Concurrency 2: The Dining Philosophers Problem

Courtney Bonn, Isaac Chan

Group #39

I. How should the TA evaluate your work? Provide detailed steps to prove correctness

There are two commands that will compile our concurrency file.

1) make con2
2) gcc dining.c -pthread -lrt -o dining

After the file is compiled, ./dining will run the program. The program will start and the philosophers will think and eat. They will do this endlessly until the user interrupts the program.

We verified the correctness of the program by first printing the numbers of the philosophers instead of their names. This ensured that we could manually check that there was no overlap of forks being used. Since there was no overlap, it meant that the fork mutexes were working correctly. We were also able to see that multiple philosophers were able to eat concurrently, as long as the forks they used were separate.

By having the program run endlessly, it shows that there is no deadlock occuring, when philosophers are trapped in an eat or think cycle with no end, and not getting stuck and starving some philosophers. It also demonstrates that no philosophers were randomly dying through the program runtime.