# Thread Lab

Requirements

* Review the provided Car, CarPanel and CarFrame classes
* Create a CarQueue class that maintains a queue of random directions (that the car should go)

1. The addToQueue method has a class that implements runnable, define the run method (add random directions into the queue and then sleep), creates an instance of the runnable object, creates a thread and starts the thread.

/\*\* Adds 0,1,2 or 3 to queue

\* 0 = up

\* 1 = down

\* 2 = right

\* 3 = left

\*/

1. It also has a deleteQueue method that returns an integer
2. In your constructor, place 5 or 6 numbers in the queue so that when the animation starts – there will be something to retrieve from the queue

* Modify the run method of the startAnimation in CarPanel so that the car will go to the right, left, up or down depending on what is retrieved from the CarQueue. The cars should go in the opposite direction if they hit a boundary.

Additional Insights from a Student from a Previous Semester

This lab centered around threads, and how Runnables are executed in a program and GUI.

I was tasked with creating the queue that would generate the car’s direction of travel.

In building the CarQueue class, I used my previously created MyQueue class to handle the enqueue and dequeue methods. I then implemented a Random object to populate the instantiated queue with random integers of 0, 1, 2, or 3 so that cars would randomly move as numbers were dequeued.

In the addToQueue method, I created a nested Runnable object that would re-populate the queue with directions before calling Thread.sleep. The addToQueue then creates a new Thread containing this Runnable object to automatically add to the queue when addToQueue is called.

In the CarPanel class, I added to its own run method to indicate what happens when a given number is dequeued from the CarQueue. Using the established boundaries of the CarFrame window, I set the x and y coordinates of the car graphics to not run outside of the window bounds. If a car is up against a boundary and tries to go past that boundary, it instead travels for 10 units in the opposite direction. I also added a line to call addToQueue at the start of the run method so that the CarQueue object would always have an integer available to dequeue.