**Without CyclicBarrier**

*public class* WithoutCyclicBarrier {

 *public static void* main(String[] args) {
 *var* service = Executors.*newFixedThreadPool*(3);
 *try* {

 WithoutCyclicBarrier **cleaner** = *new* WithoutCyclicBarrier();
 *for* (*int* i = 0; i < 3; i++) {
 service.submit(() -> cleaner.clean());
 }
 } *finally* {
 service.shutdown();
 }
 }

 *private void* clean() {
 *try* {
 vacuumFloor();
 Thread.*sleep*(Math.*round*(Math.*random*() \* 1000));
 coffeeBreak();
 Thread.*sleep*(Math.*round*(Math.*random*() \* 1000));
 mopFloor();
 Thread.*sleep*(Math.*round*(Math.*random*() \* 1000));
 turnOffLights();
 activateAlarm();
 } *catch* (InterruptedException exception ) {
 System.*out*.println(exception.getMessage());
 }
 }

 *private void* vacuumFloor() {
 System.*out*.println("Vacuming the floor...");
 }
 *private void* mopFloor() {
 System.*out*.println("Mopping the floor...");
 }

 *private void* coffeeBreak() {
 System.*out*.println("Time for a break!");
 }
 *private void* turnOffLights() {
 System.*out*.println("Turning off the lights...");
 }
 *private void* activateAlarm() {
 System.*out*.println("Activating the alarm...");
 }
}

Vacuming the floor...

Vacuming the floor...

Vacuming the floor...

Time for a break!

Mopping the floor...

Time for a break!

Time for a break!

Mopping the floor...

Turning off the lights...

Activating the alarm...

Mopping the floor...

Turning off the lights...

Activating the alarm...

Turning off the lights...

Activating the alarm...