Project 4 – A “RedBox-like” program (continued)

Due Date
- At the beginning of the lab; see the schedule, last page of the syllabus

Before Starting the Project
- Review Chapters 13 and 15 as well as Chapters 8 – 10, 12, and 18 of the CIS163 book
- Read this entire project description before starting, if you have any question please ask the instructor

Learning Objectives
After completing this project, you should be able to:
- Use inheritance and polymorphism
- Implement your own double linked list
- Use advanced Swing components like `JTable` and `AbstractTableModel`
- Save and restore objects using text files

Program description: Your assignment is to continue working on a program that simulates a DVD rental store. As expected, you can rent out a DVD and a Game with your program. The rental store program can also return items (DVD and Game) with a cost calculated based upon the number of days the unit was rented.

A completed program must have the following functionality:
- Save and load the rental database with serialized files using JFileChooser
- Save and load the rental database with text files using JFileChooser
- Uses your own double linked list to store the DVD and Games in the RentalStore
- Rent a DVD with a bought date, due date, title and a renter’s name
- Rent a Game with a bought date, due date, title, renters name and the player used (Xbox, PS4, …)
- Returns a DVD and generate a cost for the rental (see Step 7 for details)
- Returns a Game and generate a cost for the rental (see Step 7 for details)
- Undo all actions except for save and load
- **<u>This program will be demonstrated in class to show the full functionality of the program, so attending class is very important</u>**.

NOTE: Since Project 4 is based on Project 3, the following steps SHOULD NOT BE COMPLETED until AFTER Project 3 has been turned in:
- **Step 13**
- **Step 14**

*Before you turn in your work: use the **Java Style Guide** to document your project. (10 pts)*

Step 10: Add on the following functionality to RentalStore class.

Looking at the RentalStore class properties, you will see: "*private LinkedList<DVD> listDVDs;*". Your task in this step is to convert LinkedList class into your own MyDoubleLinkList class that you create: *private MyDoubleLinkedList<DVD> listDVDs*. Important, MyDoubleLinkedList has a DNode, and DNode have next and previous pointers (references).

There several methods that you will need to create so that the RentalStore class will again function correctly with your new class. You must create the following methods:

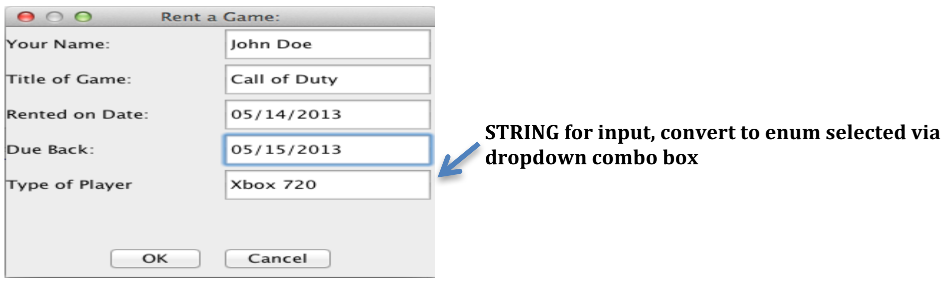
```
public class MyDoubleLinkedList<E> implements Serializable
{
    private DNode<T> top;     // DNode is a double linked list node.
    private DNode<T> tail;

    public int size;

    public MyLinkedList()
    public int size()
    public void clear ()          // remove all items
    public void add(T t)          //  add at the end.
    Public void addFirst (T t)    //  add at the top.
    public E remove(int index)    // remove first occurrence.
    public boolean removeAll(T t) // return true if at least one item removed
    public E get(int index)
    public int find (T t)         // return index if found, -1 otherwise
```

Step 11: Create a load and save button that will use text files (not serializable). Note that this functionality should not break saving or loading with serialized files.

Step 12: Create an undo button so all operations (except save and load) can be undone. You should allow for multiple undos.

Step 13: Change the RentGameDialog to use a dropdown combo box for player type. (Do NOT do this step before turning in Project 3.)

Here is a sample screen shot of a RentGameDialog:


Rent a Game:
Your Name: John Doe
Title of Game: Call of Duty
Rented on Date: 05/14/2013
Due Back: 05/15/2013
Type of Player Xbox 720
OK
Cancel
STRING for input, convert to enum selected via dropdown combo box

To get access to different types with in a enum type, the following piece of code may help you:

```
// Create the correct size array
PlayerType[] gameNames = new PlayerType[PlayerType.values().length];
for (PlayerType p: PlayerType.values())          // select each value;
        gameNames[i++] = p;
```

Step 14: Change from JList to JTable. (modify other classes as needed). (Do NOT do this step before turning in Project 3.)

Currently the public class RentalStore extends AbstractListModel change this to public class RentalStore extends AbstractTableModel {

Step 15: Create a final UML diagram of your completed program (Project 3 + Project 4, with all additions and changes. That is, this UML should be an EXACT representation of what a completed Project 4 should look like.)

--------------------------- YOU’RE DONE ☺ ---------------------------------

CIS 163 – Computer Science II
Project 4: "RedBox" Rental Program (continued)

Student Name	
Date Submitted, Days Late, Late Penalty	

Graded Item	**Pts**	**Points Secured / Comments**
Javadoc Comments and Coding Style/Technique (http://www.cis.gvsu.edu/java-coding-style-guide/) • Code Indentation (auto format source code in IDE) • Naming Conventions (see Java style guide) • Proper access modifiers for fields and methods • Use of helper (private) methods • Using good variable names • Header/class comments • Every method uses @param and @return (1 sentence after) • Every method uses a /***************** separator • Overall layout, readability, No text wrap • Using /** … / for each Instance variable • Has many inner "inner" comments	7	
• Step 10: Linked list • Step 11: Load and save a text file • Step 12: Undo button • Step 13: Combo box • Step 14: Use a JTable, not a JList	20 20 20 10 15	
Step 15: Final UML class diagram	8	
Total	**100**	

Comments: (extra credit)