Deadlock Repair

For this problem, you get to fix a program that currently exhibits a deadlock. On the course homepage, you'll find source code for a program called deadlock.c. This program creates 10 threads, and lets them run for 5 seconds. While the threads run, they repeatedly acquire locks on items they need (a hammer, a usb cable, a spatula, etc). When they have locks on all the items they want, they print out their name and then release the locks and then repeat.

In its current state, this program deadlocks (maybe not every time you run it, but probably). You're going to fix this by applying the no-circular-wait deadlock prevention technique. In its current state, the threads lock the items they want in an arbitrary order. Instead, you're going to choose an order for the objects and change the thread code so they all lock the objects in the order you've chosen. You won't change the items each thread wants to use; you'll just make sure that they acquire locks on their objects in the right order. For example, if you order the objects so the shovel comes before the camera, you'll make sure any thread that needs these two objects acquires the lock on the shovel before locking the camera.

When you're done, your program should run without deadlocking every time. Submit your working source code to the exercise_12 assignment in Moodle.