Corey Kipp (Student ID: 57723335)

kippc@uci.edu

CS 143A

Homework 6 Write-Up

Part 1

In the dining philosophers problem the four conditions for deadlock are held because…

At least one chopstick is non-shareable.

There is one philosopher holding one chopstick and waiting for another one.

No philosopher is willing to give up the chopstick they are currently holding.

Each philosopher is waiting on another philosopher in a circular pattern, making it so dependency never ends.

Deadlock could be avoided if…

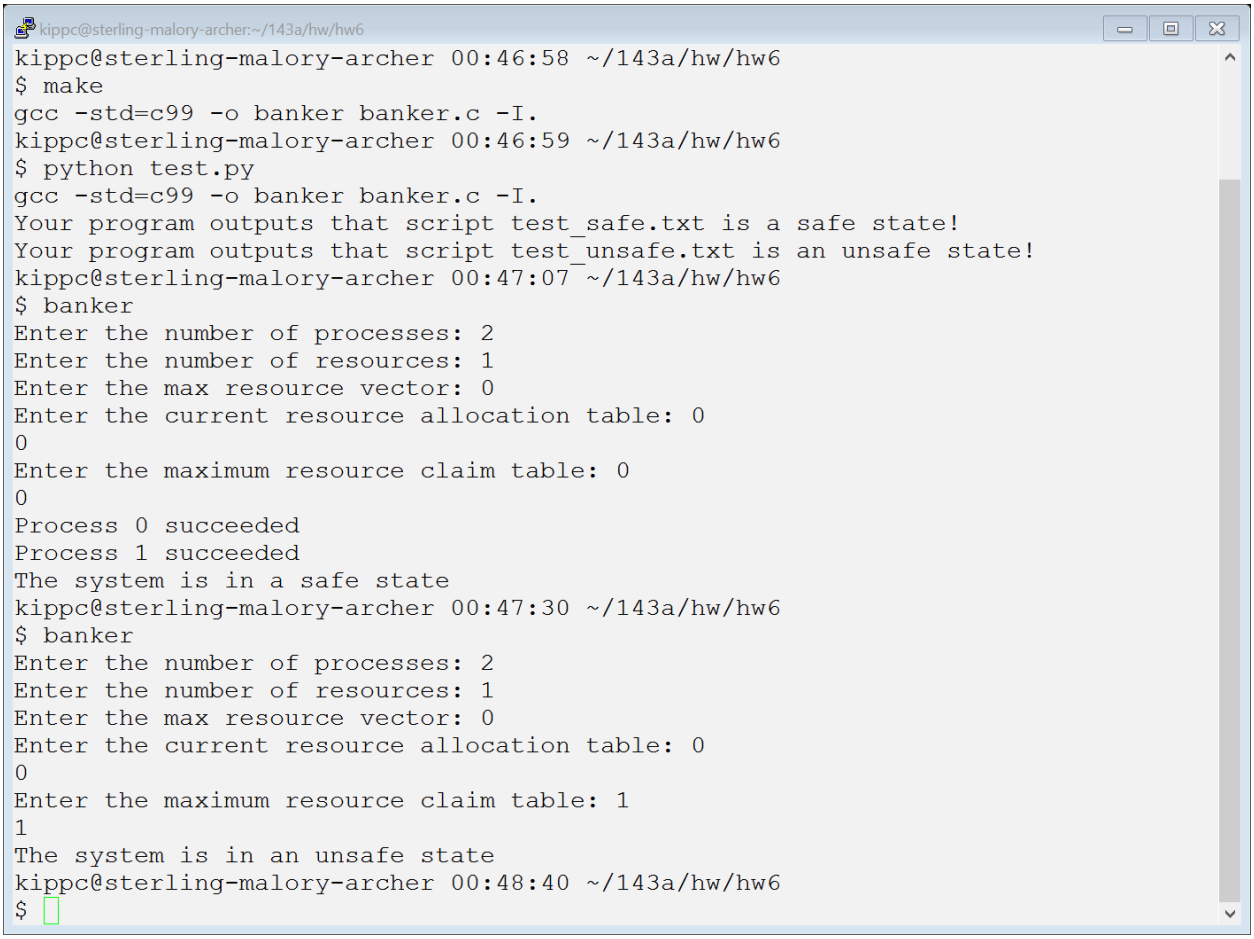
At least one chopstick becomes shareable, making at least one philosopher able to eat.

If one philosopher doesn't hold and wait and instead puts the chopstick he was holding down, then another philosopher can eat.

If a philosopher takes another philosopher's chopstick then they can eat.

If one philosopher isn't waiting in the circular pattern then the dependency can end.

Part 2


kippc@sterling-malory-archer:~/143a/hw/hw6
kippc@sterling-malory-archer 00:46:58 ~/143a/hw/hw6
$ make
gcc -std=c99 -o banker banker.c -I.
kippc@sterling-malory-archer 00:46:59 ~/143a/hw/hw6
$ python test.py
gcc -std=c99 -o banker banker.c -I.
Your program outputs that script test_safe.txt is a safe state!
Your program outputs that script test_unsafe.txt is an unsafe state!
kippc@sterling-malory-archer 00:47:07 ~/143a/hw/hw6
$ banker
Enter the number of processes: 2
Enter the number of resources: 1
Enter the max resource vector: 0
Enter the current resource allocation table: 0
0
Enter the maximum resource claim table: 0
0
Process 0 succeeded
Process 1 succeeded
The system is in a safe state
kippc@sterling-malory-archer 00:47:30 ~/143a/hw/hw6
$ banker
Enter the number of processes: 2
Enter the number of resources: 1
Enter the max resource vector: 0
Enter the current resource allocation table: 0
0
Enter the maximum resource claim table: 1
1
The system is in an unsafe state
kippc@sterling-malory-archer 00:48:40 ~/143a/hw/hw6
$