Zadanie 1.

1) Zdefiniować klasę wątku o nazwie `Dekker` demonstrującego działanie algorytmu Dekkera.
2) Zdefiniować dwie metody o nazwach `dzialanieSynchr()` i `dzialanieNiesynchr()`, które realizują algorytm z dostępem do sekcji krytycznej odpowiednio bez synchronizacji (niepoprawnie) oraz z synchronizacją.
3) Odpowiednia metoda wybierana jest w ramach inicjalizacji klasy (ustawiany odpowiedni atrybut).
4) Działanie wątku polega na wielokrotnym (zadana liczba powtórzeń) wykonywaniu po sobie tzw. spraw własnych oraz sekcji krytycznej. Sprawy własne polegać mają na wstrzymaniu wątku przez losowy czas w przedziale <1, 10> milisekund.
5) W celu realizacji algorytmu synchronizowanego zdefiniować w ramach klasy:
   a) zmienną identyfikującą wątek o nazwie `nr` typu `int`.
   b) tablicę dwuelementową wartości typu `boolean` o nazwie `chce`.
   c) zmienną `czyjaKolej` typu `int`.
6) W celu zademonstrowania niepoprawnych zjawisk związanych z niesynchronizowanym wykonywaniem wyłącznego kodu w sekcji krytycznej, należy w ramach sekcji wypisać następujący komunikat:

```
Sekcja krytyczna wątku: Dekker-1, nr powt.= 3
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

lub

```
Sekcja krytyczna wątku: Dekker-2, nr powt.= 3
----------------------------------------------------------------------
```

7) Ciąg znaków w drugiej linii powstaje poprzez 100-krotne wypisanie pojedynczego znaku.
8) Każdy z wątków ma swój własny znak `'+'` lub `'-'` dostępny przez tablicę `znaki` i numer wątku.
9) W celu zademonstrowania programu zdefiniować klasę uruchomieniową `Test`.

Zadanie 2.

1) Zdefiniować klasę wątku o nazwie `Peterson` demonstrującego działanie algorytmu Petersona.
2) Zdefiniować dwie metody o nazwach `dzialanieSynchr()` i `dzialanieNiesynchr()`, które realizują algorytm z dostępem do sekcji krytycznej odpowiednio bez synchronizacji (niepoprawnie) oraz z synchronizacją.
3) Odpowiednia metoda wybierana jest w ramach inicjalizacji klasy (ustawiany odpowiedni atrybut).
4) Działanie wątku polega na wielokrotnym (zadana liczba powtórzeń) wykonywaniu po sobie tzw. spraw własnych oraz sekcji krytycznej. Sprawy własne polegać mają na wstrzymaniu wątku przez losowy czas w przedziale <1, 10> milisekund.
5) W celu realizacji algorytmu synchronizowanego zdefiniować w ramach klasy:
   a) zmienną identyfikującą wątek o nazwie `nr` typu `int`.
   b) tablicę dwuelementową wartości typu `boolean` o nazwie `chce`.
   c) zmienną `czyjaKolej` typu `int`.
6) W celu zademonstrowania niepoprawnych zjawisk związanych z niesynchronizowanym wykonywaniem wyłącznego kodu w sekcji krytycznej, należy w ramach sekcji wypisać następujący komunikat:

```
Sekcja krytyczna wątku: Peterson-1, nr powt.= 3
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

lub

```
Sekcja krytyczna wątku: Peterson-2, nr powt.= 3
----------------------------------------------------------------------
```

7) Ciąg znaków w drugiej linii powstaje poprzez 100-krotne wypisanie pojedynczego znaku.
8) Każdy z wątków ma swój własny znak `'+'` lub `'-'` dostępny przez tablicę `znaki` i numer wątku.
9) W celu zademonstrowania programu zdefiniować klasę uruchomieniową `Test`.

Zadanie 3.

1) Zdefiniować klasę wątku o nazwie `Lamport` demonstrującego działanie algorytmu Lamporta.
2) Zdefiniować dwie metody o nazwach `dzialanieSynchr()` i `dzialanieNiesynchr()`, które realizują algorytm z dostępem do sekcji krytycznej odpowiednio bez synchronizacji (niepoprawnie) oraz z synchronizacją.
3) Odpowiednia metoda wybierana jest w ramach inicjalizacji klasy (ustawiany odpowiedni atrybut).
4) Działanie wątku polega na wielokrotnym (zadana liczba powtórzeń) wykonywaniu po sobie tzw. spraw własnych oraz sekcji krytycznej. Sprawy własne polegać mają na wstrzymaniu wątku przez losowy czas w przedziale <1, 10> milisekund.
5) W celu realizacji algorytmu synchronizowanego zdefiniować w ramach klasy:
   a) zmienną identyfikującą wątek o nazwie `nr` typu `int`.
   b) tablicę wartości typu `boolean` o nazwie `wybieranie`.
   c) tablicę wartości typu `int` o nazwie `numerek`.
6) W celu zademonstrowania niepoprawnych zjawisk związanych z niesynchronizowanym wykonywaniem wyłącznego kodu w sekcji krytycznej, należy w ramach sekcji wypisać następujący komunikat:

```
Sekcja krytyczna wątku: Lamport-1, nr powt.= 3
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

   lub

```
Sekcja krytyczna wątku: Lamport-2, nr powt.= 3
------------------------------------------------------------------------
```

7) Ciąg znaków w drugiej linii powstaje poprzez 100-krotne wypisanie pojedynczego znaku.
8) Każdy z wątków ma swój własny znak dostępny przez tablicę `znaki` i numer wątku.
9) W celu zademonstrowania programu zdefiniować klasę uruchomieniową `Test` i uruchomić 5 wątków klasy `Lamport`.

Zadanie 4.

1) Zdefiniować klasę wątku o nazwie `Sem` demonstrującego działanie algorytmu wzajemnego wykluczania z wykorzystaniem semafora (klasa Semaphore).
2) Zdefiniować dwie metody o nazwach `dzialanieSynchr()` i `dzialanieNiesynchr()`, które realizują algorytm z dostępem do sekcji krytycznej odpowiednio bez synchronizacji (niepoprawnie) oraz z synchronizacją.
3) Odpowiednia metoda wybierana jest w ramach inicjalizacji klasy (ustawiany odpowiedni atrybut).
4) Działanie wątku polega na wielokrotnym (zadana liczba powtórzeń) wykonywaniu po sobie tzw. spraw własnych oraz sekcji krytycznej. Sprawy własne polegać mają na wstrzymaniu wątku przez losowy czas w przedziale <1, 10> milisekund.
   a) W celu realizacji algorytmu synchronizowanego zdefiniować w ramach klasy atrybut będący referencją na wspólny dla wątków obiekt semafora.
5) W celu zademonstrowania niepoprawnych zjawisk związanych z niesynchronizowanym wykonywaniem wyłącznego kodu w sekcji krytycznej, należy w ramach sekcji wypisać następujący komunikat:

```
Sekcja krytyczna wątku: Sem-1, nr powt.= 3
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

   lub

```
Sekcja krytyczna wątku: Sem-2, nr powt.= 3
------------------------------------------------------------------------
```

6) Ciąg znaków w drugiej linii powstaje poprzez 100-krotne wypisanie pojedynczego znaku.
7) Każdy z wątków ma swój własny znak dostępny przez tablicę `znaki` i numer wątku.
8) W celu zademonstrowania programu zdefiniować klasę uruchomieniową `Test` i uruchomić 5 wątków klasy `Sem`.