Homework #3. Multithreading

Prepared by Dmitry Chub [dmchub@yandex.ru](mailto:dmchub@yandex.ru)

* Deadline:
  + Soft (-50 % points) 21st December 2022
  + Strict (-100% points) 23rd December 2022

## Problem A. Always know your enemy

### Problem statement

Your task is to create deadlock and livelock. Also, you need to explain (in JavaDoc comment or in a separate file) the correctness of your solution.

### Grading system

* 2 points for deadlock implementation
* 3 points for livelock implementation
* 1 point for tests / demonstrations that show the correctness of your code

### Notes

A deadlock is a state in which each member of a group of actions is waiting for some other member to release a lock.

Livelock occurs when two or more processes continually repeat the same interaction in response to changes in the other processes without doing any useful work. These processes are not in the waiting state, and they are running concurrently.

You may use [this project](https://gitlab.com/fpmi-atp/atp-java22/foreigners/-/tree/master/HW-3) for help. There you will find test cases for livelock and deadlock. However, keep in mind that these tests do not guarantee that your code works / doesn’t work, since they don’t cover all possible cases.

## Problem B. Speedtest time

### Problem statement

In this problem you need to work with concurrent collections. You may choose up to three concurrent collections that you like most. For every collection you need to describe the main functionality of the collection, what operations it can perform and a real situation where this collection can be used and emulate this situation. Also, you need to create a small report with your conclusions about this collection (compare it with other collections that can solve the same problem and the single-thread analogue, measure the performance).

### Grading system

* 3 points for every report (no more that three reports)
* 2 additional points for each report, if your emulator creates a graph, which shows the dependency between efficiency and amount of threads (the definition of efficiency may be different, for example, amount of operations per second or total time spent on all the operations)

Notes

The [demo from the seminar](https://gitlab.com/fpmi-atp/atp-java22/foreigners/-/blob/master/Sem11/src/main/java/ConcurrentDataStructuresDemo.java) is an example of concurrent hash map usage ‘emulator’.