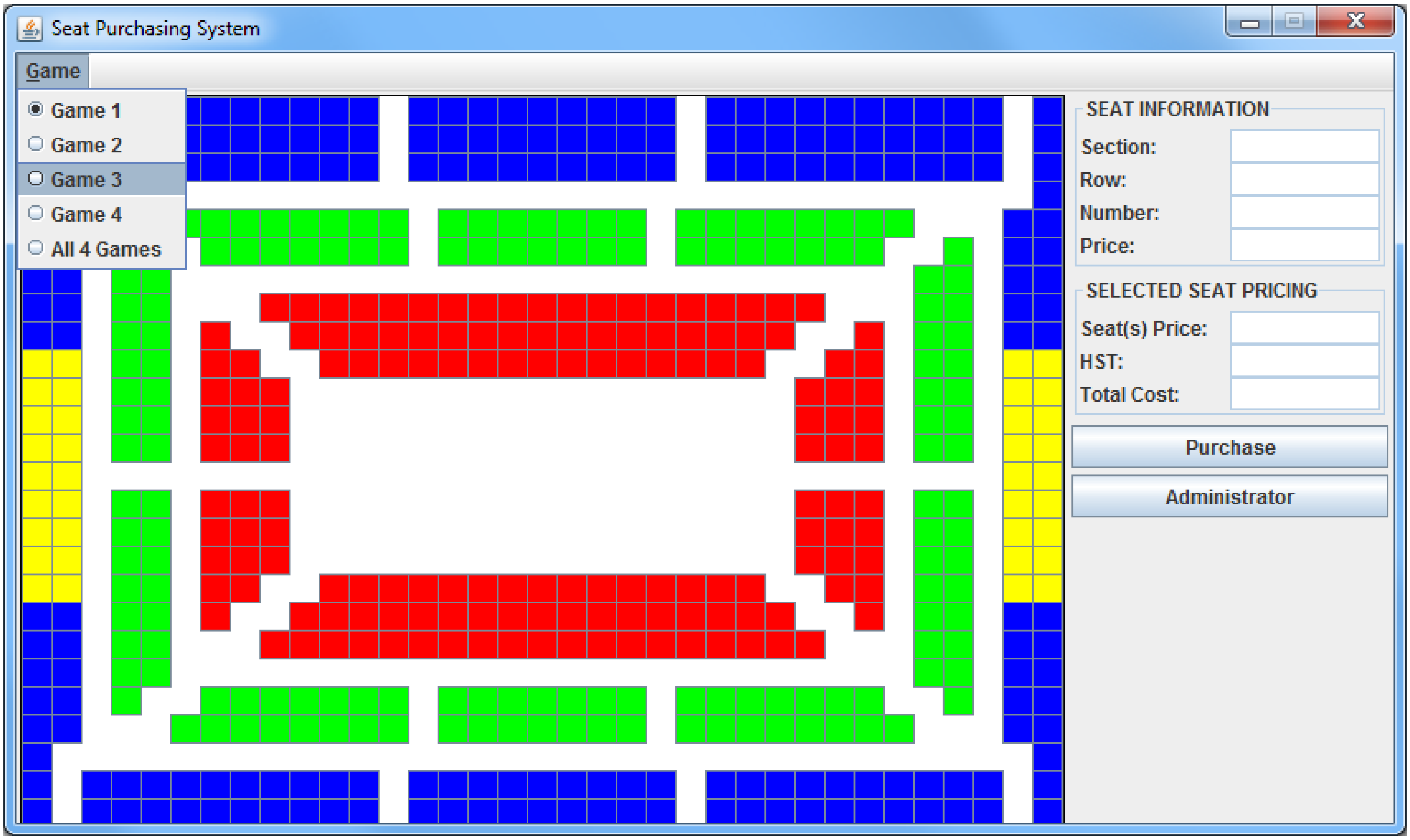
COMP1406 - Assignment #8

(Due: **Monday, March 17th @ 11:30pm)**

In this assignment, you will add a **menu bar** and **dialog boxes** to your Seat Purchasing System from the last assignment.

(1) The Games

Beginning with your code from the last assignment, add a JMenubar to your main window with a **Game** menu which contains 5 **JRadioButtonMenuItems** as "Game 1", "Game 2", "Game 3", "Game 4" and "All 4 Games". Here is what your interface may look like (note that the **Game** menu has been selected so you can see what it has inside it):



You will need to adjust your **Seat** class to show its availability for exactly 4 games. Each seat may be purchased for up to 4 games/events. Add, to the **Seat** class, a **boolean** array attribute called **sold** that keeps track of whether or not the seat has been sold for each of the exactly 4 games/events (i.e., the array will be of size 4). By default, a seat should not be sold for any of the 4 games. Therefore, when you start the application, the stadium should have all seats available for all 4 games.

Adjust your code so that when the **Purchase** button is pressed on the main application window, then the seats that are currently selected become sold and so they are no longer available for that particular game. Make sure that the following behavior is working:

1. purchased (i.e., unavailable) seats appear as **white** squares.

2. seats are purchased ONLY for the particular game that is selected in the menu.
3. if "All 4 Games" is selected from the menu, then the seats should be purchased for all 4 games and hence become unavailable for all 4 games.
4. If "All 4 Games" is selected, then the price for each seat currently selected should be multiplied by **4** and a **10% discount** (before taxes) should be applied.
5. if "All 4 Games" is selected from the menu, then only seats that are available (i.e., not sold) for **ALL** of the 4 games should be able to be selected and purchased. So if a seat has been sold even for just one game, then it should appear white when "All 4 Games" is selected from the menu.
6. whenever any game is selected from the menu, all currently **selected** seats should become **unselected**.
7. The **Purchase** button should be **disabled** if no seats are selected and enabled otherwise.

(2) Dialog boxes

Change your code so that when the **purchase** button is pressed, a dialog box appears to allow the user to enter purchasing information and to confirm the purchase. If the purchase is indeed made, then the availability of the seats changes (as you already wrote). However, if the user **cancels** the purchase, then no change to the seating availability should be made.

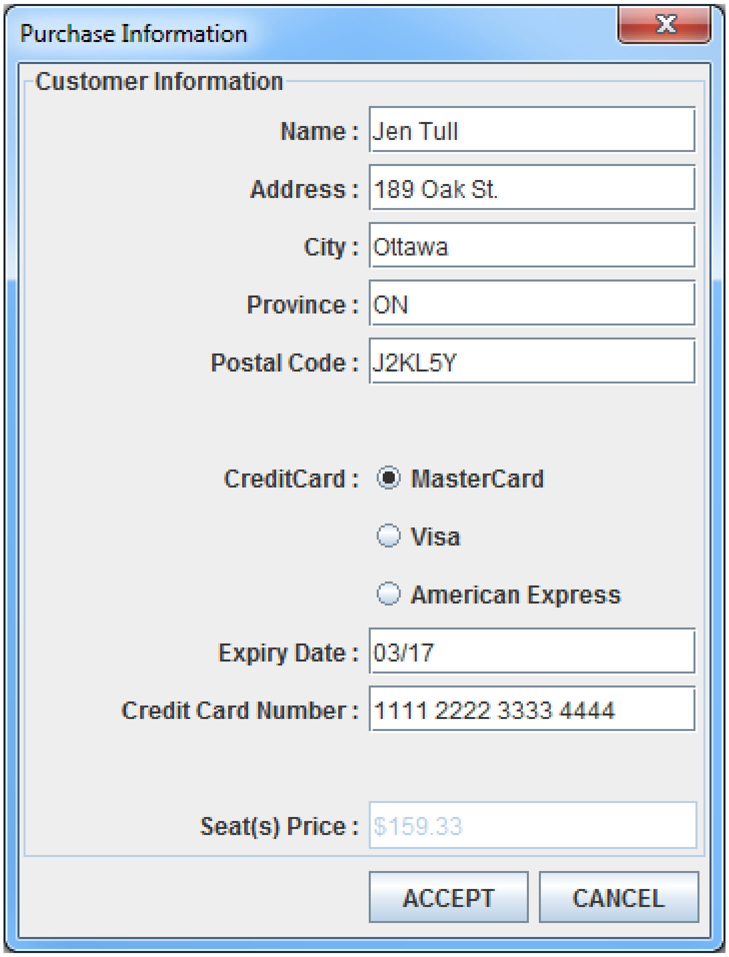
Create the dialog box in a class called **PurchaseDialog**. You should make use of the following **PurchasePanel** class.

```java
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class PurchasePanel extends JPanel {
   private JTextField nameTextField;
   private JTextField addressTextField;
   private JTextField cityTextField;
   private JTextField provinceTextField;
   private JTextField postalCodeTextField;
   private JTextField costTextField;
   private JTextField expiryDateTextField;
   private JTextField creditCardTextField;
   private String       creditCardType;

   // Make a get method so that the textfields can be accessed externally
   public JTextField getNameTextField() { return nameTextField; }
   public JTextField getAddressTextField() { return addressTextField; }
   public JTextField getCityTextField() { return cityTextField; }
   public JTextField getProvinceTextField() { return provinceTextField; }
   public JTextField getPostalCodeTextField() { return postalCodeTextField; }
   public JTextField getExpiryDateTextField() { return expiryDateTextField; }
   public JTextField getCreditCardTextField() { return creditCardTextField; }
   public JTextField getCostTextField() { return costTextField; }
   public String       getCreditCardType() { return creditCardType; }

   // Add a constructor that takes the price of the seat(s)
   public PurchasePanel (String cost) {
      // Make a nice border
      setBorder(BorderFactory.createTitledBorder("Customer Information"));

      // Create text fields
      nameTextField = new JTextField("");
      addressTextField = new JTextField("");
```

```
cityTextField = new JTextField("");
provinceTextField = new JTextField("");
postalCodeTextField = new JTextField("");
expiryDateTextField = new JTextField("");
creditCardTextField = new JTextField("");

// populate the costTextField with the price of the seat(s)
costTextField = new JTextField(cost);

// Disallow changing of price of the seat(s)
costTextField.setEnabled(false);

// Set the layoutManager and add the components
setLayout(new GridLayout(13,2,5,5));

JLabel aLabel = new JLabel("Name :");
aLabel.setHorizontalAlignment(JLabel.RIGHT);
add(aLabel);
add(nameTextField);

aLabel = new JLabel("Address :");
aLabel.setHorizontalAlignment(JLabel.RIGHT);
add(aLabel);
add(addressTextField);

aLabel = new JLabel("City :");
aLabel.setHorizontalAlignment(JLabel.RIGHT);
add(aLabel);
add(cityTextField);

aLabel = new JLabel("Province :");
aLabel.setHorizontalAlignment(JLabel.RIGHT);
add(aLabel);
add(provinceTextField);

aLabel = new JLabel("Postal Code :");
aLabel.setHorizontalAlignment(JLabel.RIGHT);
add(aLabel);
add(postalCodeTextField);

add(new JLabel(""));
add(new JLabel(""));

aLabel = new JLabel("CreditCard :");
aLabel.setHorizontalAlignment(JLabel.RIGHT);
add(aLabel);

ButtonGroup aButtonGroup = new ButtonGroup();
JRadioButton aButton;
aButton = new JRadioButton("MasterCard");
aButtonGroup.add(aButton);
add(aButton);
aButton.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
         creditCardType = e.getActionCommand();
      }});

add(new JLabel(""));
aButton = new JRadioButton("Visa");
aButtonGroup.add(aButton);
add(aButton);
aButton.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent e) {
      creditCardType = e.getActionCommand();
   }});
add(new JLabel(""));
```

```
          aButton = new JRadioButton("American Express");
          aButtonGroup.add(aButton);
          add(aButton);
          aButton.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                   creditCardType = e.getActionCommand();
                }});

          aLabel = new JLabel("Expiry Date :");
          aLabel.setHorizontalAlignment(JLabel.RIGHT);
          add(aLabel);
          add(expiryDateTextField);

          aLabel = new JLabel("Credit Card Number :");
          aLabel.setHorizontalAlignment(JLabel.RIGHT);
          add(aLabel);
          add(creditCardTextField);

          add(new JLabel(""));
          add(new JLabel(""));

          aLabel = new JLabel("Seat(s) Price :");
          aLabel.setHorizontalAlignment(JLabel.RIGHT);
          add(aLabel);
          add(costTextField);
       }
    }
```

The dialog box must have a **CANCEL** button and an **ACCEPT** button.   Here is an example of what your dialog box should look like:



Name : Jen Tull

Address : 189 Oak St.

City : Ottawa

Province : ON

Postal Code : J2KL5Y

CreditCard : ● MasterCard
○ Visa
○ American Express

Expiry Date : 03/17

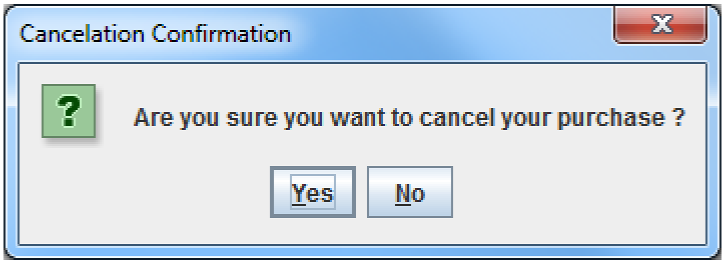
Credit Card Number : 1111 2222 3333 4444

Seat(s) Price : $159.33

If the user presses the **ACCEPT** button on the dialog box, the transaction must be accepted, which should cause the seats to be sold and an appropriate "receipt" printed to the **System.out** console showing all user-entered information from the dialog box as well as a list of all seat/game information for the purchases seats...along with the breakdown of the price paid....basically, it should display all possible available information from the dialog box as well as the seats as follows:

```
Confirmation of Purchase:
Name: Jen Tull
Address: 189 Oak St.
City: Ottawa
Province: ON
Postal Code: J2KL5Y
Credit Card Type: MasterCard
Expiry Date: 03/17
Credit Card Number: 1111 2222 3333 4444
Seat(s) Description for Game Number 2:
    Section: 2   Row: G   Number: 5   Price: $47.00
    Section: 2   Row: G   Number: 6   Price: $47.00
    Section: 2   Row: G   Number: 7   Price: $47.00
Seat(s) Price: $141.00   HST: $18.33   Total: $159.33
```

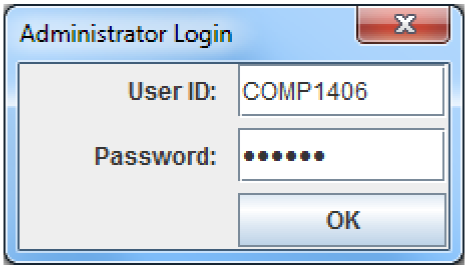
Then the dialog window should close.   However, if the user pressed the **CANCEL** button, then the following **JOptionPane** should appear:


Cancelation Confirmation

Are you sure you want to cancel your purchase ?

If the user then selects **No**, then the **JOptionPane** disappears and the purchase dialog box remains. If the user selects **Yes**, then the seats do not get purchased, the purchase dialog box closes and the main window should appear as it looked before the dialog box was opened.

If the user clicks on the **Administrator** button, an **AdminLoginDialog** box should appear. Here is the code, you just need to complete the **okButtonClicked()** method:

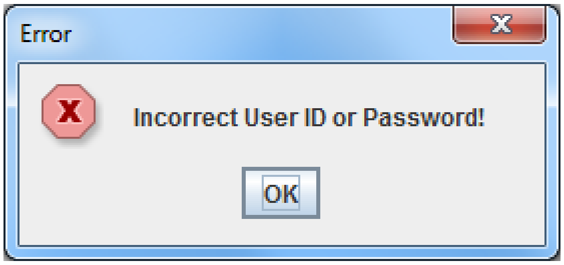

Administrator Login

User ID: COMP1406

Password: ••••••

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class AdminLoginDialog extends JDialog {
   private JButton         okButton;
   private JTextField      userTextField;
   private JPasswordField  passwordField;

   // The client (i.e. caller of this dialog box)
   private DialogClientInterface client;

   // A constructor that takes the model and client as parameters
   public AdminLoginDialog(Frame owner, DialogClientInterface cli){
      super(owner,"Administrator Login",true);
      client = cli;

      // Add the components using a grid layout
      setLayout(new GridLayout(3,3, 5, 5));

      JLabel aLabel = new JLabel("User ID:  ");
      aLabel.setHorizontalAlignment(JLabel.RIGHT);
      add(aLabel);

      userTextField = new JTextField("");
      add(userTextField);

      aLabel = new JLabel("Password:  ");
      aLabel.setHorizontalAlignment(JLabel.RIGHT);
      add(aLabel);

      passwordField = new JPasswordField ("");
      add(passwordField);

      add(new JLabel("")); // leave a blank
      add(okButton = new JButton("OK"));

      // Listen for ok button click
      okButton.addActionListener(
         new ActionListener() {
            public void actionPerformed(ActionEvent event){
               okButtonClicked();
      }});

      // Listen for window closing: treat like cancel button
      addWindowListener(
         new WindowAdapter() {
            public void windowClosing(WindowEvent event) {
               okButtonClicked();
      }});

      setSize(220, 120);
      setResizable(false);
   }

   private void okButtonClicked(){
      // This code needs to be written
   }
}
```

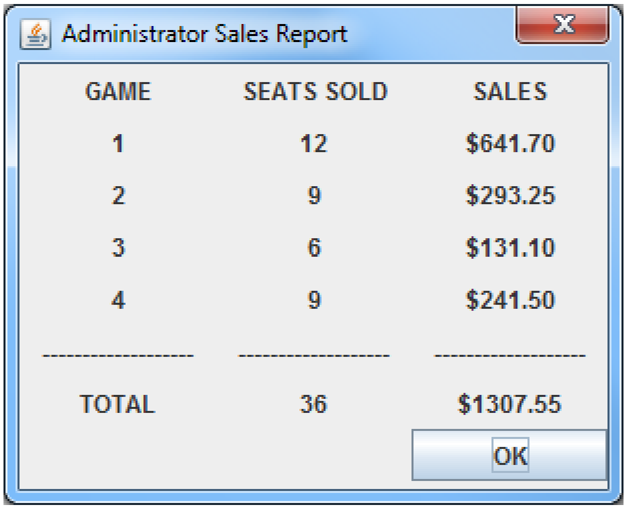
If the user enters the username "COMP1406" and password "SECRET" and then presses the **OK** button, then the dialog box should disappear and a new **SeatInfoDialog** box (see below) should appear showing information regarding all of the seats that were sold so far. If any other username or password has been entered, the **SeatInfoDialogBox** does not appear, but instead, the following **JOptionPane** should appear stating that the username and/or password was invalid --------------------------->



Incorrect User ID or Password!

Whether the login was valid or not, the **AdminLoginDialog** should disappear once **OK** is clicked.

Create the **SeatInfoDialog** class as a subclass of **JDialog** such that it displays information as follows:



GAME	SEATS SOLD	SALES
1	12	$641.70
2	9	$293.25
3	6	$131.10
4	9	$241.50
-------------------	-------------------	-------------------
TOTAL	36	$1307.55

The modal dialog box can use a 8 x 3 **GridLayout**, showing various **JLabels** (for the bottom row you can use two blank labels before the OK button). To obtain the number of seats sold per game and the total sales, you can simply count all seats in the Stadium that have been sold. Also, you can get the price of these seats, adding 13% HST. (Hint: Calculate these total first in two arrays of size 4 each...one for seats sold, one for sales... then make JLabels from that data). Make sure that the dialog box has one button on it at the bottom labeled **OK**, which when clicked closes the dialog box. There does not need to be a client for this dialog box since the main window does not need to be informed when this dialog box closes. Here is some code to get you started:

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class SeatInfoDialog extends JDialog {
      private JButton         okButton;
      private JTextField      userTextField;
      private JPasswordField  passwordField;

      public SeatInfoDialog(Stadium stadium){
            super((JDialog)null,"Administrator Sales Report",true);

            // Add the components
            setLayout(new GridLayout(8,3));

            JLabel aLabel = new JLabel("GAME");
            aLabel.setHorizontalAlignment(JLabel.CENTER);
            add(aLabel);
            aLabel = new JLabel("SEATS SOLD");
            aLabel.setHorizontalAlignment(JLabel.CENTER);
            add(aLabel);
            aLabel = new JLabel("SALES");
            aLabel.setHorizontalAlignment(JLabel.CENTER);
            add(aLabel);


            // ...
            // Replace the FOR loop below with your own code to add all the game stats
            for (int temp=0; temp<12; temp++) add(new JLabel(""));  // Leave blanks
            // ...

            aLabel = new JLabel("-------------------");
            aLabel.setHorizontalAlignment(JLabel.CENTER);
            add(aLabel);
            aLabel = new JLabel("-------------------");
            aLabel.setHorizontalAlignment(JLabel.CENTER);
            add(aLabel);
            aLabel = new JLabel("-------------------");
            aLabel.setHorizontalAlignment(JLabel.CENTER);
            add(aLabel);

            aLabel = new JLabel("TOTAL");
            aLabel.setHorizontalAlignment(JLabel.CENTER);
            add(aLabel);
            aLabel = new JLabel("???");
            aLabel.setHorizontalAlignment(JLabel.CENTER);
            add(aLabel);
            aLabel = new JLabel("???");
            aLabel.setHorizontalAlignment(JLabel.CENTER);
            add(aLabel);

            aLabel = new JLabel("");      // Leave a blank
            add(aLabel);
            aLabel = new JLabel("");      // Leave a blank
            add(aLabel);
            add(okButton = new JButton("OK"));

            // Prevent the window from being resized
            setSize(300, 240);
            setResizable(false);
      }
}
```

NOTE: Submit all **.java** and **.class** files needed to run. You **MUST NOT use packages** in your code, **nor projects**. Submit ALL of your files in one folder such that they can be opened and compiled individually in JCreator. Some IDEs may create packages and/or projects automatically. You **MUST** export the .java files and remove the package code at the top if it is there. Do NOT submit JCreator projects either. **JUST SUBMIT the JAVA and CLASS FILES**. Note that if your internet connection at home is down or does not work, we will not accept this as a reason for handing in an assignment late ... so make sure to submit the assignment WELL BEFORE it is due !

Please NOTE that you WILL lose marks on this assignment if any of your files are missing. You will also lose marks if your code is not written neatly with proper indentation. See examples in the notes for proper style.