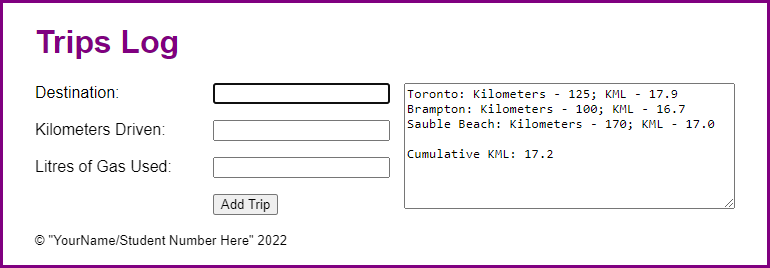
**Assignment 6-1 Instructions: Use the module pattern to create private state**

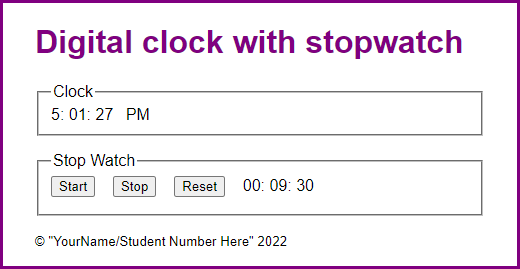
In this assignment, you’ll update the Trips application so it uses the module pattern to create private state for the object that stores the array of trips. When you’re done, the application should work the same as it did before.



1. Open the starter application(zipped) attached to this Assignment folder:
   1. Evaluations\Assignments\Assignment 6\Part1-trips.zip
   2. Run and test the application.
2. Review the library file (lib\_trips.js). Note that it provides a class named Trip that you can use to create trip objects and a class named Trips that you can use to create trips objects.
3. Modify the library file, so it uses the module pattern to create a trips object that uses a private constant to store the array of Trip objects but provides public properties and methods named push(), totalKml, and toString().
4. Open the main file (trips.js) and modify it so it uses the trips object defined by the library instead of using the Trips class to create an object. Note how this makes the code shorter.
5. In index.html, change the text in the <small> tag to your name and student number.
6. Run and test the application again to make sure it still works as expected.

**Assignment 6-2 Instructions: Convert the Clock application to closures**

In this assignment, you’ll convert a Clock application to use closures.



1. Open the starter application(zipped) attached to this Assignment folder:
   1. Evaluations\Assignments\Assignment 6\Part2-clock\_closures.zip
2. Note that the folder contains two library files named library\_clock.js and library\_stopwatch.js. In the main JavaScript file, note that the variables, objects, and functions provide all the functionality for the clock and stopwatch but without using the libraries.
3. In the library\_clock.js file, there’s a start for a function named createClock(). Note that this function has parameters for the span tags that display the clock in the page.
4. In the library\_stopwatch.js file, there’s a start for a function named createStopwatch(). Note that this function has parameters for the span tags that display the stopwatch in the page.
5. In the clock.js file, find the functions that run the clock and move them to the private state section of the library\_clock.js file. Then, in the public methods section of the clock library file, code and return an object that contains a method named start(). This method should use the private state to start the clock. Adjust the code as needed to make this work.
6. In the clock.js file, find the variables, objects, and functions that run the stopwatch and move them to the private state section of the library\_stopwatch.js file. Then, in the public methods section, code and return an object that contains methods named start(), stop(), and reset(). These methods should use the private state to start, stop, and reset the stopwatch. Adjust the code as needed to make this work.
7. Still in the clock.js file, rewrite the remaining code so the ready event handler calls the functions in the library files, passes them the span tags they need, and uses the returned objects to start the clock and attach the stopwatch event handlers.
8. In index.html, change the text in the <small> tag to your name and student number.