Lab Exercise #7
Menus
Computer Science 2334

Group #: _____ Section #: _____

Members: _____

_____

_____

_____

_____

Learning Objectives:

- Understand how to create a menu system using *JMenuItem*, *Jmenu,* and *JMenuBar*.

The program you will work with implements a simple inventory system. The user can add and remove items from the collection. For this exercise, you will add a menu system to the inventory program that allows the user to load an inventory from a binary file, save the list of items a binary file, and exit the program through the menu.

Instructions:

This lab exercise requires a laptop with an Internet connection. Once you have completed the exercises in this document, your group will submit it for grading. All group members should legibly write their names at the top of this lab handout.

Make sure you read this handout and look at all of the source code posted on the class website for this lab exercise before you begin working.

1. Review the source code for the *ButterfingerGUI and ButterfingerListModel* classes.
2. There is no step 2.
3. Create a menu item for each menu option ('open', 'save', and 'exit') to be added to the program under a 'File' menu . The type of the menu item should be *JMenuItem*. These objects should be initialized in the constructor of the *MovieGUI* class.

   Should the references to these *JMenuItem* objects be stored as class variables or variables local to a specific method?

   To initialize the *JMenuItem*, the code will be ***similar*** to

   ```
   JMenuItem copyMenuItem = new JMenuItem( "Copy" );
   ```

4. Inside the constructor for the *MovieGUI* class you must also register an *ActionListener* on the *JMenuItems* by calling *addActionListener()* on each JMenuItem object. The *MovieGUI* class should be used as the class that implements the *ActionListener* interface.
5. Create a *JMenuBar* to hold the menus and a *JMenu* to hold the menu items under the 'File' menu. Be sure to add the 'File' menu to the menu bar. The following code shows how to do this step:

```
JMenu editMenu = new JMenu( "Edit" );
JmenuBar menuBar = new JmenuBar();
menuBar.add( editMenu );
```

6. Add each *JMenuItem* to the appropriate *JMenu* by calling the add method of *JMenu* and passing it a reference to the *JMenuItem*.

7. Now tell the *JFrame* of the program which *JMenuBar* object will serve as the menu. It is very important that you "set" the menu bar for the *JFrame* instead of trying to "get" a menu bar from the *JFrame* and then trying to add menus and menu items to the menu bar.
```
frame.setJMenuBar( menuBar ); // Be sure to use setJMenuBar() instead of getJMenuBar().
```

8. Add an *if*-statement for each menu item to the *actionPerformed*() method of *MovieGUI* that determines if the event occurred on the menu item. The contents of the if-statement should call *System.exit*(0), the *loadAction*() method, or the *saveAction*() method depending on the object the event occurred on.

9. Test the program to make sure it correctly responds to menu selection events and properly reads and writes data files.

10. Submit this lab electronically by transferring all of the required Java files to codd.cs.ou.edu and saving them in a folder named lab7. Then ssh into codd.cs.ou.edu and use the following commands to submit your lab.
```
cd lab7
/opt/cs2334/bin/submit cs2334-<section #> lab7 *.java
```
where <*section #*> is replaced by the section number for your lab section (011, 012).