# Array Review Program

A bicycle shop maintains information about bicycles it has in stock. A bicycle is represented by the Bicycle class. Information about the inventory of bicycles at the shop is stored in a BicycleInventory class, which contains a list of the bicycles in stock at the shop.

Your program must display your name before it displays anything else.

The details for each class are below:

Bicycle class:

Instance variables:

* Type, which describes the type of bike: mountain, speed, hybrid, road, dirt
* wheelSize, which represents the size of the wheel
* assembled, which represents whether the bike comes assembled
* gender, which is a single character- f or m

Methods:

* default constructor
* custom constructor that accepts all four instance variables
* getters for all four instance variables- this should be created automatically by Eclipse
* toString- should be automatically created by Eclipse, and then modified so that gender prints out the full gender, not just the character.

BicycleInventory class

Instance variables:

* a Bicycle array that stores each Bicycle in the inventory
* a variable that represents the size of the Bicycle array

Methods:

* default constructor that instantiates the array to a size of 100 and sets the variable size to zero
* readInventory- this method reads the data from the bicycle.txt file and stores it in the array.
* getChoices- this method will be overloaded multiple times to accept each of the four parameters, as well as a number. It will retrieve a specific number of bikes indicated by the number input. In other words, it will search through the list to find the bikes that meet the criteria indicated in the parameter but will only find as many as the number indicated in the parameter list. So if there are 5 bikes that meet the criteria, and the number in the parameter list is 3, it will return the first three bikes it finds. In addition to the four instance variables, you will create an additional overloaded method with all four instance variable values and the number, for a total of five getChoices methods.
* chooseOne- this method will also be overloaded multiple times with the same parameters as getChoices. It will call getChoices with the same parameters. This method returns a randomly selected bicycle from the list of bicycles provided from getChoices. There should be five chooseOne methods.
* Getters for the two instance variables. The method should return a copy of the array, not the original array.

BicycleDriver class:

Methods:

* main- this method will drive the flow of the program in a loop to process the users choice after displaying the menu of options.
* menuOptions- this method will display the users options:

1- Choose one for me

2- List inventory by wheelSize

3- List inventory by type

4- List inventory by gender

5- List inventory by assembled

6- List entire inventory

7- Quit

* determineSearch- this method displays the menu options if the user chooses option 1 from the main menu.

1- Choose one by wheelSize

2- Choose one by type

3- Choose one by gender

4- Choose one by assembled

5- Choose one by all

* determineSearchCriteria- this method will process the users choice after displaying the menu in determineSearch
* outputLIst- this method accepts a parameter that determines the search criteria used to filter output. For example, if the user chooses option 2 of the main menu, then you would pass a 1 to the outputList, which would then output all the data for a given size of wheel. This method should determine what the value is for the specific search field and then traverse the list and find all the bikes that satisfy that criteria.

The list of methods provided are simply a suggested approach. You may add or change any methods that you wish, in order to meet the criteria. However, the program must have the same menu options as indicated and every menu choice must work correctly. The output may be altered, as long as the same information is displayed. You must use arrays only to store your bicycle information. The bicycle.txt file is included. You must not alter this file in any way. You must overload methods and not give them separate names for each overloaded method.

Grading Criteria:

* (10 points) Methods appropriately overloaded
* (5 points) Program repeats until the user quits
* (7 points) Returns a single bike with the correct wheel size
* (7 points) Returns a single bike with the correct type
* (7 points) Returns a single bike with the correct gender
* (7 points) Returns a single bike with the correct assembly choice
* (7 points) Returns a single bike based on all four criteria
* (10 points) All bikes returned are randomly selected
* (4 points) Correctly displays a list of all bikes with selected wheel size
* (4 points) Correctly displays a list of all bikes with selected type
* (4 points) Correctly displays a list of all bikes with selected gender
* (4 points) Correctly displays a list of all bikes with selected assembly choice
* (4 points) Correctly displays a list of all bikes in inventory
* (5 points) All displays include gender written out.
* (5 points) Uses Javadoc comments
* (5 points) Uses internal comments
* (5 points) Outputs students name