You may continue to work with your partner on this assignment, or you may work by yourself, whichever you choose.

In this project we will create three classes that will enable us to create an airline flight list. You are to write the three classes from scratch according to the following instructions. When you are finished with each class, you should then run the provided tester class to validate your work.

The first class is a `Time` class that represents a certain hour and minute in the day. The second class is a `Flight` class that represents a given airline flight. The third class is a `Trip` class, that represents one or more `Flight`s.

Your task is to complete a new project called Flight List that will contain the `Time`, `Flight`, and `Trip` classes that you will write as specified below. Download, extract, and save the Flight List project files to your network directory (or flash drive). You can run the `FlightListTest` program anytime you want and it will show you the status of your work so far.

Good luck. Show me the results of your successful test when you are finished. Feel free to see me if you have any questions or difficulties in the meantime.

Part A Write the `Time` class according to the instructions below. Instance variables should be `private`, while methods should be `public`.

1. Declare two instance variables (both integers), on for the hours, and one for the minutes of the day. For our purposes we will be using a 24-hour clock (military time), and it is presumed that all `Time` objects are for the same day.
2. Write a constructor for `Time(int h, int m)` that will accept the hours and minutes as parameters. Be sure to assign the passed parameters to the hours and minutes instance variables.
3. Write a `setTime` method that (like the constructor) accepts hours and minutes as parameters, and will then set the instance variables to those values. This method will be used to change the time of an already instantiated `Time` object.
4. Write a `getHours` method that will return the hours as an integer value.
5. Write a `getMinutes` method that will return the minutes as an integer value.
6. Write an `isValidTime` method that returns a boolean value. This method checks to see if the `Time` object is set to a valid time of day. Valid hours are between 0 - 23 inclusive, and valid minutes are between 0 - 59 inclusive.
7. Write a `toString` method that returns a `String` description of the `Time` object. The proper format for the `toString` method should be "hours:minutes". Challenge: can you format the output so that single digit hours or minutes have a leading zero (i.e. 5 hours, 3 minutes would be written as 05:03)?
8. Write a `minutesUntil(Time other)` method that returns an integer value of the number of minutes between `this` `Time` object and the `other` `Time` object. If this `Time` object is earlier than the other `Time` object, then the returned value is positive. If this `Time` object is later than the other `Time` object, then the returned value is negative.

Part B - In the same Flight List project, create another class called `Flight`, according to the instructions below.

1. Create three instance variables, namely a flight number, a departure time, and an arrival time. The flight number is a `String` variable (because it may also include letters), and both departure and arrival are `Time` objects. This class is not concerned about departure and arrival locations.
2. Write a constructor for `Flight(String f, Time d, Time a)` that will accept the flight number, and departure and arrival times as parameters. Be sure to assign the passed parameters to the appropriate instance variables that you just created.
3. Write a `setFlight` method that (like the constructor) accept the flight number, departure time, and arrival time as parameters, and will then set the instance variables to those values. This method will be used to change an already instantiated `Flight` object.
4. Write a `getDepartureTime` method that returns a `Time` object of when the flight departs.
5. Write a `getArrivalTime` method that returns a `Time` object of when the flight arrives.
6. Write a `toString` method that returns a `String` description of the `Flight` object. The description should include the flight number, the departure and arrival times, as well as the length of the flight in minutes.

Part C - In the same Flight List project, create another class called `Trip`, according to the instructions below.

1. Create two instance variables, namely a trip description, and list of flights. The trip description is a `String` variable named `trip`, and the list of flights is a `List` of `Flight` objects named `flights`. (It is important that you use those particular variable names, as they are used in the `toString` method provided below.)
2. Write a constructor for `Trip(String t, List<Flight> f)` that will accept the trip description, and a `List` of `Flight` objects as parameters. Be sure to assign the passed parameters to the appropriate instance variables that you just created.
3. Write the `getDuration` method, which returns the number of minutes (an integer) from the departure of the first flight to the arrival of the last flight if there are one or more flights in the trip. If there are no flights in the trip, then `getDuration` returns 0.
4. Write the `getShortestLayover` method, which returns the smallest number of minutes (an integer) between the arrival of a flight and the departure of the flight immediately after it, if there are two or more flights in the trip. If there are zero or one flights in the trip, then `getShortestLayover` returns -1. You should assume that the departure time for one flight is later than the arrival time of the preceding flight.
5. Write a `toString` method as follows (copy this code):

```
public String toString()
{
    String t = "The " + trip + " trip itinerary is as follows:\n\n";
    for (int i = 0; i < flights.size(); i++)
    {
        t += flights.get(i);
        if (i < flights.size() - 1)
        {
            t += "\nThe layover between flights is ";
            t += flights.get(i).getArrivalTime().minutesUntil
                      (flights.get(i + 1).getDepartureTime());
            t += " minutes\n\n";
        }
    }
    t += "\nThe duration of the entire trip is " + getDuration() + " minutes\n";
    t += "The shortest layover is " + getShortestLayover() + " minutes";
    return t;
}
```

If you have completed all of the above steps, then run the tester program to verify that everything is correct. If everything is hunky-dory, then you will see the message that you have passed the test. Otherwise, keep working at it until you are done. Once again, if you are stuck, please consult with another classmate or else see me for help.