Project: Air Ticket – Part 3

Deliverables

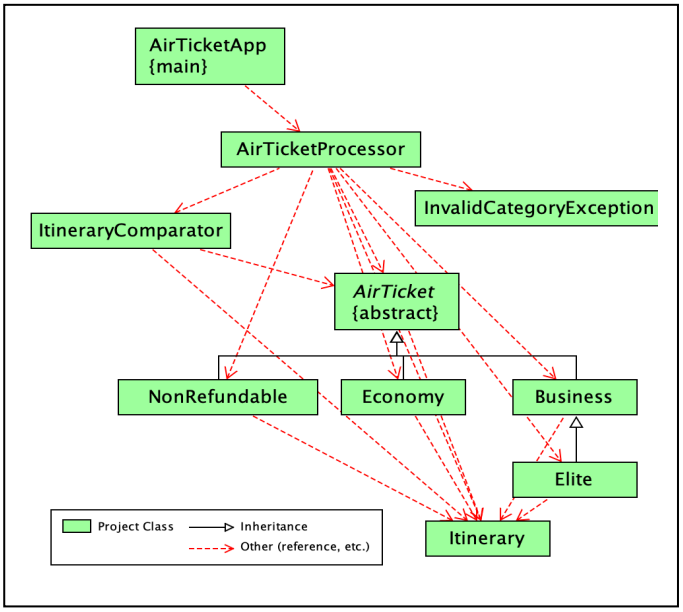
Your project files should be submitted to Web-CAT by the due date and time specified. Note that there is also an optional Skeleton Code assignment which will indicate level of coverage your tests have achieved (there is no late penalty since the skeleton code assignment is ungraded for this project). The files you submit to skeleton code assignment may be incomplete in the sense that method bodies have at least a return statement if applicable or they may be essentially completed files. In order to avoid a late penalty for the project, you must submit your completed code files to Web-CAT no later than 11:59 PM on the due date for the completed code assignment. If you are unable to submit via Web-CAT, you should e-mail your project Java files in a zip file to your TA before the deadline. Test files are not required for this project. If submitted, you will be able to see your code coverage, but this will not be counted as part of your grade.

Files to submit to Web-CAT (*test files are optional*):

Classes from Project 9
Itinerary.java, *ItineraryTest.java*
AirTicket.java
NonRefundable.java, *NonRefundableTest.java*
Economy.java, *EconomyTest.java*
Business.java, *BusinessTest.java*
Elite.java, *EliteTest.java*

Classes from Project 10
ItineraryComparator, *ItineraryComparatorTest*
AirTicketProcessor, *AirTicketProcessorTest*
AirTicketApp, *AirTicketAppTest*

New Classes in Project 11
InvalidCategoryException



Recommendations

You should create new folder for Part 3 and copy your relevant Part 2 source and optional test files to it. You should create a jGRASP project with these files in it, and then add the new source and optional test files as they are created.

Specifications – Use arrays in this project; ArrayLists are not allowed!

Overview: This project is the third of three projects that will involve the pricing and reporting of air tickets. The completed class hierarchy is shown in the UML class diagram above. Part 3 of the project focuses on handling exceptions that are thrown because of erroneous input from the command line or the data file. In the AirTicketApp class, the main method, which reads in the file name as a command line argument, will need to handle a FileNotFoundException that may result from attempting to open the file (e.g., if the file does not exist). Also, the readFile method in

AirTicketProcessor will need to handle exceptions that occur while processing the data file, including a new exception called InvalidCategoryException. You will need to do the following: (1) create a new class named InvalidCategoryException which extends the Exception class, (2) add try-catch statements to catch FileNotFoundException, InvalidCategoryException, and NumberFormatException, and (3) add three methods to the AirTicketProcessor class. As you develop and debug your program, you may find it helpful to use the "viewer canvas" feature in conjunction with the debugger or interactions.

- **Itinerary, AirTicket, NonRefundable, Economy, Business, Elite, ItineraryCompare,**

  Requirements and Design: No changes from the specifications in Project 10

- **InvalidCategoryException.java**

  Requirements and Design: InvalidCategoryException is a user defined exception created by extending the Exception class. This exception is to be thrown and caught in the readAirTicketFile method in the AirTicketProcessor class when a line of input data contains an invalid category. The constructor for InvalidCategoryException takes a single String parameter representing *category* and invokes the super constructor with the following String:

  ```
  "For category: " + "\"" + category + "\""
  ```

  This string will be the toString() value of an InvalidCategoryException when it occurs. For a similar constructor, see InvalidLengthException.java in 11_Exceptions\Examples\Polygons from this week's lecture notes.

- **AirTicketProcessor.java**

  Requirements and Design: In addition to the specifications in Project 10, three new methods, getTickets(), getInvalidInput(), and generateReportForInvalidInput(), should be implemented in the AirTicketProcessor class, and the existing readAirTicketFile method must be modified to recognize invalid input data and catch InvalidCategoryException and NumberFormatException.

  - `getTickets`: Accepts no parameters and returns the value of the field for the array of type AirTicket (i.e., the return type is `AirTicket[]`). Note that this method is primarily to facilitate testing.
  - `getInvalidInput`: Accepts no parameters and returns the value of the field for the array of type String containing the invalid input read from the file (i.e., the return type is `String[]`). Note that this method is primarily to facilitate testing.
  - `generateReportForInvalidInput`: Accepts no parameters and returns a String representing the Air Ticket Report for Invalid Input. The report is assembled by processing the invalid input array which contains the records in the order they were encountered. The printed Air Ticket Report for Invalid Input is shown below near the end in the Example Output.
  - `readAirTicketFile`: Has no return value and accepts the data file name as a String and throws FileNotFoundException. This method creates a Scanner object to read in the file and then reads it in line by line. I recommend that you set up a second scanner on the

line to scan it. For each *correct* line scanned, you will create the "appropriate" AirTicket object and add it to the array of AirTicket field. Each line in the file begins with a category for the ticket (N, E, B, and F are valid categories for ticket class indicating **N**onRefundable, **E**conomy, **B**usiness, and **F**irst (or Elite) respectively; note that **F** (First) will indicate our Elite class. For each *incorrect* line scanned (i.e., a line of data contains an invalid category or invalid numeric data), your method will need to handle the invalid items properly. If the line of data begins with an invalid category, your program should throw an InvalidCategoryException (see description above). If a line of data has a valid category, but includes invalid numeric data (e.g., the value for *miles* contains an alphabetic character), a NumberFormatException will be thrown automatically by the Java Runtime Environment (JRE). Your readAirTicketFile method should catch and handle InvalidCategoryException and NumberFormatException as follows. In each catch clause, a String object should be created consisting of

```
e + " in: " + line
```

where *e* is the exception and *line* is the line with the invalid data. The String object should be added to the array of invalid input.

- **AirTicketApp.java**

Requirements and Design: The AirTicketApp class contains the main method for running the program. In addition to the specifications in Project 10, the main method should be modified in Two ways.

(1) The main method should not include the throws FileNotFoundException in the declaration. Instead, the main method should include a try-catch statement to catch FileNotFoundException when/if it is thrown in the readAirTicketFile method in the AirTicketProcessor class. This exception will occur when an incorrect file name is passed to the readAirTicketFile method. This exception will be propagated from readAirTicketFile and caught in the main method, which will print the messages below and end.

*** File not found.
Program ending.

(2) In addition to the three reports in Part 2, AirTicketApp should also print the Air Ticket Report for Invalid Input as shown in the Example Output beginning on the next page. Note that the lines in the Invalid Input Report are shown "wrapped". However, in your output for Air Ticket Report for Invalid Input, these lines should not be wrapped. An example data file (air_ticket_data2.csv) can be downloaded from the Lab web page.

Example Output

Output when no file name is passed to main in as a command line argument.

```
----jGRASP exec: java AirTicketApp
*** File name not provided by command line argument.
Program ending.

 ----jGRASP: operation complete.
```

Output when a bad file name is passed to main in as a command line argument.

```
----jGRASP exec: java AirTicketApp bad_file_name.cvs
*** File not found.
Program ending.

 ----jGRASP: operation complete.
```

Output when `air_ticket_data2.csv` is passed to main in as a command line argument.

```
----jGRASP exec: java AirTicketApp air_ticket_data2.csv
----------------------------
Air Ticket Report
----------------------------

Flight: DL 1865
ATL-LGA (2021/11/21 1400 - 2021/11/21 1640) 800 miles (1600 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 2.0
Total Fare: $1,000.00 (class Business)
   Includes Food/Beverage: $50.00 Entertainment: $50.00

Flight: DL 1867
ATL-LGA (2021/11/21 1500 - 2021/11/21 1740) 800 miles (1200 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 1.0
Total Fare: $450.00 (class Economy)
   Includes Award Miles Factor: 1.5

Flight: DL 1863
ATL-LGA (2021/11/21 0900 - 2021/11/21 1140) 800 miles (1600 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 2.5
Total Fare: $1,325.00 (class Elite)
   Includes Food/Beverage: $50.00 Entertainment: $50.00
   Includes: Comm Services: $100.00

Flight: DL 1861
ATL-LGA (2021/11/21 0800 - 2021/11/21 1040) 800 miles (800 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 0.45
Total Fare: $182.25 (class NonRefundable)
   Includes DiscountFactor: 0.9

Flight: DL 1866
LGA-ATL (2021/11/21 1400 - 2021/11/21 1640) 800 miles (1600 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 2.0
Total Fare: $1,000.00 (class Business)
   Includes Food/Beverage: $50.00 Entertainment: $50.00

Flight: DL 1868
LGA-ATL (2021/11/21 1500 - 2021/11/21 1740) 800 miles (1200 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 1.0
Total Fare: $450.00 (class Economy)
   Includes Award Miles Factor: 1.5
```

```
Flight: DL 1864
LGA-ATL (2021/11/21 0900 - 2021/11/21 1140) 800 miles (1600 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 2.5
Total Fare: $1,325.00 (class Elite)
   Includes Food/Beverage: $50.00 Entertainment: $50.00
   Includes: Comm Services: $100.00

Flight: DL 1862
LGA-ATL (2021/11/21 0800 - 2021/11/21 1040) 800 miles (800 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 0.45
Total Fare: $182.25 (class NonRefundable)
   Includes DiscountFactor: 0.9

-------------------------------------
Air Ticket Report (by Flight Number)
-------------------------------------

Flight: DL 1861
ATL-LGA (2021/11/21 0800 - 2021/11/21 1040) 800 miles (800 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 0.45
Total Fare: $182.25 (class NonRefundable)
   Includes DiscountFactor: 0.9

Flight: DL 1862
LGA-ATL (2021/11/21 0800 - 2021/11/21 1040) 800 miles (800 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 0.45
Total Fare: $182.25 (class NonRefundable)
   Includes DiscountFactor: 0.9

Flight: DL 1863
ATL-LGA (2021/11/21 0900 - 2021/11/21 1140) 800 miles (1600 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 2.5
Total Fare: $1,325.00 (class Elite)
   Includes Food/Beverage: $50.00 Entertainment: $50.00
   Includes: Comm Services: $100.00

Flight: DL 1864
LGA-ATL (2021/11/21 0900 - 2021/11/21 1140) 800 miles (1600 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 2.5
Total Fare: $1,325.00 (class Elite)
   Includes Food/Beverage: $50.00 Entertainment: $50.00
   Includes: Comm Services: $100.00

Flight: DL 1865
ATL-LGA (2021/11/21 1400 - 2021/11/21 1640) 800 miles (1600 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 2.0
Total Fare: $1,000.00 (class Business)
   Includes Food/Beverage: $50.00 Entertainment: $50.00

Flight: DL 1866
LGA-ATL (2021/11/21 1400 - 2021/11/21 1640) 800 miles (1600 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 2.0
Total Fare: $1,000.00 (class Business)
   Includes Food/Beverage: $50.00 Entertainment: $50.00

Flight: DL 1867
ATL-LGA (2021/11/21 1500 - 2021/11/21 1740) 800 miles (1200 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 1.0
Total Fare: $450.00 (class Economy)
   Includes Award Miles Factor: 1.5

Flight: DL 1868
LGA-ATL (2021/11/21 1500 - 2021/11/21 1740) 800 miles (1200 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 1.0
```

```
Total Fare: $450.00 (class Economy)
   Includes Award Miles Factor: 1.5

--------------------------------------
Air Ticket Report (by Itinerary)
--------------------------------------

Flight: DL 1861
ATL-LGA (2021/11/21 0800 - 2021/11/21 1040) 800 miles (800 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 0.45
Total Fare: $182.25 (class NonRefundable)
   Includes DiscountFactor: 0.9

Flight: DL 1863
ATL-LGA (2021/11/21 0900 - 2021/11/21 1140) 800 miles (1600 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 2.5
Total Fare: $1,325.00 (class Elite)
   Includes Food/Beverage: $50.00 Entertainment: $50.00
   Includes: Comm Services: $100.00

Flight: DL 1865
ATL-LGA (2021/11/21 1400 - 2021/11/21 1640) 800 miles (1600 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 2.0
Total Fare: $1,000.00 (class Business)
   Includes Food/Beverage: $50.00 Entertainment: $50.00

Flight: DL 1867
ATL-LGA (2021/11/21 1500 - 2021/11/21 1740) 800 miles (1200 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 1.0
Total Fare: $450.00 (class Economy)
   Includes Award Miles Factor: 1.5

Flight: DL 1862
LGA-ATL (2021/11/21 0800 - 2021/11/21 1040) 800 miles (800 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 0.45
Total Fare: $182.25 (class NonRefundable)
   Includes DiscountFactor: 0.9

Flight: DL 1864
LGA-ATL (2021/11/21 0900 - 2021/11/21 1140) 800 miles (1600 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 2.5
Total Fare: $1,325.00 (class Elite)
   Includes Food/Beverage: $50.00 Entertainment: $50.00
   Includes: Comm Services: $100.00

Flight: DL 1866
LGA-ATL (2021/11/21 1400 - 2021/11/21 1640) 800 miles (1600 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 2.0
Total Fare: $1,000.00 (class Business)
   Includes Food/Beverage: $50.00 Entertainment: $50.00

Flight: DL 1868
LGA-ATL (2021/11/21 1500 - 2021/11/21 1740) 800 miles (1200 award miles)
Base Fare: $450.00 Fare Adjustment Factor: 1.0
Total Fare: $450.00 (class Economy)
   Includes Award Miles Factor: 1.5

--------------------------------------
Air Ticket Report for Invalid Input
--------------------------------------
InvalidCategoryException: For category: "A" in: A,DL 1865,ATL,LGA,2021/11/21
1400,2021/11/21 1640,800,450,2.0,50.0,50.00
java.lang.NumberFormatException: For input string: "800.0" in: E,DL
1867,ATL,LGA,2021/11/21 1500,2021/11/21 1740,800.0,450,1.0
```

```
java.lang.NumberFormatException: For input string: "$450" in: F,DL
1863,ATL,LGA,2021/11/21 0900,2021/11/21 1140,800,$450,2.5,50.0,50.00,100.00
java.lang.NumberFormatException: For input string: "0.90%" in: N,DL
1861,ATL,LGA,2021/11/21 0800,2021/11/21 1040,800,450,0.45,0.90%
java.lang.NumberFormatException: For input string: "50.x" in: B,DL
1866,LGA,ATL,2021/11/21 1400,2021/11/21 1640,800,450,2.0,50.x,50.00
java.lang.NumberFormatException: For input string: "1/0" in: E,DL
1868,LGA,ATL,2021/11/21 1500,2021/11/21 1740,800,450,1/0
java.lang.NumberFormatException: For input string: "100.oo" in: F,DL
1864,LGA,ATL,2021/11/21 0900,2021/11/21 1140,800,450,2.5,50.0,50.00,100.oo

 ----jGRASP: operation complete.
```

Due to space constraints, the lines in the Invalid Input Report are shown "wrapped" in the example above. However, in your output, these lines should not be wrapped. That is, each record in the invalid record array should be on a single line rather than broken/wrapped over two lines.