UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Department of Computer Science
COS 226 - Concurrent Systems

Practical 6

- **Date issued:** 13 October 2022
- **Deadline:** 20 October 2022, 11:59 PM

1 Introduction

1.1 Objectives and Outcomes

This practical aims to further explore synchronization by evaluating locking in alternative data structures to queues.

You must complete this assignment individually. Copying will not be tolerated.

1.2 Submission and Demo Bookings

You are NOT provided with any skeleton code for this practical, you will have to implement everything yourself.

Submit your code to **clickup** before the deadline.

You will have to demonstrate each task of this practical during the **physical** practical lab session. So be sure to create copies of your source code for each task separately. Booking slots will be made available for the practical demo.

1.3 Mark Allocation

For each task in this practical, in order to achieve any marks, the following must hold:

- Your code must produce console output. (As this is not marked by fitchfork, formatting is not that strict)
- Your code must not contain any errors. (No exceptions must be thrown)

- Your code may not use any external libraries apart from those highlighted in the textbook.
- You must be able to explain your code to a tutor and answer any questions asked.

The mark allocation is as follows:

Task Number	**Marks**
Tasks combined	10
Total	10

2 Practical Requirements

The task for this practical can be found in Chapter 10 of the prescribed textbook. It is part of the chapter 10.9 Exercises. Implement the following tasks.

2.1 Tasks - Exercise 121

Design a **bounded lock-based queue** implementation using an array instead of a linked list.

1. Allow parallelism by using two separate locks for head and tail.
2. Try to transform your algorithm to be lock-free. Where do you run into difficulty?

Note:

- You code must demonstrate concurrency on the new design using a variable number of threads.
- Provide separate implementation for the second task.
- **NB! Please note that you must be able to explain the implementations, and difficulties expressed in the second task if any.**

2.2 Output

The following output is expected:

- Output to demonstrate Enqueue
- Output to demonstrate Dequeue