

< Return to "C++" in the classroom

DISCUSS ON STUDENT HUB

Program a Concurrent Traffic Simulation

REVIEW CODE REVIEW HISTORY Meets Specifications Hello there, Congratulations on finishing the project 🕭 Keep doing the great work and all the best for future project. FP.1 Create a TrafficLight class



FP.2: Implement a cycleThroughPhases method

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.

The cycle duration should be a random value between 4 and 6 seconds, and the while-loop should use std::this_thread::sleep_for to wait 1ms between two cycles.

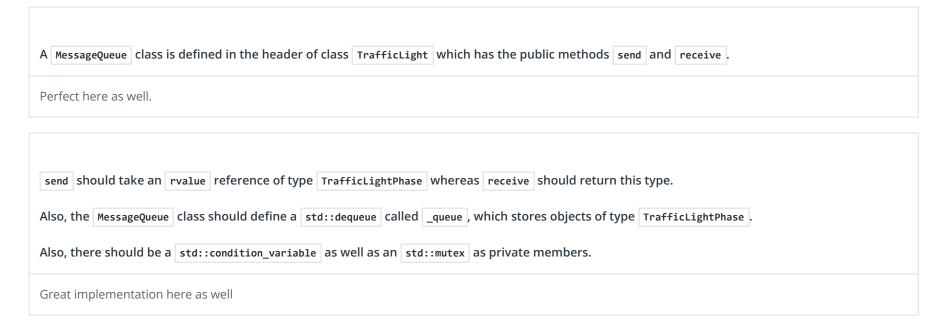
Well done! Your class has a cycleThroughPhases method implemented, and this method has a cycle duration between 4 and 6 seconds with a while loop that sleeps for 1ms between cycles.

Note that for improved randomness, you can also use a random device and a Mersenne Twister pseudo-random generator as follows:

```
std::random_device rd;
std::mt19937 eng(rd());
std::uniform_int_distribution<> distr(4000, 6000);
double cycleDuration = distr(eng); // duration of a single simulation cycle in ms
```

The private cycleThroughPhases() method should be started in a thread when the public method simulate is called. To do this, a thread queue should be used in the base class. Great work!

FP.3 Define class MessageQueue



FP.4 Implement the method `send`

The method send 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. In the class | TrafficLight |, a private member of type | MessageQueue | should be created and used within the infinite loop to push each new TrafficLightPhase into it by calling send in conjunction with move semantics.

This is well implemented.

FP.5 Implement the methods `receive` and `waitForGreen`

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.

This works like a charm!

The method waitForGreen is completed, 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.

This option is included as well.

FP.6 Implement message exchange

In class Intersection, a private member _trafficLight of type TrafficLight should exist.

The method _Intersection::simulate(), should start the simulation of _trafficLight.

The method _Intersection::addVehicleToQueue , should use the methods _TrafficLight::getCurrentPhase and _TrafficLight::waitForGreen to block the execution until the traffic light turns green.

This looks great!

■ DOWNLOAD PROJECT

RETURN TO PATH

Rate this project