Solution Serie 9 - Fairness and Optimism

Exercise 1

Answer the following questions: (1 point each)

1. What criteria might you use to prioritize threads? **Answer:**
   
   Priority may depend on any of:
   
   - Intrinsic attributes of tasks (class & instance variables).
   - Representations of task priority, cost, price, or urgency.
   - The number of tasks waiting for some condition.
   - The time at which each task is added to a queue.
   - Fairness guarantees that each waiting task will eventually run.
   - Expected duration or time to completion of each task.
   - The desired completion time of each task.
   - Termination dependencies among tasks.
   - The number of tasks that have completed.

2. What are different possible definitions of fairness? **Answer:**

   - Weak fairness: If a process continuously makes a request, eventually it will be granted.
   - Strong fairness: If a process makes a request infinitely often, eventually it will be granted.
   - Linear waiting: If a process makes a request, it will be granted before any other process is granted the request more than once.
   - FIFO (First-in First out): If a process makes a request, it will be granted before that of any process making a later request.

3. What are Pass-Throughs and Lock-Splitting? **Answer:**

   - Pass-Throughs: The host maintains a set of immutable references to helper objects and simply relays all messages to them within unsynchronized methods.
   - Lock-Splitting: Instead of splitting the class, split the synchronization locks associated with subsets of the state.

4. When should you consider using optimistic methods? **Answer:**
• Clients can tolerate either failure or retries. If not, consider using guarded methods.
• You can avoid or cope with livelock.
• You can undo actions performed before failure checks:
  - **Rollback/Recovery**: undo effects of each performed action. If messages are sent to other objects, they must be undone with “anti-messages”
  - **Provisional action**: “pretend” to act, delaying commitment until interference is ruled out.

**Exercise 2 (6 points)**

In this exercise you have to build a class that represents graphical objects that consist of an x-coordinate, a y-coordinate, a width and a height. The class has to implement methods for:

- Increase the x-coordinate by 10% and decrease the y-coordinate by 20% (change position)
- Increase the width by 50% and decrease the height by 80% (change dimension)
- Increase the y-coordinate by 40% and decrease the height by 60% (change position and dimension)

Implement it once using Lock-Splitting and once using Pass-Throughs (use the Shape interface listed below).

```java
public interface Shape {
    public void changePosition();
    public void changeDimension();
    public void changePositionAndDimension();
}
```

**Answer:**

*The solutions can be found in CP-Series9-Sol.zip*.¹