This program uses arrays to take in groups of input from the user. Those arrays are used to simulate bridge traffic given weight and direction constraints. The data is updated using a lock and condition variable to ensure accuracy. The bridge itself is not an object, but rather a collection of data points representing its state. The vehicles that “traverse” the bridge are represented by threads running on a routine that will bring them to the bridge, across the bridge, and off of the bridge. There are on-screen print outs to allow the user a glimpse into the state of the program and its moving parts. Following is the pseudocode for the program.

MAIN FUNCTION

Prompt user for group quantity

Scan input into groups variable

Initialize vehicle, direction probability, delay time arrays with the group quantity as size

For i in groups

Prompt user for vehicle quantity, direction probability, delay time

Scan input into arrays

For i in groups

Set the vehicle counter to 0

Initialize thread array

For j in vehicle array at index i

Initialize thread array

Seed the RNG

Initialize the random number

If the random number is less than or equal to 0.5

The vehicle is a car

Else the vehicle is a van

If the probability is 100 or 0

The vehicle goes north or south

Else

Assign a new random number

If the new number is less than or equal to the direction probability at i

The vehicle goes north

Else the vehicle goes south

Create thread with id = j+1, type, and direction, using the vehicle routine

Sleep for the value at index i of the delay time array

VEHICLE ROUTINE FUNCTION

Call arrive function

Call cross function

Call leave function

ARRIVE FUNCTION

Obtain the lock

While the bridge is too heavy

Wait for the bridge to have less weight

Choose a lane

Increment that lane count

Add weight to the bridge

Release the lock

CROSS FUNCTION

Print the crossing message

Sleep for 3 seconds

LEAVE FUNCTION

Print the leaving message

Obtain the lock

Subtract the weight from the bridge

Decrement the lane count

Release the lock

Signal any waiting thread