**Data Structures and synchronization**

For lab 1, I first thought about what I would need to store the vehicles in order to send them through the traffic program. Since the idea is to have an in order traffic system, the data structure I chose for both sides of the bridge were queues. I also had global variables storing the current traffic direction, the current weight on the bridge, an array to store the vehicles currently on the bridge, a mutex lock to ensure security of global variables and a waitgroup for the main function to wait until all the vehicle threads were finished running. I created structs for individual vehicles to store vehicle ids, type (Car or truck), the direction it traveled, and its condition variable. My queue struct that stores an array of vehicles. This struct is used for both directions in order to organize the queue at each end for a first in first out structure. I created a Tuple struct for organization in my main method for creating my group sizes and delays. This made it more organized for me to create my loops for creating vehicle threads and incorporating their group delays.

**High Level Overview**

My program prompts the user for the number of vehicle groups, the size of each group and their delays, the chance of each vehicle's direction and type, then creates the vehicles in terms of threads to pass through the intersection synchronously. The constraints of this algorithm are that no vehicle should wait for more than 5 opposite direction vehicles to cross, the weight on the bridge should not exceed 750 units of weight, and the bridge should only have one direction of traffic flowing at a time as it is a one lane bridge. Each vehicle can have the option of being a car or a truck with weights 100 or 300 respectively.

**Main**

The main function is used to create the vehicle threads. The user inputs the number of groups, vehicles per group, chance of each vehicle being a car or truck, and the chance of each vehicle's direction being south or north. Each vehicle is created through the representation of a thread and is considered in the waitgroup variable for the program to wait until all threads have completed their task of crossing the bridge. Each vehicle has its own unique identification number. There is a potential delay in seconds that may be inputted by the user between each group.

**OneVehicle**

oneVehicle is called by main which is the initiation of a new thread. This thread is created based on a vehicle that was created in main that must cross the bridge safely.

**Arrive**

Once a thread reaches this function, it is printed on the screen to let the user know this thread was created and is approaching the bridge. If there are more than 5 vehicles currently crossing the bridge from the other direction then the present vehicle will receive the chance to cross the bridge next once all traffic has left the bridge from the opposite direction. If traffic is already flowing in the vehicle's direction and a vehicle from the other side has not been waiting for more than 5 vehicles, then the vehicle is allowed to enter the cross function.

**Cross**

Once a thread reaches this function, it must ensure it is completely safe and fair to cross the bridge. It must check if it is the first vehicle in the queue to cross the bridge, the bridge traffic is going in the vehicles direction, the weight of the bridge is going to exceed 750 units with the addition of the current vehicle, and if the vehicle on the opposite side has been waiting for more than 5 vehicles to cross. If all these conditions are met, the vehicle will then start to cross the bridge which takes 2 seconds. If they are not met, the vehicle will wait to be prompted to cross by another vehicle.

**Leave**

Once a thread reaches this function, it is almost done with its journey across the bridge. The bridge is adjusted based the leaving vehicle deciding on which vehicle to allow to cross the bridge next. If there is a vehicle behind them that is able to enter the bridge, that vehicle will be informed to cross. If a vehicle from the opposite direction has been waiting, that vehicle will be allowed to cross after all the vehicles crossing from the current direction have left the bridge. This will switch the direction and traffic will flow from the new direction. Once the vehicle has done its job of letting the next driver know they may cross the bridge, the vehicle will leave and the thread will exit.