CS1083 Assignment #6 – Fall 2021

Due: Wednesday, October 27th before 11:00pm (Atlantic time) in the Assignment 6 Desire2Learn submission folder. (See submission instructions below).

The purpose of this assignment is to gain practice handling exceptions and file I/O.

This assignment is to be done individually. If you have questions, direct them to a tutor/assistant during a help session in the "Faculty of Computer Science Student Success Centre" team or to your course instructor.

Adding Error Handling and File I/O to Assignment 4

For assignment 4 you designed and implemented a hierarchy of categories of vehicle that part of an inventory of a dealership. No exception handling was required, so the implementation would terminate with an exception if the input didn't have the correct format.

For this assignment you will modify a solution to assignment 4 (provided in D2L) to add file I/O and exception handling. You will also create a new application to merge two binary inventories (files of `Vehicle` objects).

InvalidVehicleException

The input data for trucks requires a field to indicate the size of the bed (short or standard). Cars require a field to indicate if the car has either a trunk ('T') or a hatchback ('H') and has either 2 or 4 doors. Create a new exception called InvalidVehicleException as a subclass of Exception, that will be thrown if the fields for the cars or trucks do not have the correct values. The Truck constructor should throw this exception if bed is not "short", "standard", or "long". The Car constructor should throw this exception if type is not 'T' or 'H' and door is not 2 or 4. Note: your program needs to recover from an InvalidVehicleException such that the other valid input is stored in the array.

File I/O

The `Driver` application should be modified so that it reads passenger information from a text file (such as the provided in `inventory.dat` and `inventory2.dat` in the assignment 6 student files) and writes the sorted inventory array to a **binary file** (in addition to printing it to standard output). **The elements of the array must be written one at a time to the file**. The name of the input file and the name of the output file should be passed as command line parameters, and the application should fail gracefully with an error message if these parameters are not included when the program is run.

Remember, the `Vehicle` object will have to be modified in order to write `Vehicle` objects to a binary file.

Handling Exceptions

The `Driver` application should handle the exceptions:

1. `FileNotFoundException`
2. `IOException`
3. `InvalidVehicleException`, which is thrown by the `Truck` and `Car` constructors. The message printed should indicate the reason the exception is thrown (i.e. which class threw it).

Test the driver with input that throws each of the exceptions you wrote. Describe each test case, and show the actual output.

Merging Vehicle Files

Write an application in a separate driver class called `MergeVehicles` that reads two binary files of sorted `Vehicle` objects, and prints the vehicles in sorted order to the standard output stream as text data (do not store the values in an array but print them directly to standard output as they are read from the file). The `EOFException` can be caught to determine when the end of a file is reached. If vehicles are the same type and the same price then it doesn't matter what order they are printed in.

Instructions continued on the next page…

NOTE:

- You are NOT permitted to store the `Vehicle` objects in a data structure (e.g. array, ArrayList, LinkedList, etc.), as this is not needed.
- Run your program with two binary files produced by the modified `Driver` application, using two sample input files (available via Desire2Learn):
  - `inventory.dat`
  - `inventory2.dat`

For this assignment, only an electronic submission is required. Instructions are on the next page.

Your electronic submission (submitted via Desire2Learn) will consist of two files:

i. a written report. This should begin with a title page. As always, your title page should include: the course (CS 1083), the assignment number (Assignment #6 in this case), your full name, and your UNB student number. That should be followed by each part clearly identified with a section heading. Include:

   a. the source code for the Vehicle, Car, Truck, Van, InvalidVehicleException, and Driver classes and the test cases (input and output) for the Driver class and discuss the testing done of exception handling
   b. the source code for the MergeVehicle class and the sample output of merging 2 binary files

   This written report should be prepared using a word processor; we recommend using Microsoft Word (i.e. create a .docx file for your report). Copy & paste your java source code & required output into the report document. Add appropriate headings for each part. Fix up the formatting where necessary, adjusting line breaks & page breaks to ensure that your document is easy to read. Use a monospaced font for your code to maintain proper indentation.) Once the report is complete and you've checked it all over, save the .docx file for your own records, and then **save a second copy in pdf format for submission**. (Note: Be sure to open that file in a pdf viewer to verify that the pdf was generated correctly.) The **SINGLE pdf file** containing your report will be submitted to the appropriate assignment submission folder on Desire2Learn. (It is important that you submit a pdf file and NOT the original Word document. This pdf will allow the marker to write comments directly on your work to give you better feedback.)

   Note: Please name this report as follows:
   YourName_As6_Report.pdf

ii. an archive file (.zip) that contains all your work for this assignment. Make sure that your archive includes **all source code** (.java files, input and output files - in case the marker wishes to compile & run your code). This archive should be submitted as a single file to the appropriate drop box on Desire2Learn.

Note: Please name this archive file as follows:
YourName_As6_Archive.zip