MIST 4600. Spring 2014. Dr. Janine E. Aronson. Assignment F

As always, your initial file is LastnameFirstname-Assignment*. Download the zipped BlueJ Project start folder, unzip it, and rename the unzipped folder with your Lastname and Firstname. When you finish this assignment, put your name and a comment about what works and what does not at the top of the main class where indicated. Save the BlueJ Project, close the BlueJ Project, zip the folder properly and submit the zipped file properly to eLC Assignment *. Note that if after you submit your file, you may submit an update until the due date. eLC will retain your most recent submission. Note that if the testers do not work, then you earn 0. Your code must interface properly with the testers.

I suggest that before you touch the keyboard, you look at and understand the existing code, and think about the task and how to accomplish it. Also, as soon as you make one or two changes to the code, recompile it.

You will need to start with the AssignmentF-Start project folder. In Windows, rename it.

The starting project has three methods **in the Store class**:

readFromMovieFile() reads from file, creates movie objects, and puts each movie object into the "movies" ArrayList.

printAllMovieInfoForThisMovie(int desiredIndex) prints out movie info for a particular movie by the index of the item in the ArrayList movies.

listMoviesWithWhile() uses a while loop to print out all movies in the ArrayList movies.

Your assignment is to add four new methods **in the Store class**, as follows. Make sure that you spell each one correctly and correctly set up any parameters with the correct type and name. If the tester does not work, I will not score it.

1 listAllMoviesWithIndices(). This is much the same as the listMoviesWithWhile() method, but you need to (1) add some headers to the printout, and (2) use a line of code such as the following before the main System.out.println statement used in listMoviesWithWhile():

```
System.out.print("   "   +  index + "  ")
```
[Note that the above is **not** System.out.println(" " + index + " ")]

The above statement will print out the value of the index, but then wait without spacing down a line for the next thisMovie.print(). In this way you will get, all on the same line, the index value for each movie before its info is printed out.

2 addANewMovie(int pMovieKey, String pMovieName, double pPrice). This takes the parameter values, creates a new Movie object, and then adds that new Movie object to

the end of the ArrayList. To add to the end of the ArrayList, you will need to use the "add(nameOfObject)" method with the "movies" ArrayList as was done in the readFromMovieFile method. Note that renterID, dueDate and timesRented are not in the parameter list. For new movies, it is assumed they have not yet ever been checked out by anyone. You will need to provide to the Movie constructor all the needed data as parameters. Note that this includes dueDate, renterID and timesRented, even though their values will all be zero. For the other parameters, choose your favorite movie or movies and make up any data needed. **Again, remember that the order of the parameters matters!**

Don't worry about duplicate movieKey values. We can consider that later.

3 removeMovie(int desiredIndex) This causes the movie that has that index to be removed (permanently) from the movies ArrayList. (Note that you will never be removing movies from the "MovieData.txt" file, so you can always get back to the starting position if you want.) You will need to use the ArrayList "remove(int index)" method to do this for ArrayList movies.

Protect your program from bombing. Make sure that if someone inters an index less than zero or bigger than the size of the ArrayList, you catch it and do not try to execute the change. Without this protection you could get an "index out of bounds" error. Try your program without this protection to see.

4 UpdateMoviePrice(int pIndex, double pNewPrice). This method allows you to change the price of a particular movie in your ArrayList. I recommend that you "get" the particular movie indicated by pIndex and put it into the holding memory location "thisMovie", and then change the price for that movie. Hint: the Movie class has a method called updatePrice(double pNewPrice).

You'll want similar protection against a bad index here as you put in the third new method above.

Test your program well. Put your name at the top of Store, tell me the outcome of your test. Include comments on every while, if, and other non-obvious line of code.

Note: Think before you start coding. This entire assignment can be done by adding very few code lines.