Break A Plate:

I started by setting up my display. I added a JButton that said, “PLAY!!” for when the user wished to start the program. I added a JLabel to visualize the plates being broken. I added one JLabel that said, “You won: “ and another that displayed the prize the user won. A final JLabel is used to display the text, ‘Carnival Game,” to make the program more realistic. After making the display look pretty with my changing colours, font and sizing, I began coding the game.

I downloaded images of none plates broken, two plates broken, all plates broken, a sticker prize, and a plush prize, and imported them into the software.

To start the program, the user presses the Jbutton, “PLAY!!” This action starts a method which the rest of the code remains in.

At this time, the image of all three plates unbroken is displayed. The program then generates a random number between 0 and 2 using the code: int ran = (int)(Math.random() * 2); (“ran” is the name of the variable that will hold the value of the randomly generated number).

Using an if statement, the program tests the variable “ran” to see if the value is equal to one. If the value is one, the program displays the picture of all three plates broken on one of the JLabels, and then displays the plush image on another JLabel.

Using another if statement, the program retests the variable “ran”. If the value of ran is equal to 0, the program displays the image of two plates broken on one JLabel, and displays the sticker prize on another JLabel.

Local Bank:

I start this program off with formatting and adding elements to my display. I started off with a combo box so that the user can select their desired banking transaction. The following options are displayed: "Deposit", "Withdrawal", "Add Account", and "Delete an account." I then added a Jlabel that displays "Account number,” and below it a textfield for the user to input it. I added a Jlabel that displayed, “Amount of deposit/withdrawal,” and below that a textfield for the user to input that information. I then created Jlabels that displayed “First name” and “Last name,” along with text
fields below each, so that the user could input their name into the desired box. I added a blank Jlabel to display the output, and finally, a JButton that displays “Process Transaction.”

When this JButton is pressed, a new public void is created, and the remaining code is within it.

I first declare these variables to hold imputed values: String firstName;

```
String lastName;
String oldBalance;
String option;
```

```
int newBalance;
String message;
double nBalance;
```

I start by creating an if statement that tests the condition of the combo box. If the option selected is: Deposit, the variable “message” (used to describe the output) becomes equal to deposit information from the class provided by the teacher, ‘myBank.” Finally, this information is displayed in the empty Jlabel stating, “Your new balance is: “ the value of the variable, “message.”

The net if statement I created was to test if the user chose the “Add account” option on the combo box. If this selection was made, the value for the variable “nBalance” is equal to the double form of the originalBalance variable. The value for the variable “message” is then (coming from the “myBank” class created by the teacher) the first name, last name and account balance, all pulled from the second myBank class. The program then outputs this information to the user, saying, “Your new account is: “ and then the value of the message, which is pulled from the second class.

Finally, if the user wishes to use the “Delete an account” option, an if statement is activated. If this option on the combo box is selected, the value of “nBalance” is the value of “originalBalance” converted to a double data type. Then, the value of the variable “message” (using a string data type) is the value pulled from the “deleteAccount” section of the class “myBank.” This information is then output to the user using the code, “output.setText,” presenting the message. “Your account has been deleted.”

ticTacToe:

I start the program by inserting JButtons for each of the 9 squares that the user can press on their turn. I called the buttons, b1, b2, b3, b4, b5, b6, b7, b8, b9. The formatting of the buttons is the following:

B1, b2, b3
B4, b5, b6
B7, b8, b9

I then created a jLabel that would output who the winner is, named “winner.”

I then formatted all of the text, fonts and orientations to make them look pretty and professional.

The next thing I did was create a method for when the buttons are pressed. The first step in the method is testing if the button has been pressed before. If the button has been pressed, the program will ignore the input, so the same button can't be pressed twice in one game.

The next thing I did was create an if statement that would verify the possible solutions to win. It would continue to verify, and if there was a winner, the program would output: “(whatever player won) wins!!” and then disable the buttons. If there is no winner yet, the program would continue to switch the player’s turn. It does this by checking if the value of the variable “player” is equal to “x” using an if statement. If it is, it changes this value to O. If the value is not x (O) (using an else statement), the value of “player” turns to x.

To check for a winner, the code is split into three sections. One section checks for a horizontal win, another checks for a vertical win, and the final checks for a diagonal win.

Starting with rows (horizontal):

```
(b1.getText().equals(player) && b2.getText().equals(player) && b3.getText().equals(player)) ||
(b4.getText().equals(player) && b5.getText().equals(player) && b6.getText().equals(player)) ||
(b7.getText().equals(player) && b8.getText().equals(player) && b9.getText().equals(player)) ||
```

Each line checks a row. If all of the spots in the row = player (the one who wins will hold the value of player, so if they all equal player, they win) they win.

The same strategy is used for columns and diagonals.

Columns (vertical)

```
(b1.getText().equals(player) && b4.getText().equals(player) && b7.getText().equals(player)) ||
(b2.getText().equals(player) && b5.getText().equals(player) && b8.getText().equals(player)) ||
(b3.getText().equals(player) && b6.getText().equals(player) && b9.getText().equals(player)) ||
```

Diagonals

```
(b1.getText().equals(player) && b5.getText().equals(player) && b9.getText().equals(player)) ||
(b3.getText().equals(player) && b5.getText().equals(player) && b7.getText().equals(player))) {
```

When there is a winner, the JLabel “winner” displays who won by first printing the player who won, and then “wins!” (winner.setText(player + " Wins!");)

***MAYBE WRITE SOMETHING ABOUT THE LAST PART (DISABLES)