Comp 110-003 - Assignment 8: Arrays and Enums

Date Assigned: Mon Nov 14, 2011

Completion Date: Tues Nov 22 (midnight)

In this assignment, you will gain experience with arrays. In addition, you will strengthen your structured object skills. You do not need to write a console-based interface. Assume that all interaction will take place using ObjectEditor. *Each part of the assignment is illustrated using screen shots to help you visualize what the ObjectEditor windows will look like and show you how to interact with your program*.

The submission date assumes that by the end of the lecture on Nov 18, we will cover the parts of chapters 10, 11, 12, 13, and 14 that are needed for the assignment. If this does not happen, we will postpone the submission by two weekdays. The assignment does not cover any of the material in unit 15.

Part 1 – CarHistory

Implement a history of cars (`ACarHistory` class and `CarHistory` interface). The history holds a maximum of 30 entries. To get started, take a look at `StringHistory` and `PointHistory` examples in the notes. As in the case of `PointHistory`, ObjectEditor should be able to display the history of your cars in an edit window. Figure 1 below illustrates how to interactively add a car to your car history.

Part 2 – Rabbit

In this section, implement a rabbit (class and interface). For full points, your rabbit should be represented as rectangle that is 50 pixels wide and 100 pixels high. The rabbit should initially be 200 pixels to the right of the X axis and to the bottom of the Y axis. As was the case with `AVehicle`, ObjectEditor should be able to display your rabbit. ***If you make your rabbit look more realistic, you will receive a 10% bonus.***

Part 3 – Highway

Implement a highway class that contains a car history and a rabbit. The highway should also:

- Define the following three editable `int` properties: CarMoveDistance, PreviousCarDistance, and RabbitMoveDistance.
- Define an add operation that adds a new car to the car history, which takes two `int` parameters, the width and height of the new car. The new car should be PreviousCarDistance pixels **to the right** of the previous car in the history. In other words, the left-most edge of the new car should be PreviousCarDistance pixels away from the right-most edge of the previous car in the history. You are free how far from the Y-axis the cars are as long as they are all the same distance away. The easiest is simply not to worry about a Y offset for now. Also,
    - when adding the first car, assume that the left-most edge should be 0 pixels away from the Y-axis.
    - you should assume that the cars are added in such a fashion that they never collide with one another.
- Define an `enum` property called RabbitStatus in highway that can take one of two values: ALIVE and ROAD_KILL.

Figure 3 shows how an instance of highway is displayed in ObjectEditor implement the specifications in the above three bullets.

The highway should additionally:

- Define a method `isRabbitColliding` that updates the value of RabbitStatus to ROAD_KILL if the rabbit is colliding with the car; otherwise, it sets the value of RabbitStatus to ALIVE. Use the following to test for a collision:
    - If the bounding-rectangle of the rabbit is touching or overlapping with a bounding rectangle of the car, then the car and the rabbit are colliding; otherwise, they are not colliding.
    - ***BONUS: if you make a more realistic rabbit, then you can implement the following collision test for 10% more bonus***
        - The car and the rabbit are colliding if and only if any part of the rabbit is touching or overlapping with any part of the car in the ObjectEditor window for the highway. In other words, they are colliding if and only if you can see the collision in the ObjectEditor window. Note that with the default collision detection test, the rabbit and a car can collide without the collision being visually shown in the ObjectEditor window.

- Define an operation `moveAllCars` that moves all of the cars in the car history MoveCarDistance pixels **to the right**. It then calls `isRabbitColliding`.

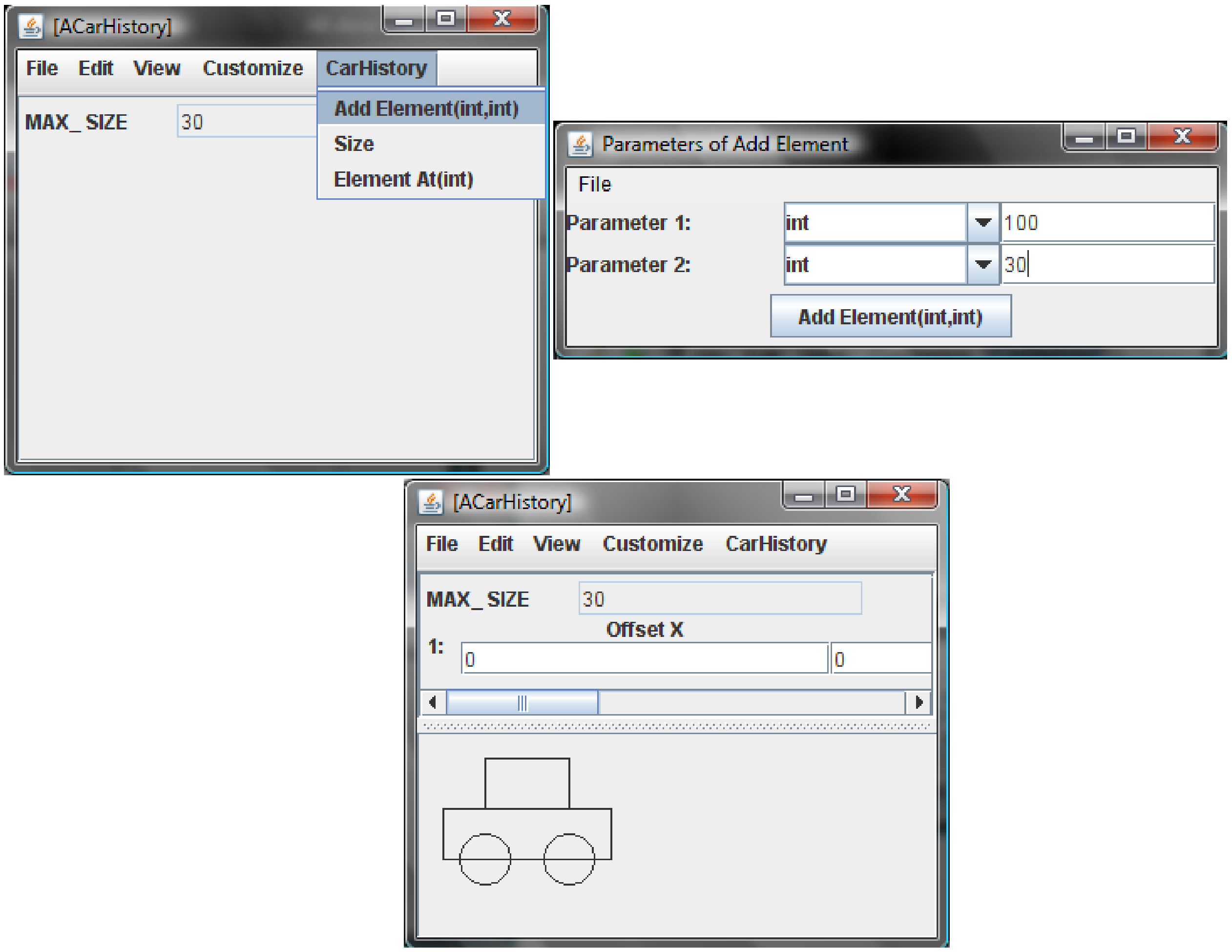
Figure 3 shows the highway after setting PreviousCarDistance to 50 and then adding the first (Figure 3 left) and second (Figure 4 right) cars, both of which are of width 100 and height 30. Figure 4 shows the highway after setting MoveCarDistance to 50 and calling `moveAllCars`.

- Define an operation `moveRabbit` that moves the rabbit RabbitMoveDistance pixels **up**. It then calls `isRabbitColliding`.

Figure 5 shows the effect of setting MoveRabbitDistance to 50 and invoking `moveRabbit` on the highway shown in Figure 4. Note that the rabbit status should be ROAD_KILL.

There are no “god bunnies” in allowed – in other words, it must be possible to position the cars and the rabbit in a way that results in ROAD_KILL as the rabbit status.

Hope everyone has fun!



MAX_ SIZE 30

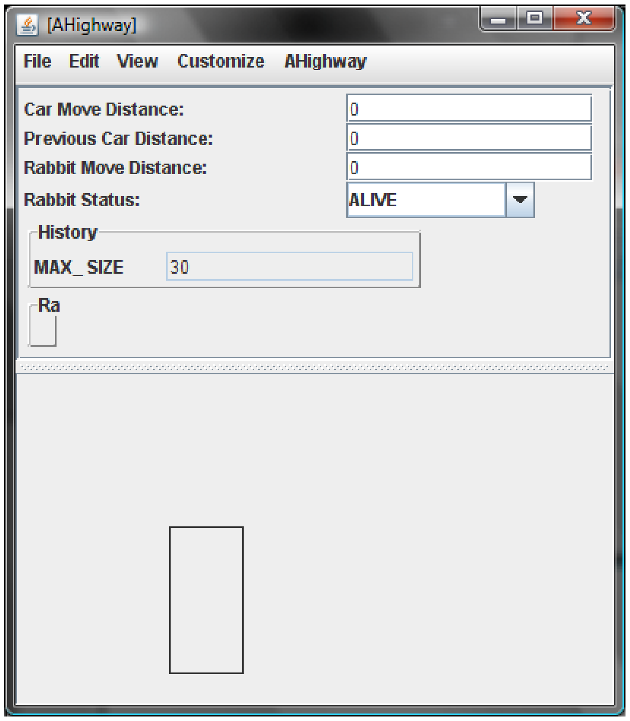
Parameter 1: int 100

Parameter 2: int 30

MAX_ SIZE 30

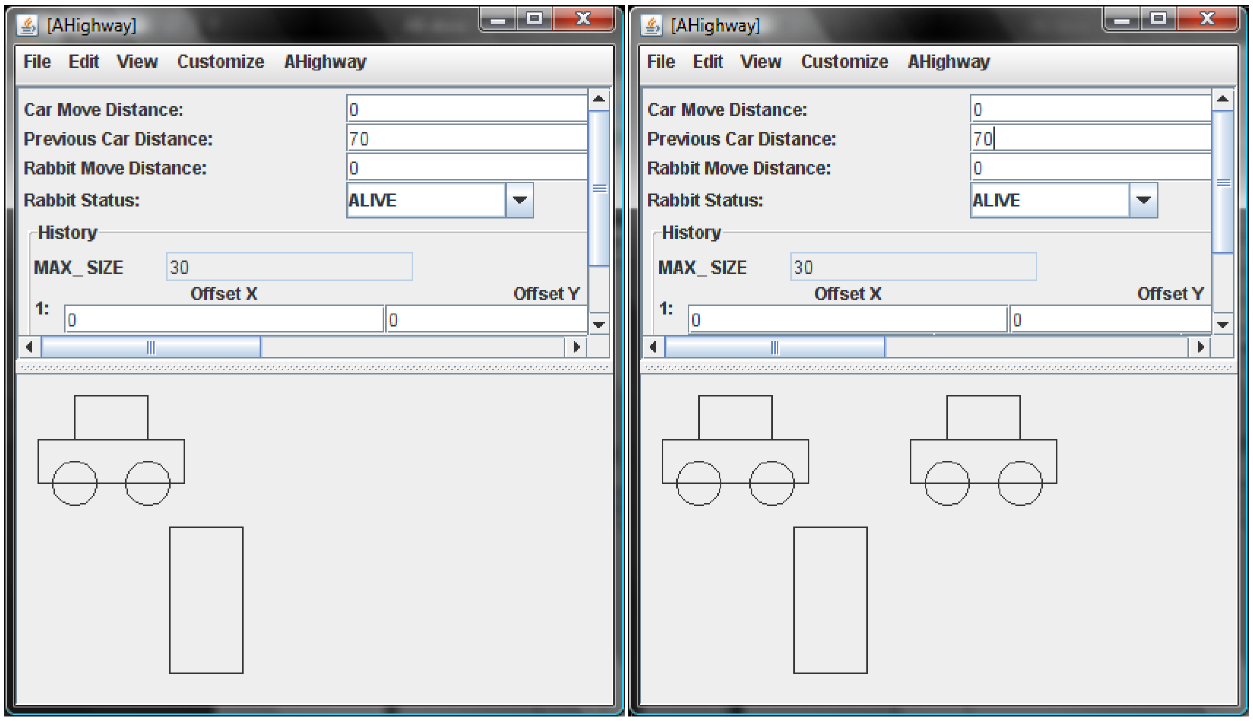
1: 0 0

Figure 1. Inserting a car into the history



Car Move Distance: 0
Previous Car Distance: 0
Rabbit Move Distance: 0
Rabbit Status: ALIVE
History
MAX_ SIZE 30
Ra

Figure 2. Initial instance of `AHighway` `with a Rabbit.`

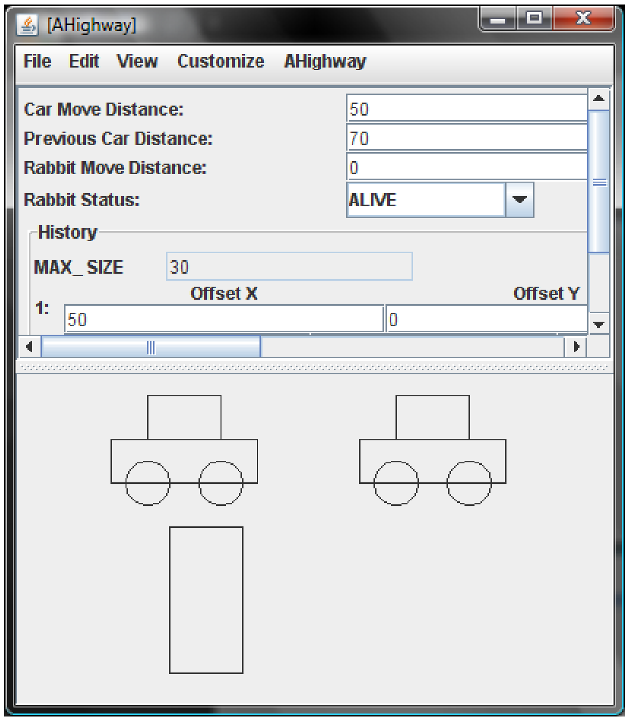

[AHighway]
File Edit View Customize AHighway

Car Move Distance: 0
Previous Car Distance: 70
Rabbit Move Distance: 0
Rabbit Status: ALIVE
History
MAX_ SIZE 30
Offset X Offset Y
1: 0 0

[AHighway]
File Edit View Customize AHighway

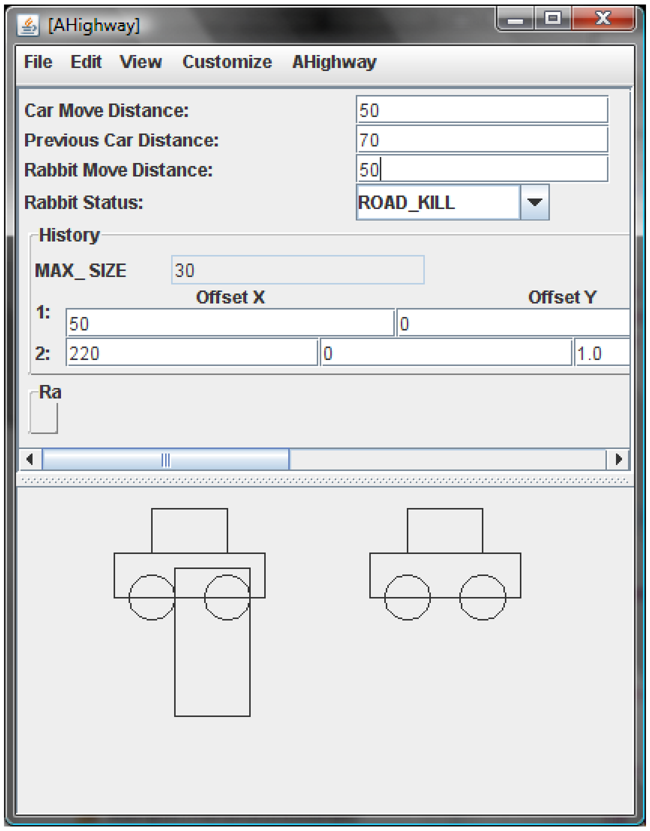
Car Move Distance: 0
Previous Car Distance: 70
Rabbit Move Distance: 0
Rabbit Status: ALIVE
History
MAX_ SIZE 30
Offset X Offset Y
1: 0 0

Figure 3. Highway after adding two cars, both with width 100 and height 30, assuming that the value of PreviousCarInstance is 70



Car Move Distance: 50
Previous Car Distance: 70
Rabbit Move Distance: 0
Rabbit Status: ALIVE
History
MAX_ SIZE 30
Offset X Offset Y
1: 50 0

Figure 4. Result of setting the value of MoveCarDistance to 50 and invoking `moveAllCars` after adding two cars as shown in Figure 5



Car Move Distance: 50
Previous Car Distance: 70
Rabbit Move Distance: 50
Rabbit Status: ROAD_KILL

History
MAX_ SIZE 30

	Offset X		Offset Y	
1:	50		0	
2:	220	0		1.0

Ra

Figure 5. Setting RabbitMoveDistance to 50 and invoking `moveRabbit` assuming highway was in state shown in Figure 4