji między wątkami należy zastosować semafor.

   Przykładowy kod uruchamiający wątki.

   ```
   Semaphore gotowe = new Semaphore(0);
   int [] liczby = new int [2];
   Czytajacy watek1 = new Czytajacy(liczby, gotowe);
   Liczacy watek2 = new Liczacy(liczby, gotowe);
   watek1.start();
   watek2.start();
   watek1.join();
   watek2.join();
   ```

2. Napisać program symulujący korzystanie z parkingu samochodowego. Zarówno wjazd, jak i wyjazd z parkingu zabezpieczone są szlabanami podnoszonymi po naciśnięciu przycisku. Szlaban wjazdowy jest podnoszony pod warunkiem, że na parkingu są wolne miejsca. W przeciwnym razie kierowcy sygnalizowane jest zapełnienie parkingu i musi on czekać na zwolnienie miejsca.

   Sterowaniem pojedynczym samochodem zawarte jest w klasie `Samochod`. Każdy samochód działa w osobnym wątku. Samochód wykonuje cyklicznie następujące czynności (określoną w konstruktorze liczbę razy):

   - wjeżdża na parking,
   - stoi na parkingu (1-3 sek.),
   - wyjeżdża z parkingu,
   - pewien czas jeździ poza parkingiem (1-3 sek.).

   Kontrolą miejsc parkingowych zajmuje się klasa `Parking`, która ma następujące metody publiczne:

   - `void wjedz()` - metoda kończy się, gdy na parkingu jest wolne co najmniej jedno miejsce, w przeciwnym razie metoda powoduje wstrzymanie wątku, który ją wywołał;

- `void wyjedz()` - metoda aktualizuje stan wolnych miejsc na parkingu, wykonanie kończy się natychmiast.

Liczba miejsc parkingowych wynosi 5, liczba samochodów wynosi 10. Przy próbie wjazdu oraz przy wyjeździe z parkingu klasa `Samochod` wyświetla komunikaty informacyjne. Podobnie klasa `Parking` działająca w odrębnym wątku co kilka sekund wyświetla informacje o liczbie wolnych miejsc parkingowych.

Wskazówka: w klasie `Parking` zastosować semafor.

3. W problemie Producenta-Konsumenta modelowany jest układ dwóch obiektów:
   - obiekt klasy Producent „produkuje" liczby, które umieszcza w buforze o ograniczonej pojemności;
   - obiekt klasy Konsument „konsumuje" liczby, które w buforze umieścił producent;
   - obiekt bufora jest, oczywiście, współdzielony między producentem a konsumentem, natomiast producent nie ma bezpośredniego dostępu do konsumenta i na odwrót.

   Zaimplementuj niezbędne klasy (Producent, Konsument, Bufor) i wykonaj symulację, w której producent produkuje ciąg $N = 10^i$, $i \in \{1, 2, 3, 5\}$ losowych liczb całkowitych z zakresu [1, 10]. Liczby te są pobierane przez konsumenta. Pojemność bufora jest ograniczona do 5. Zarówno producent, jak i konsument powinni wypisywać informacje o każdej wyprodukowanej, czy skonsumowanej liczbie.

   Do synchronizacji kooperacji między wątkami należy zastosować mechanizm **semaforów**. Należy również pamiętać o konieczności synchronizacji *operacji na buforze za pomocą blokad.*

   W celu dodatkowej weryfikacji poprawności działania programu należy dodać obliczanie sumy kontrolnej przez producenta dla wyprodukowanych, a przez konsumenta dla odebranych liczb. Oczywiście, sumy te powinny się zgadzać.

   Przykład:

```
Producent: produkuję 6
Producent: produkuję 3
Producent: produkuję 5
Konsument: otrzymałem 6
Producent: produkuję 4
Konsument: otrzymałem 3
Producent: produkuję 4
Konsument: otrzymałem 5
```

4. Plemię tubylców ucztuje zebrane wokół dużego kotła zawierającego m porcji mięsa z upolowanej gazeli. Gdy jakiś tubylec ma ochotę się poczęstować, to sięga do kotła po porcję. Jeżeli kocioł jest pusty, to tubylec budzi kucharza, który ponownie napełnia kocioł. Każdy tubylec zaspokoi swój głód po zjedzeniu k porcji.

Napisać program współbieżny złożony z n procesów *Tubylec oraz procesu* Kucharz symulujący opisany scenariusz. Kucharz powinien być budzony jedynie wtedy, gdy kocioł jest pusty.

Program przetestować dla wartości $n = 3$, $m = 10$, $k = 5$ oraz dla zestawu większych wartości.

Przykładowy wydruk:

```
Tubylec 1, zjadłem 1. sztukę
Tubylec 3, zjadłem 1. sztukę
Tubylec 2, zjadłem 1. sztukę
Tubylec 1, zjadłem 2. sztukę
Tubylec 3, zjadłem 2. sztukę
Tubylec 2, zjadłem 2. sztukę
Tubylec 1, zjadłem 3. sztukę
Tubylec 3, zjadłem 3. sztukę
Tubylec 2, zjadłem 3. sztukę
Tubylec 1, zjadłem 4. sztukę
Kucharz: napełniam kocioł
Napełniono kocioł 10 sztukami mięsa
Tubylec 3, zjadłem 4. sztukę
Tubylec 2, zjadłem 4. sztukę
Tubylec 1, zjadłem 5. sztukę
Tubylec 3, zjadłem 5. sztukę
Tubylec 2, zjadłem 5. sztukę
Uczta zakończona
```

5. Wariant zadania producent-konsument. W systemie dany jest jeden wątek *producenta oraz n wątków* konsumentów. Producent produkuje liczby całkowite i wstawia je do bufora o rozmiarze b, z którego są pobierane przez konsumentów. Łącznie produkowane jest m liczb. Napisać program symulujący współpracę producent–konsumenci przy założeniach:

   - każda wiadomość wysłana przez producenta musi zostać odczytana przez wszystkich konsumentów
   - każdy konsument musi odebrać wiadomości w kolejności w jakiej zostały wysłane.

   Program przetestować dla różnych wartości n, b oraz m, przykładowo $n = 4$, $b = 3$, $m = 10$.

   Podobnie jak w zadaniu pierwszym każdy konsument powinien wyznaczać sumę kontrolną otrzymywanych liczb.