Student ID_____**Student Name**_____

ASD practice final exam solution

PRIVATE AND CONFIDENTIAL

1. **Allotted exam duration is 2 hours.**
2. **Closed book/notes.**
3. **No personal items including electronic devices (cell phones, computers, calculators, PDAs).**
4. **Cell phones must be turned in to your proctor before beginning exam.**
5. **No additional papers are allowed. Sufficient blank paper is included in the exam packet.**
6. **Exams are copyrighted and may not be copied or transferred.**
7. **Restroom and other personal breaks are not permitted.**
8. **Total exam including questions and scratch paper must be returned to the proctor.**

3 blank pages are provided for writing the solutions and/or scratch paper. All 3 pages must be handed in with the exam

BE VERY CAREFUL WITH THE GIVEN 2 HOURS AND USE YOUR TIME WISELY. THE ALLOTED TIME IS GIVEN FOR EVERY QUESTION.

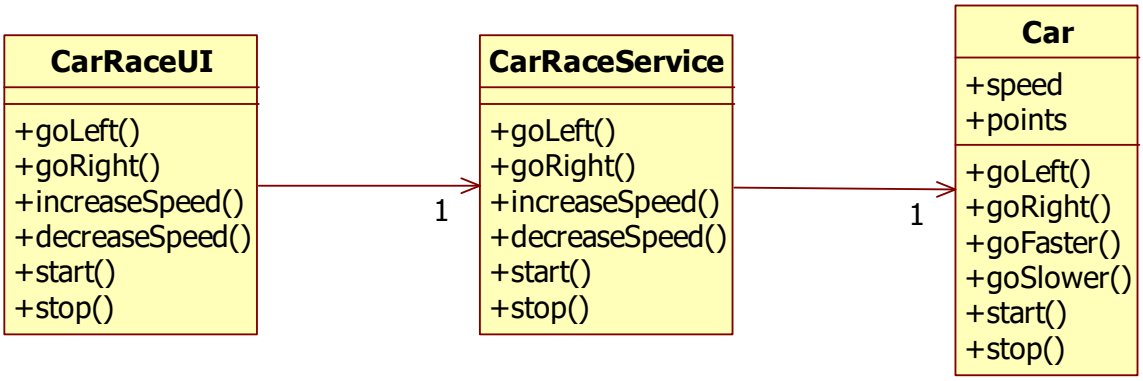
Write your name and student id at the top of this page.

Question 1 [30 points] {30 minutes}

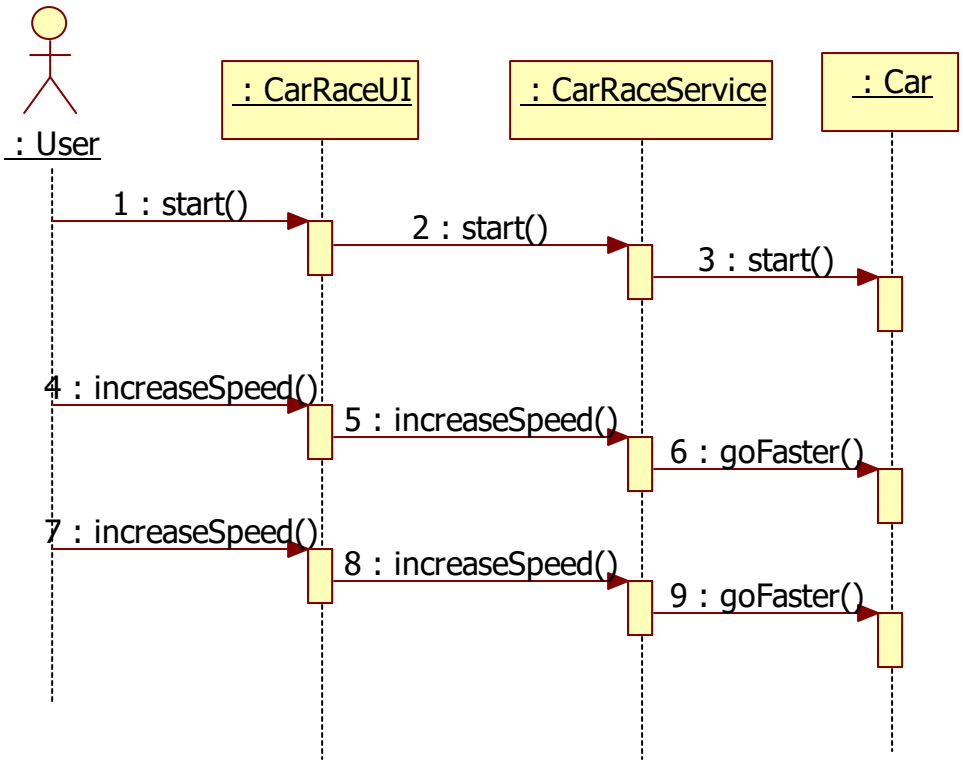
Suppose you need to design a framework for a simple car racing game with the following requirements:

- The user can do the following actions: go left, go right, increase speed and decrease speed.
- The user gets points for how well it stays on the road, and how fast it can race.

For this question we will only focus on the Car, and the actions on the Car. Suppose we have the following **partial** class diagram of the framework:

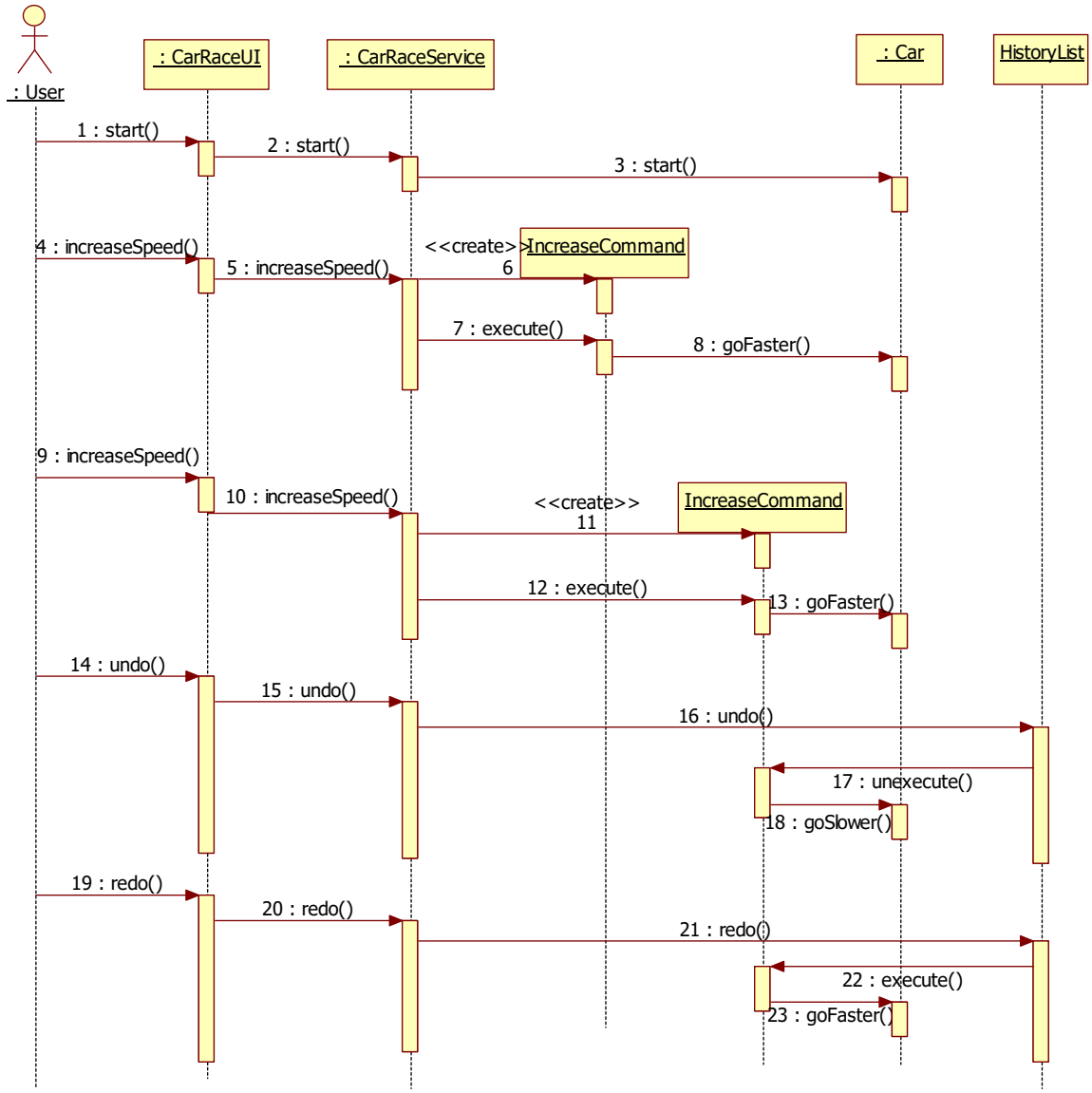


The sequence diagram of a possible scenario looks like this:



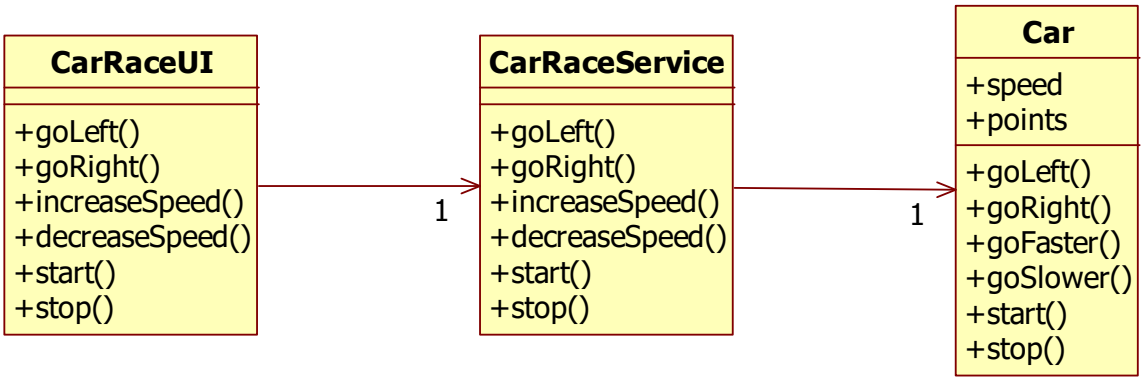
Now suppose we get a new requirement that the framework should also support undo and redo functionality for the actions **increaseSpeed**() and **decreaseSpeed**(). This means the CarRaceUI class gets 2 more buttons: an **undo** button and a **redo** button.

Draw the sequence diagram of the following given scenario using your new design that supports undo/redo functionality:



Question 2 [30 points] {35 minutes}

Suppose we have the same simple car racing game framework that we started with in question 1. Suppose we have the following **partial** class diagram of the framework:



But now we get a new requirement that the framework should support different car modes. The modes supported by the framework are:

1. Rookie Racer: This means you have less than 1000 points, and if you press the increaseSpeed button, your speed is increased with 1 mile per hour. If you press the decreaseSpeed button, your speed is decreased with 1 mile per hour

2. Racer: This means you have between 1000 and 5000 points, and if you press the increaseSpeed button, your speed is increased with 2 miles per hour. If you press the decreaseSpeed button, your speed is decreased with 2 miles per hour

3. Pro Racer: This means you have between more than 5000 points, and if you press the increaseSpeed button, your speed is increased with 3 miles per hour. If you press the decreaseSpeed button, your speed is decreased with 3 miles per hour

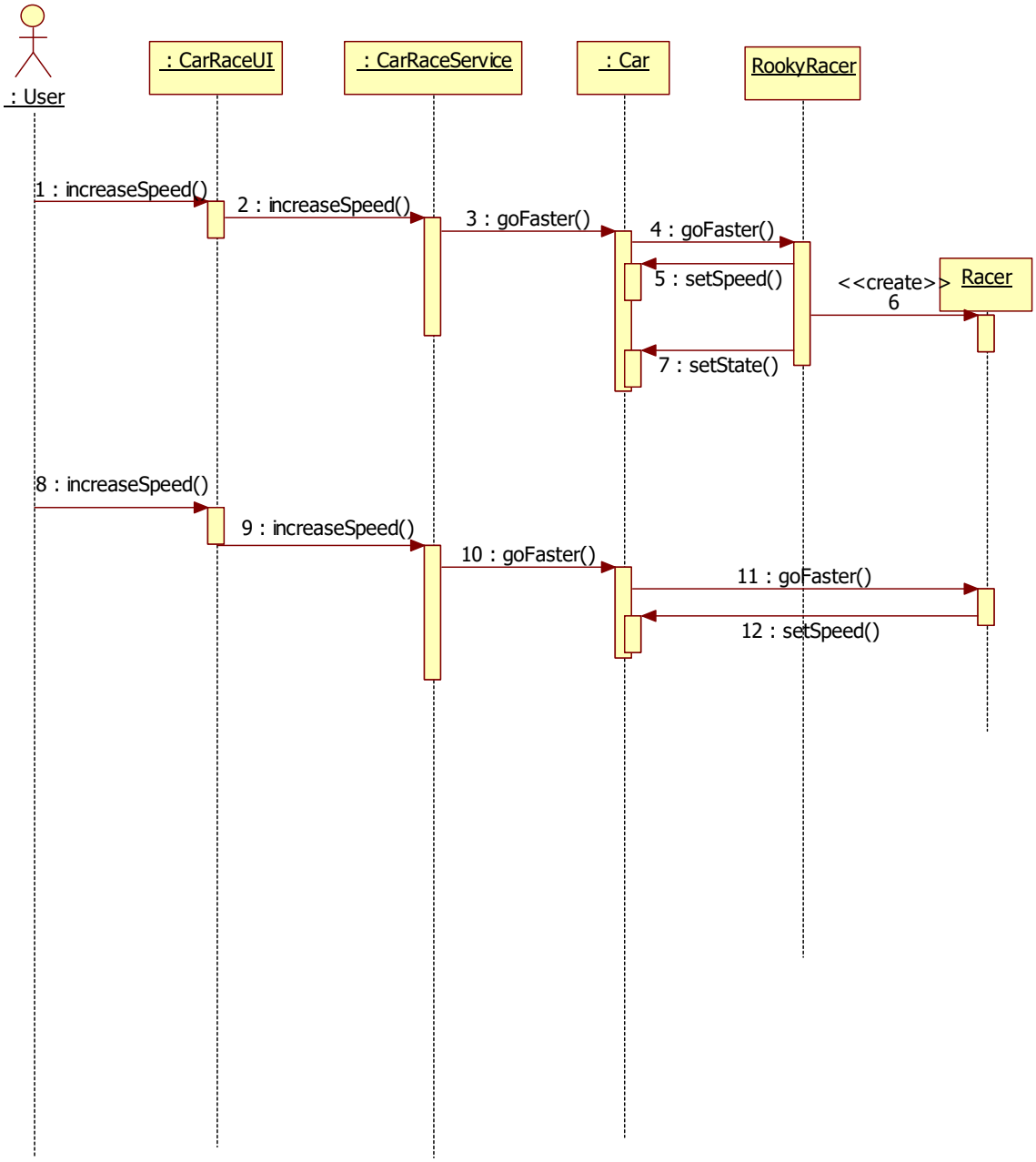
If the car has enough points, it will automatically be upgraded to the next node. The car can only be upgraded to a higher mode if the goFaster() or goSlower() method is called on the Car.

It should be easy to write racing applications using the framework that support different car modes. For this question we do **NOT**support undo/redo functionality.

Draw the sequence diagram of the following given scenario using your new design that supports different car modes:
Scenario:

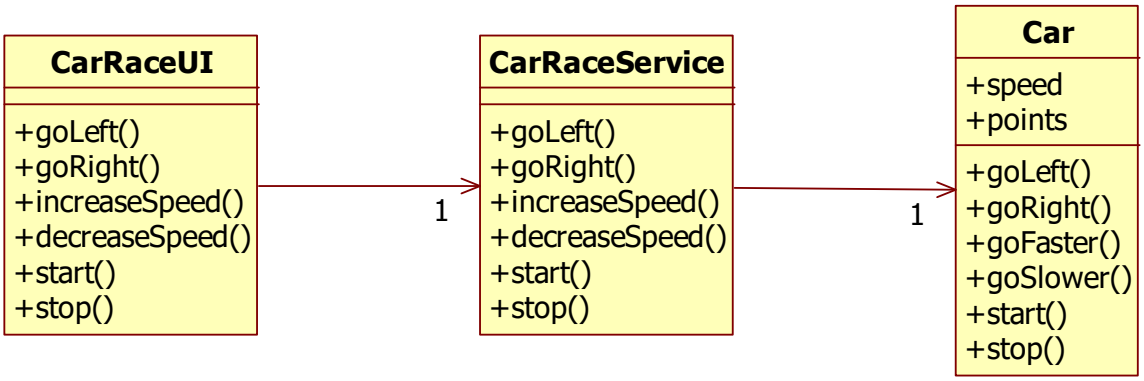
1. The scenario starts when the car is in Rooky Racer mode, and the car has 990 points.
2. The user clicks the increaseSpeed button and the speed goes up with 1 mile per hour. At the same time, the user gets 15 more points, and should be promoted to the Racer mode.
3. Then the user clicks the increaseSpeed button again and the speed goes up with 2 miles per hour.

Complete the partial given sequence diagram.


: User
: CarRaceUI
: CarRaceService
: Car
RookyRacer
Racer
1 : increaseSpeed()
2 : increaseSpeed()
3 : goFaster()
4 : goFaster()
5 : setSpeed()
<<create>>
6
7 : setState()
8 : increaseSpeed()
9 : increaseSpeed()
10 : goFaster()
11 : goFaster()
12 : setSpeed()

Question 3 [25 points] {30 minutes}

Suppose we have the same simple car racing game framework that we started with in question 1. Suppose we have the following **partial** class diagram of the framework:



But now we get the following new requirements:

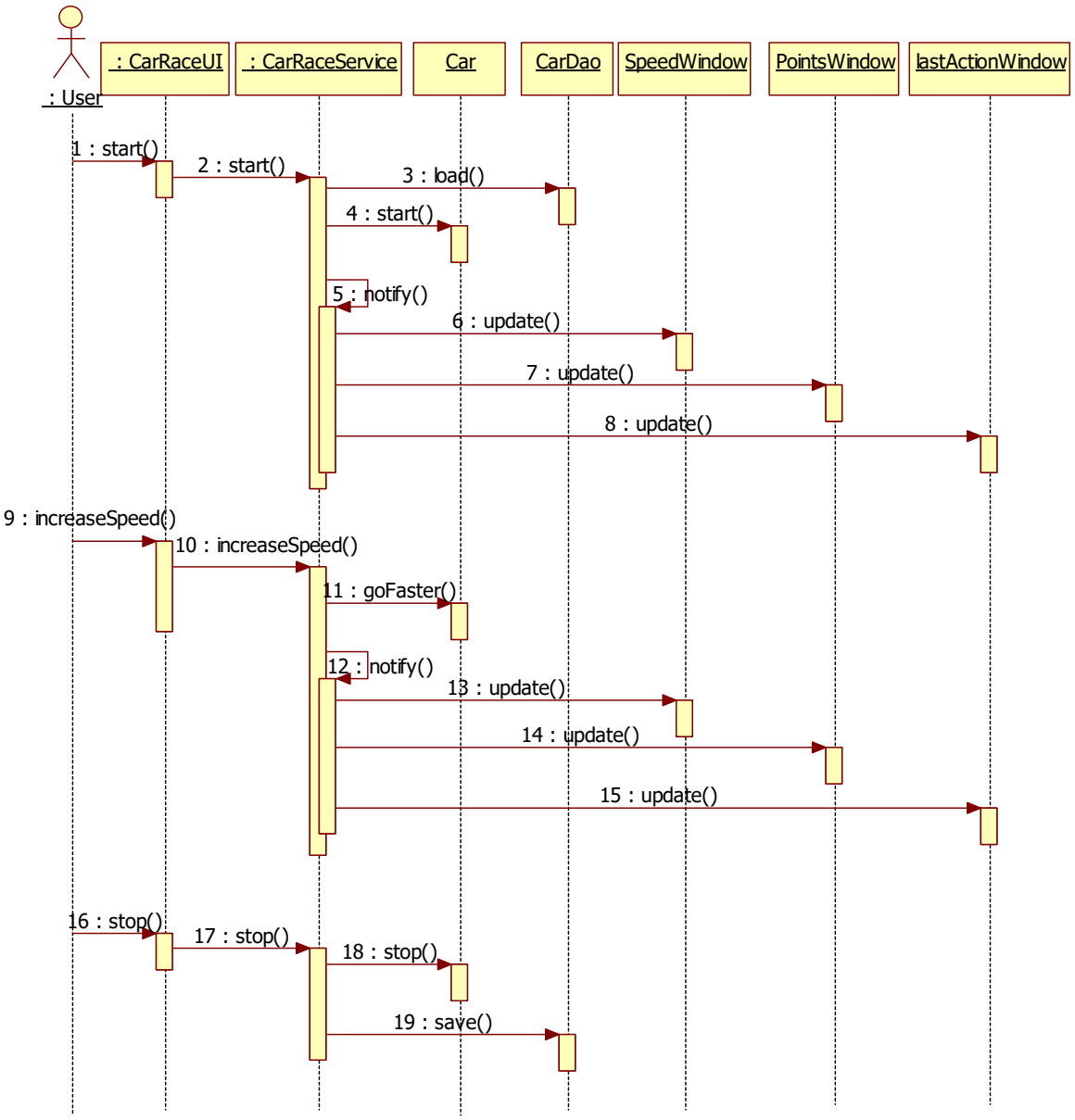
1. When we start the game by pressing the start button on the CarRaceUI, the car speed and number of points is loaded from the database. When we stop the game, the car speed and number of points is stored in the database.
2. In the racing game, we want to show all kind of car information (speed, points, current user action) on the screen in different window frames. The user can then choose which window frames will be shown on the screen. The framework should support the following window frames:
    - A window that shows the current speed of the car
    - A window that shows the number of points
    - A window that shows the last user action

It should be easy to write racing applications using the framework that support different window frames, for example a window that shows both the speed and the points next to each other. For this question we do **NOT** support undo/redo functionality, and we do **NOT** support different car modes.

Draw the sequence diagram of the following given scenario :

1. The user clicks the start button, and the start() method is called on the CarRaceUI. On the screen we see the following window frames:
    - A window that shows the current speed of the car
    - A window that shows the number of points
    - A window that shows the last user action
2. The user clicks the increaseSpeed button, and the different windows are updated with the correct data
3. Then the user clicks the stop button, and the stop() method is called on the CarRaceUI

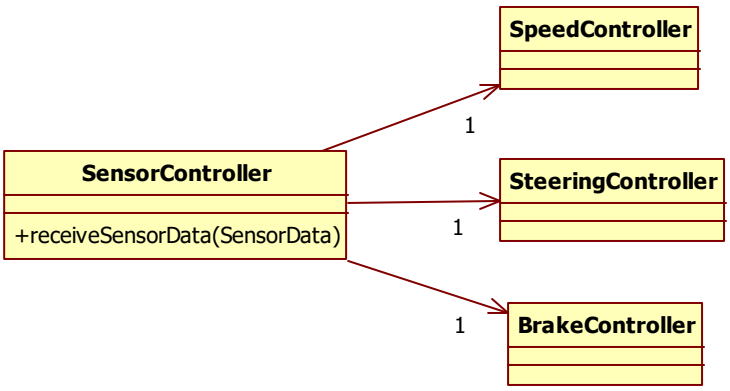
Complete the partial given sequence diagram.


: User
: CarRaceUI
: CarRaceService
Car
CarDao
SpeedWindow
PointsWindow
lastActionWindow
1 : start()
2 : start()
3 : load()
4 : start()
5 : notify()
6 : update()
7 : update()
8 : update()
9 : increaseSpeed()
10 : increaseSpeed()
11 : goFaster()
12 : notify()
13 : update()
14 : update()
15 : update()
16 : stop()
17 : stop()
18 : stop()
19 : save()

Question 4 [10 points] {15 minutes}

Suppose you need to design a framework that can be used for writing self-driving car software. Cars are full of cameras and sensors (temperature, speed, GPS location, etc) that give certain data records to the car software. There are also many different car parts (steering controller, speed controller, brake controller, etc) that need to act on these data records that are send by the cameras and sensors. The problem is that every car has different kind of sensors and cameras, and every car has different kind of controllers. One characteristic of this software is that a controller needs to know different data that comes from different cameras and/or sensors.

Our first partial design looks like this:



All cameras and sensors send their SensorData to the SensorController by calling the method receiveSensorData(). This SensorData is nothing more than a set of key – value pairs. The SensorController then knows which Controllers need to be called depending on the SensorData.

a. What is the problem with this solution?
   If we add a new controller, the SensorController has to change

b. Which pattern can we apply to solve this problem?
   Chain of Responsibility

Question 5 [5 points] {10 minutes}

Describe how Aspect Oriented Programming relates to one or more of the SCI principles you know. Your answer should be about half a page, but should not exceed one page (handwritten). The number of points you get for this question depends on how well you explain the relationship between Aspect Oriented Programming and the principles of SCI.

Your answer: