## **Task List**

* Task FP.1 : Define a class TrafficLight which is a child class of TrafficObject. The class shall have the public methods void waitForGreen() and void simulate() as well as TrafficLightPhase getCurrentPhase(), where TrafficLightPhase is an enum that can be either red or green. Also, add the private method void cycleThroughPhases(). Furthermore, there shall be the private member \_currentPhase which can take red or green as its value.
* Task FP.2 : Implement the function with an infinite loop that measures the time between two loop cycles and toggles the current phase of the traffic light between red and green and sends an update method to the message queue using move semantics. The cycle duration should be a random value between 4 and 6 seconds. Also, the while-loop should use std::this\_thread::sleep\_for to wait 1ms between two cycles. Finally, the private method cycleThroughPhases should be started in a thread when the public method simulate is called. To do this, use the thread queue in the base class.
* Task FP.3 : Define a class MessageQueue which has the public methods send and receive. Send should take an rvalue reference of type TrafficLightPhase whereas receive should return this type. Also, the class should define an std::dequeue called \_queue, which stores objects of type TrafficLightPhase. Finally, there should be an std::condition\_variable as well as an std::mutex as private members.
* Task FP.4 : Implement the method Send, which should use the mechanisms std::lock\_guard<std::mutex> as well as \_condition.notify\_one() to add a new message to the queue and afterwards send a notification. Also, in class TrafficLight, create a private member of type MessageQueue for messages of type TrafficLightPhase and use it within the infinite loop to push each new TrafficLightPhase into it by calling send in conjunction with move semantics.
* Task FP.5 : The method receive should use std::unique\_lock<std::mutex> and \_condition.wait() to wait for and receive new messages and pull them from the queue using move semantics. The received object should then be returned by the receive function. Then, add the implementation of the method waitForGreen, in which an infinite while-loop runs and repeatedly calls the receive function on the message queue. Once it receives TrafficLightPhase::green, the method returns.
* Task FP.6 : In class Intersection, add a private member \_trafficLight of type TrafficLight. In method Intersection::simulate(), start the simulation of \_trafficLight. Then, in method Intersection::addVehicleToQueue, use the methods TrafficLight::getCurrentPhase and TrafficLight::waitForGreen to block the execution until the traffic light turns green.