Project #4 (50 points)

Due Date

- Monday, November 16, by 11:59pm.

Submission

1. Group Submission (one copy per team)
   (a) You must designate a submitter (one of the team members) and **submit the zipped project folder** to Canvas.
   (b) You must include both team members' names in the comment block on top of EVERY Java file.
   (c) Your project folder must include the following **subfolders/files for grading**.
   - Source folder, including all Java files, 2 controller files and 2 .fxml files. [35 points]
   - JUnit test classes. [10 points]
2. Individual Submission (everyone must submit a copy)
   - Personal time log, using the template posted on Canvas. [5 points]

Project Description

You will be using JavaFX to develop a software system for ordering sandwiches. For simplicity, let's abstract away the details of the customer who is ordering the sandwiches, and abstract away the checkout functionality. However, your software must meet the following functional requirements. **-2 points** for each requirement not implemented.

1) The system shall provide the options of 3 types of sandwiches, Chicken, Beef and Fish. Each type of sandwiches has its basic ingredients. Customers can customize the sandwiches by adding extra ingredients to the sandwiches.
2) The default sandwiches selected on the GUI shall be set to Chicken.
3) Upon the selection of the sandwich type, an image of the sandwich shall be displayed on the GUI, together with the basic ingredients and the price of the sandwich.
4) The system shall provide a list of at least 10 extra ingredients for the customer to choose from. The customer can add or remove the extra ingredients.
5) Maximum number of extra ingredients added to the selected sandwich will be 6. The customer can also add the sandwich to the order as is, without adding any extra ingredients.
6) When the customer is adding the extra ingredients, the system shall not allow the same ingredients to be added more than once.
7) The system shall keep track of the sandwich price and display the price while the customer is adding and removing the ingredients.
8) The customer can add multiple sandwiches to an order. Each sandwich is identified by a serial number in the order; that is, each added sandwich is an order line on the order.
9) The system shall be able to show the order details with a list of sandwiches added in a new window.
10) For each sandwich in the order, the system shall print out the serial number (line number), sandwich type, the list of basic ingredients, extra ingredients and the price. At the end of the list, a total amount of the order shall be displayed.
11) In the order details window, the customer can
    - select a sandwich on the order and add the same sandwich to the order with a new serial number.
    - select a sandwich on the order and remove the sandwich from the order and reorder the serial numbers on the order (move everyone up.)
    - clear the order; that is, remove all the sandwiches on the order and start a new order; in this case, the serial number will be reset.
12) The system shall be able to save the order to a file, one order at a time.

Requirements

1. This is a **group assignment**. You MUST work in pair in order to get the credit for this program. You MUST follow the software development ground rules, or **you will lose points** for not having a good programming style.
2. You are required to log your times working on this project with the template provided on Canvas. **The time log is an individual assignment**. **You will lose 5 points** if the log is not submitted. If the times and comments are not properly logged, you will only get partial credits. You must type, handwriting is not acceptable.
3. Each Java class must go in a separate file. **-2 points** if you put more than one Java class into a file.
4. You can use any JavaFX components. However, you MUST include the following JavaFX components, or **-5 points** for each violation.
   (a) **ComboBox** for the sandwich type, Chicken, Beef and Fish.
   (b) **ImageView** for the sandwich images; in this case, you need 3 different images.
   (c) **ListView** – one for listing the extra ingredient options and one for the extra ingredients selected.
5. All input and output should be done on the GUI, you will **lose 10 points** for displaying any messages on the console with **System.out, OR** getting the input from the console.
6. You MUST use a 2nd Stage (window) to show the order details, which is a list of sandwiches added to the order. You will lose **10 points** if you do not implement a 2nd stage for the order details. The details of each sandwich should include the serial number, name of the sandwich type, the list of basic and extra ingredients, and the sandwich price. At the end of the order, you must display a running total of the order. **-1 point** for each item not displayed.
7. You MUST create a **second .fxml** and a **second controller** for the 2nd stage, or you will **lose 10 points.**
8. You MUST provide a way to "**Clear"** the order on the order detail window, and a "**Back**" button to close the 2nd Stage (window) to return to the original window. **-2 points** for each violation.
9. You can use the Java library class **ArrayList** to handle a list of objects. You MUST include the classes below, and you cannot change the signatures provided. **-5 points** for each class missing OR changed OR not used. You CANNOT add additional instance variables except the necessary static variables and constants. **-3 points** for each violation. Make sure you use good OO practices: encapsulation and polymorphism.

```
public interface Customizable {
   boolean add(Object obj);
   boolean remove(Object obj);
}

public class Order implements Customizable {
   public static int lineNumber; //reset for each new order;
   private ArrayList<OrderLine> orderlines;
}

public class OrderLine {
   private int lineNumber; //a serial number created when a sandwich is added to the order
   private Sandwich sandwich;
   private double price;
}

public abstract class Sandwich implements Customizable {
   static final int MAX_EXTRAS = 6;
   static final double PER_EXTRA = 1.99;
   protected ArrayList<Extra> extras;

   public abstract double price();
   public String toString() { }
}
```

10. You must define **Chicken, Beef** and **Fish** classes that extend the Sandwich class. **-5 points** for each subclass missing. Each subclass must implement the **price()** abstract method to return the price for the sandwich, and override the **toString()** method. **-3 points** for each method missing. The prices for different sandwich types and their basic ingredients are listed in the table below.

Sandwich Type	Basic Ingredient	Price	Extra Ingredients
Chicken	Fried Chicken, Spicy Sauce, Pickles	$8.99	$1.99 for each extra ingredient You can decide what the extra ingredients are, at least 10 options.
Beef	Roast Beef, Provolone Cheese, Mustard	$10.99	
Fish	Grilled Snapper, Cilantro, Lime	$12.99	

11. You must provide the functionality to save the order details to an external text file. The order details in the text file must be consistent with the order details show on the GUI. You will **lose 5 points** if this is not provided, and **lose 2 points** for any inconsistency between the order details on the GUI and the text file.

Program Testing

1. Your program must always run in a sane state and **should not crash in any circumstances**. In other words, the graders will try to produce exceptions in any way they can while running your GUI. You must catch all Java Exceptions and your program will continue to run until the user stops the program execution or closes the window. **You will lose 2 points** for each exception not caught with a **maximum of 10 points off.**
2. Create a **JUnit test class** for the **Order class.** You must **test all the public methods**. Use the Black-Box testing technique to design the test cases. You must show all test cases are passed. **This part is worth 10 points.**