Assignment 1 report

Jesper Wingren

[Jw223rn@student.lnu.se](mailto:Jw223rn@student.lnu.se)

1dt909

Innehållsförteckning

[1. Dequeue 3](#_Toc159774575)

[1.1 Setup 3](#_Toc159774576)

[1.2 Results 3](#_Toc159774577)

[2. 3Sum 4](#_Toc159774578)

[2.1 Setup 4](#_Toc159774579)

[2.2 Results 4](#_Toc159774580)

# Dequeue

## Setup

The dequeue is setup so that the head and tail will always be fixed values but when handling the values it in a way ignores these so for example addFront doesn’t make it the head it adds it to the front but inside the head and tail. The find method is used to find the back of the dequeue making it possible to add and pop from the back.

## Locking

My dequeue implementation uses fine-grained locking by each node having its own mutex lock. In the code it only locks the node or nodes it Is currently handling for example the addFront it only locks the head node and not the whole code making it possible to concurrently handle other methods. This fine-grain locking also reduces the chances of methods concurring each other and avoiding deadlocks. By using a find method for the back method, the program only locks the two nodes its currently watching meaning in a longer dequeue it can use find concurrently because it’s at different spots in the queue, hence not locking each other out.

# Password-cracker

## Setup

We begin by creating two channels, one for passwords to check and one for the found password.

## Results

ss