Assignment 3: to be done at home, with no partner

Contents

[Assignment 3 of 3: to be done at home on your own (no partner) 2](#_Toc445045052)

[General Mark Deductions: 2](#_Toc445045053)

[LamborghiniCarLot 3](#_Toc445045054)

[Instance Variables 3](#_Toc445045055)

[Constructors 3](#_Toc445045056)

[Other methods 3](#_Toc445045057)

[BookStore 5](#_Toc445045058)

[Instance Variables 5](#_Toc445045059)

[Constructors 5](#_Toc445045060)

[Other methods 5](#_Toc445045061)

[LamborghiniDealershipGroup 6](#_Toc445045062)

[Instance Variables 6](#_Toc445045063)

[Constructor 6](#_Toc445045064)

[Other Methods 6](#_Toc445045065)

[Test Data for the classes 8](#_Toc445045066)

[Lamborghini objects to add to the ArrayList of Lamborghini 8](#_Toc445045067)

[Books to Add to the BookStore Array of Books 10](#_Toc445045068)

[LamborginiCarLot objects to add to LamborghiniCarDealership’s carLots ArrayList 11](#_Toc445045069)

[Test Class 12](#_Toc445045070)

[Submission Due Date and Requirements 12](#_Toc445045071)

# Assignment 3 of 3: to be done at home on your own (no partner)

Assignment 3 involves writing multiple classes, as described below. **Lates not accepted.**

**If your class does not compile, you will receive a mark of 0.**

**This is an individual assignment. Group work on this will be considered plagiarism!**

# General Mark Deductions:

|  |  |
| --- | --- |
| Requirement | Deductions if not done |
| Create a project named Assignment3 | -1 |
| Bad indentation or difficult-to-read/follow code | -1 each |
| Each public method must have a javadoc comment with an @param or @return tag (as appropriate) | -1 each |
| Your code has magic numbers. THERE WILL BE NO CAP ON DEDUCTIONS (50 magic numbers means -50 marks) | -1 each |
| Zip your file and named it as follows:  lastname-firstname-assignment-3-comp-1409.zip (use your own last name and first name: e.g. woods-tiger-assignment-3-comp-1409.zip)  …and upload it to BCIT's server before the due date (see last page). | -2 |
| Total mark | /91 |

This assignment consists of Three classes you will make:

1. LamborghiniCarLot
2. BookStore
3. LamborghiniDealershipGroup

**A few notes to start:**

* Use the classes provided in this folder as a start.
* If you can avoid writing the same method twice, do so.
* Each method must carefully consider when objects / collections may be null, and protect themselves against that possibility (e.g. before using a collection’s .size() method, ensure the collection is not null; before using a String’s .length() method, ensure the String is not null, etc.).

# LamborghiniCarLot

## Instance Variables

* private String lotName; // e.g. "Jason's Used Lambos"
* private ArrayList<Lamborghini> inventory; // collection of Lamborghini objects

## Constructors

* **public LamborghiniCarLot(){}** 
  + creates the inventory ArrayList object
  + **the inventory remains empty** (no Lamborghini objects are added when this constructor is called)
* **public LamborghiniCarLot(String lotName){}**
  + creates the inventory ArrayList object
  + stores the lotName parameter in the lotName instance variable, but only if it does not equal "Taylor's Used Lambos". If the parameter is "Taylor's Used Lambos", store the name as "Jason's Used Lambos" instead.
  + Adds the following Lamborghinis found [here](#_Lamborghini_objects_to_1) to the inventory

## Other methods

* **Standard accessor and mutator methods for instance variables.**
* **public void addLamborghini(Lamborghini newLamborghini)**
  + adds a new Lamborghini to the inventory
  + if the newLamborghini parameter is null, it is not added to the inventory.
* **public int howManyBetweenTheseYears(int startYear, int endYear)**
  + returns the number of Lamborghinis in the inventory whose model year is between these years, inclusive
  + use a foreach loop (NOT an iterator)
* **public int howManyAreRearWheelDrive()**
  + returns the number of Lamborghinis in the inventory that are rear-wheel drive
  + use an iterator (NOT a foreach loop)
* **public ArrayList<Lamborghini> getCarsFromThisYear(int year)**
  + returns an empty ArrayList if there are no Lamborghini objects for the year specified in the parameter
  + returns an ArrayList of all Lamborghini objects for the year specified in the parameter
* **public Lamborghini[] getCarsWithHorsepowerRange(double lowHP, double highHP)**
  + returns an **array** of all the Lamborghini objects that have horsepower in the range passed in as parameters (between lowHP and highHP).
  + If there are no cars that match the range specified, it return an empty Lamborghini array.
  + The array returned from this method is to be the exact size of how many matches there are (i.e. if there are 2 matches in the inventory of 10 cars, the return array size will be 2, NOT 10).
* **public boolean hasCarModel(String modelName)**
  + returns true if modelName exists in carLot
  + returns false if modelName is not found, or if carLot ArrayList is null
  + uses a case-insensitive search (e.g. I can pass in “diablo”, or “DIABLo”, and still find the same, correct model name)
* **public double getAverageHorsepowerOfYear(int modelYear)**
  + returns the average horsepower of all Lamborghini objects that match the modelYear specified as the parameter.
  + 0.0 is returned if no Lamborghini cars match the model year specified
  + the value returned by this method **MUST** be a decimal number (10 / 3 = 3.3333334, not 3).

# BookStore

## Instance Variables

* private String storeName; // e.g. "Jason's New Books"
* private Book[] inventory; // Array of Book objects

## Constructors

* **public BookStore(){}** 
  + creates the inventory Array of 20 Book object (see the list [here](#_Books_to_Add_1))
  + note that the list does not contain birth years nor death years for the authors, nor years published for the books…just use 2016 for all of these dates
* **public BookStore(){String storeName}**
  + creates the inventory Array of 20 Book object (see the list [here](#_Books_to_Add_1))
  + note that the list does not contain birth years nor death years for the authors, nor years published for the books…just use 2016 for all of these dates
  + stores the storeName parameter in the storeName instance variable, but only if it does not equal "Taylor's Used Books". If the parameter is "Taylor's Used Books", store the name as "Jason's Used Books" instead.

## Other methods

* **Standard accessor and mutator methods for instance variables.**
* **public int howManyBooksDidThisAuthorWrite(String authorLastName)**
  + returns the number of books written by an author whose last name is authorLastName
  + use a for loop
  + use .equalsIgnoreCase()
* **public String getAuthorFullName(String title)**
  + returns the full name of the author who wrote the book by this title
  + returns null if there is no Book with this title, or if title is null or ""
  + use a for loop
* **public Book[] getBooksWrittenBy(String authorLastName)**
  + returns null if there is no Book written by an author with authorLastName as their last name
  + otherwise, returns an Array of Book objects (hint: declare and return a local Book[] object) containing all Book objects which were written by an author with this last name (e.g. Orwell has two books in the list: 1984 and ANIMAL FARM)
* **public Book getBookWrittenBy(String firstName)**
  + returns the first Book object written by an author with this first name
* **public String[] getBookTitlesContaining(String wordInTitle)**
  + returns an array of Strings (the titles of all the books that contain wordInTitle in them)
  + (hint: declare and return a local String[] object)
  + returns null if there is no Book whose title contains wordInTitle, or if wordInTitle is null or ""
  + use the String class's .contains() method

# LamborghiniDealershipGroup

LamborghiniDealershipGroup defines a collection of LamborghiniCarLot classes.

## Instance Variables

* // e.g. Taylor’s Auto Group   
  private String dealershipGroupName;
* // A collection of LamborghiniCarLots   
  private ArrayList< LamborghiniCarLot> carLots;

## Constructor

* **public LamborghiniDealershipGroup() {}**
  + Creates carLots ArrayList object
  + Does **NOT** add any LamborghiniCarLot objects to carLots ArrayList
* **public LamborghiniDealershipGroup(String dealershipGroupName) {}**
  + Creates carLots ArrayList object
  + Creates the following objects found [here](#_LamborginiCarLot_objects_to).

## Other Methods

* **Standard accessor and mutator methods for instance variables.**
* **public void addCarLot(LamborghiniCarLot carLot)**
  + Adds the carLot provided to the carLots ArrayList
  + Does not add to the carLots ArrayList if carLot is null
* **public int getTotalInventoryCount()**
  + Using a foreach loop, gets the total inventory of car lots
  + Returns 0 if carLots ArrayList is null
* **public boolean hasModelName(String modelName)**
  + Returns true if the model name exists in ANY of the car lots
  + Returns false if the model name cannot be found, or if the carLots ArrayList is null
  + Uses a case-insensitive search (e.g. I can pass in “diablo”, or “DIABLo”, and still find the same, correct model name)
* **public LamborghiniCarLot[] getAllCarLotsWithAverageHorsepowerInRangeForYear(int modelYear, double lowHP, double highHP)**
  + returns an array of all the car lots whose average horsepower for a given year is within the range specified.
  + parameters are as follows:
    - modelYear – The model year to find the average horsepower for
    - lowHP – the lower end of the range of average horsepower
    - highHP – the higher end of the range of horsepower to find
  + The array returned from this method is to be the exact size of how many matches there are (i.e. if there are 2 matches in the arraylist of 10 carLot objects, the return array size will be 2, **NOT** 10).

# Test Data for the classes

## Lamborghini objects to add to the ArrayList of Lamborghini

inventory.add(new Lamborghini(1965, "350GT", 365.3, true));

inventory.add(new Lamborghini(1966, "400GT 2+2", 366.3, false));

inventory.add(new Lamborghini(1967, "MIURA", 367.3, true));

inventory.add(new Lamborghini(1968, "espada", 368.3, false));

inventory.add(new Lamborghini(1969, "islero", 369.3, true));

inventory.add(new Lamborghini(1970, "jarama", 370.3, false));

inventory.add(new Lamborghini(1971, "urraco", 371.3, false));

inventory.add(new Lamborghini(1971, "urraco", 371.3, false));

inventory.add(new Lamborghini(1973, "countache", 373.3, false));

inventory.add(new Lamborghini(1974, "silhouette", 374.3, false));

inventory.add(new Lamborghini(1975, "jalpa", 375.3, true));

inventory.add(new Lamborghini(1976, "silhouette", 476.3, false));

inventory.add(new Lamborghini(1977, "silhouette", 377.3, true));

inventory.add(new Lamborghini(1978, "countache", 478.3, false));

inventory.add(new Lamborghini(1978, "countache", 578.3, false));

inventory.add(new Lamborghini(1978, "countache", 378.3, true));

inventory.add(new Lamborghini(1979, "silhouette", 479.3, false));

inventory.add(new Lamborghini(1980, "countache", 580.3, false));

inventory.add(new Lamborghini(1981, "jalpa", 381.3, true));

inventory.add(new Lamborghini(1981, "jalpa", 481.3, true));

inventory.add(new Lamborghini(1982, "countache", 582.3, true));

inventory.add(new Lamborghini(1983, "countache", 383.3, false));

inventory.add(new Lamborghini(1984, "countache", 484.3, false));

inventory.add(new Lamborghini(1985, "countache", 585.3, false));

inventory.add(new Lamborghini(1986, "lm002", 386.3, true));

inventory.add(new Lamborghini(1987, "jalpa", 487.3, false));

inventory.add(new Lamborghini(1988, "countache", 588.3, false));

inventory.add(new Lamborghini(1989, "countache", 389.3, true));

inventory.add(new Lamborghini(1990, "diablo", 490.3, false));

inventory.add(new Lamborghini(1991, "lm002", 591.3, false));

inventory.add(new Lamborghini(1991, "lm002", 391.3, true));

inventory.add(new Lamborghini(1991, "lm002", 491.3, false));

inventory.add(new Lamborghini(1991, "lm002", 591.3, true));

inventory.add(new Lamborghini(1992, "lm002", 392.3, false));

inventory.add(new Lamborghini(1993, "lm002", 493.3, false));

inventory.add(new Lamborghini(1994, "diablo", 594.3, false));

inventory.add(new Lamborghini(1994, "diablo", 394.3, true));

inventory.add(new Lamborghini(1995, "diablo", 495.3, false));

inventory.add(new Lamborghini(1996, "diablo", 596.3, false));

inventory.add(new Lamborghini(1997, "diablo", 397.3, false));

inventory.add(new Lamborghini(1998, "diablo", 498.3, false));

inventory.add(new Lamborghini(1999, "diablo", 599.3, false));

inventory.add(new Lamborghini(1999, "diablo", 599.3, false));

inventory.add(new Lamborghini(2000, "diablo", 400.3, false));

inventory.add(new Lamborghini(2000, "diablo", 500.3, false));

inventory.add(new Lamborghini(2001, "murcielago", 501.3, false));

inventory.add(new Lamborghini(2002, "murcielago", 502.3, false));

inventory.add(new Lamborghini(2003, "gallardo", 303.3, true));

inventory.add(new Lamborghini(2004, "murcielago", 404.3, true));

inventory.add(new Lamborghini(2005, "murcielago", 505.3, true));

inventory.add(new Lamborghini(2006, "gallardo", 506.3, true));

inventory.add(new Lamborghini(2007, "gallardo", 507.3, true));

inventory.add(new Lamborghini(2008, "reventon", 308.3, false));

inventory.add(new Lamborghini(2008, "reventon", 508.3, true));

inventory.add(new Lamborghini(2009, "gallardo", 409.3, true));

inventory.add(new Lamborghini(2010, "murcielago", 310.3, false));

inventory.add(new Lamborghini(2011, "aventador", 411.3, false));  
inventory.add(new Lamborghini(2012, "sesto elemento", 512.3, true));

## Books to Add to the BookStore Array of Books

ULYSSES by James Joyce

THE GREAT GATSBY by F. Scott Fitzgerald

A PORTRAIT OF THE ARTIST AS A YOUNG MAN by James Joyce

LOLITA by Vladimir Nabokov

BRAVE NEW WORLD by Aldous Huxley

THE SOUND AND THE FURY by William Faulkner

CATCH-22 by Joseph Heller

DARKNESS AT NOON by Arthur Koestler

SONS AND LOVERS by D.H. Lawrence

THE GRAPES OF WRATH by John Steinbeck

UNDER THE VOLCANO by Malcolm Lowry

THE WAY OF ALL FLESH by Samuel Butler

1984 by George Orwell

I, CLAUDIUS by Robert Graves

TO THE LIGHTHOUSE by Virginia Woolf

AN AMERICAN TRAGEDY by Theodore Dreiser

THE HEART IS A LONELY HUNTER by Carson McCullers

SLAUGHTERHOUSE-FIVE by Kurt Vonnegut

ANIMAL FARM by George Orwell

OF HUMAN BONDAGE by W. Somerset Maugham

## LamborginiCarLot objects to add to LamborghiniCarDealership’s carLots ArrayList

//Create the first car lot

LamborghiniCarLot carlot1 = new LamborghiniCarLot();

carlot1.setLotName("Jason's vintage lambos.");

carlot1.addLamborghini(new Lamborghini(1965, "350GT", 365.3, true));

carlot1.addLamborghini(new Lamborghini(1966, "400GT 2+2", 366.3, false));

carlot1.addLamborghini(new Lamborghini(1967, "MIURA", 367.3, true));

carlot1.addLamborghini(new Lamborghini(1968, "espada", 368.3, false));

carlot1.addLamborghini(new Lamborghini(1969, "islero", 369.3, true));

carlot1.addLamborghini(new Lamborghini(1970, "jarama", 370.3, false));

carlot1.addLamborghini(new Lamborghini(1971, "urraco", 371.3, false));

carlot1.addLamborghini(new Lamborghini(1971, "urraco", 371.3, false));

carlot1.addLamborghini(new Lamborghini(1973, "countache", 373.3, false));

carlot1.addLamborghini(new Lamborghini(1974, "silhouette", 374.3, false));

carlot1.addLamborghini(new Lamborghini(1975, "jalpa", 375.3, true));

carlot1.addLamborghini(new Lamborghini(1976, "silhouette", 476.3, false));

carlot1.addLamborghini(new Lamborghini(1977, "silhouette", 377.3, true));

carlot1.addLamborghini(new Lamborghini(1978, "countache", 478.3, false));

carlot1.addLamborghini(new Lamborghini(1978, "countache", 578.3, false));

carlot1.addLamborghini(new Lamborghini(1978, "countache", 378.3, true));

carlot1.addLamborghini(new Lamborghini(1979, "silhouette", 479.3, false));

//Create the second car lot

LamborghiniCarLot carlot2 = new LamborghiniCarLot();

carlot2.setLotName("Taylor's brand new lambos.");

carlot2.addLamborghini(new Lamborghini(2006, "gallardo", 506.3, true));

carlot2.addLamborghini(new Lamborghini(2007, "gallardo", 507.3, true));

carlot2.addLamborghini(new Lamborghini(2008, "reventon", 308.3, false));

carlot2.addLamborghini(new Lamborghini(2008, "reventon", 508.3, true));

carlot2.addLamborghini(new Lamborghini(2009, "gallardo", 409.3, true));

carlot2.addLamborghini(new Lamborghini(2010, "murcielago", 310.3, false));

carlot2.addLamborghini(new Lamborghini(2011, "aventador", 411.3, false));

carlot2.addLamborghini(new Lamborghini(2012, "sesto elemento", 512.3, true));

//Add them to the arraylist

carLots.add(carlot1);

carLots.add(carlot2);

# Test Class

Like the previous assignments, you will be provided a testing class to make sure everything is working. This will be provided by Session 11. Until then please test everything on your own.

**Your assignment must compile and successfully run against the Assignment 3 testing tool.**

# Submission Due Date and Requirements

TO **SUBMIT THIS ASSIGNMENT** SEND FILES TO D2L Dropbox

**Late assignments are not accepted.**

**This assignment is due before 11:59pm the night before lesson 12.**