Jinshan Jiang

CIS 35B

21 March 2016

Professor Singh

**Error Corrections**

* **Assignment #1**

|  |  |  |  |
| --- | --- | --- | --- |
| Points Deducted | Correction | Location | Points Earn Back |
| -2  Packages not used | **package** driver;  **import** model.Automobile;  **import** util.ReadSource;  **import** util.Serial;  **import** adpter.\*;  **package** adpter;  **import** scale.\*;  **import** model.\*;  **import** util.\*;  **package** model;  **import** model.OptionSet.Option; | **package** driver  Driver  **package** adpter  ProxyAutomobile  **package** model  Automobile | I think I can earn full credit back for this one because now I’ve included all the package that the program needed to run. So, +2. |
| -1 Properties of Option set and Option are protected | **public** **class** OptionSet **implements** Serializable{  **private** **static** **final** **long** ***serialVersionUID*** = 1L;  **private** Option choice;  **private** ArrayList<Option> opt = **new** ArrayList<Option>(1);  **class** Option **extends** OptionSet{  **private** String nameOfSet;  **private** **static** **final** **long** ***serialVersionUID*** = 1L;  **private** String nameOfOption;  **private** **float** price; | **package** model  OptionSet Class  Option Class | All the properties now are protected. So, +1. |

* **Assignment #2**

|  |  |  |  |
| --- | --- | --- | --- |
| Points Deducted | Correction | Location | Points Earn Back |
| -1 You must submit a consolidated class diagram, showing relation between all the classes present. | 35B%20Java/Project_1_Jinshan_Jiang_Server/Assignment%20%233%20Design%20Diagram.pdf |  | +1 |
| -1 you must submit documentation explaining design choices | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Assignment #2 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  In this assignment, Building multiple interfaces in adapter package to help proxyauto to configure Automobile, and help to build a capsulation structure to hide all the structure about configuring the Automobile object.  To be able to do the capsulation, the bridge that I built for the external to access the internal code, I have BuildAuto, which implements all the interfaces but has nothing in the body, and all the methods declared in the interfaces are written in proxyAuto.  Another important goal to achieve in this assignment is to build a self-healing program. Therefore, I created a new package called Exceptions, which contains a ExceptionManager, and some helper class to dealing with different exceptions that occur in the program. After capture the exceptions, I have those helper classes to fix them; also, I have a method that will log them in to an Exceptional Log.txt file.  To be able to easy access the exception manager, I made it managed by numbers. To illustrate that, for model package, I used 100 to 199 to represents different exceptions; and 200 to 299 to represent exceptions from util package; and so on. Using this method can be easier to manage program, and also easier to pass parameter for fixing.  Program is tested in Drive, a new designed main method tested the interfaces and the capsulation works, exception log created and fixing methods are all working, result is correct. |  | +1 |
| -1 no exception logging mechanism implemented | **public** **void** **writetofile**(**int** errno)  {  **try** {  java.util.Date **date**= **new** java.util.Date();  File **file** = **new** File("Exception Log.txt");  *// if file doesn't exists, then create it*  **if** (!file.exists()) {  file.createNewFile();  }    FileWriter **fw** = **new** FileWriter("Exception Log.txt", **true**);  BufferedWriter **bw** = **new** BufferedWriter(fw);  bw.write(String.*valueOf*(**new** Timestamp(date.getTime())));  bw.newLine();  bw.write("Exception " + errno + " Captured!");  bw.newLine();  bw.newLine();  bw.close();  } **catch** (IOException **e**) {  e.printStackTrace();  }  } | **package** Exceptions  Class  ExceptionManager | **+1** |

* **Assignment #3**

|  |  |  |  |
| --- | --- | --- | --- |
| Points Deducted | Correction | Location | Points Earn Back |
| -1.5 documentation explaining design choices expected | In this assignment, Option Set Array is refactored to a ArrayList<OptionSet>, and also introduced to a new element which is choice. In the class Automobile, choice is a ArrayList<Option>, it also appear in OptionSet class as a property of Option. In Class OptionSet, it has a method called SetChoice, which will return a option that is chosen as choice. And the method is called in Automobile Class to store them in a ArrayList<Option>. Choice contains all the user selections, and it will store in the array for later total price calculation. |  | In my opinion, I think I can earn full credit back for this because now it is well written and clear explained the design of choice. So, +1.5 |

* **Assignment #4**

|  |  |
| --- | --- |
| Points Deducted | Correction |
| -2 object locking should be implemented in methods of automobile class |  |