Project #4 (100 points)

Due Date

Monday, November 15, by 11:59pm.

Submission

- You are responsible to **double check** your submissions in Canvas. If there are any missing files resulting the project doesn't compile or run, **you will get 0 points**.
- You must include both team members' names in the comment block on top of EVERY Java file.
- Your project folder must include the following **subfolders/files for grading**.
  (1) Source folder, including all packages you have with the following files [75 points]
    - All Java file, i.e., *.java; these include all Java classes and controller classes. The file names of the controller classes MUST contain the key word "Controller", or you will **lose 2 points**.
    - The fxml files; the file names of the fxml files MUST contain the keyword "View", or you will **lose 2 points**.

  (2) A JUnit test class for one of the subclasses of the abstract Pizza class. [10 points]
  (3) Class diagram [10 points]
  (4) Javadoc [5 points]

Project Description

RU Pizzeria sells pizzas. Your team will develop a software for the Pizzeria to manage the orders. Your team must use JavaFX to develop the GUIs for taking the orders, placing the orders, and cancelling an order. Currently, the Pizzeria offers 3 flavors of pizzas, Deluxe, Hawaiian and Pepperoni, with store customized toppings. However, the Pizzeria allows the customers to customize the toppings as well. The store staff will be the one using the software to take the orders from the customers. Each order is uniquely identified by the customer's 10-digit telephone number. For simplicity, the system will not keep track of the dates of the orders. The store manager has given you the list of requirements as follows.

(1) The Pizzeria offers 3 different sizes of pizzas, small, medium, or large, for each flavor.
  - Deluxe pizza includes 5 toppings, the price for a small size is $12.99.
  - Hawaiian pizza includes 2 toppings, the price for a small size is $10.99.
  - Pepperoni pizza includes 1 topping, the price for a small size is $8.99.
  - Add $2.00 for each size increase
  - Add $1.49 for each additional topping, up to 7 toppings maximum.
  - Pizza prices do not decrease if the number of toppings dropped below the number of store customized toppings.

(2) The system shall allow the store staff to choose the pizza flavors with different sizes.
(3) Upon the selection of the pizza flavor, the system shall display the image of the selected pizza, the list of store-customized toppings, a list of available toppings for customization, and a list of sizes to choose from.
(4) The store staff shall be able to customize the selected pizza by adding or removing the toppings and the system shall display the running subtotal with 2 decimal places.
(5) The system shall allow the store staff to add multiple pizzas to the same order and remove selected pizzas from the order.
(6) The store staff shall be able to check the detail of the current order before placing the order. These shall include the list of pizzas with the selected toppings, subtotal for each pizza, total amount of the pizzas in the order, sales tax amount and the order total, which is the total amount plus sales tax. The tax rate is 6.625%.

(7) The system shall be able to keep track of all the store orders, allow the store staff to browse the store orders and cancel an order. These shall include displaying all the store orders by the customers' phone numbers, the order total for each order with 2 decimal places, and the list of pizzas in each order.

(8) The system shall be able to export the store orders and save them in a text file, which includes a list of store orders. Each store order shall include the customer's phone number, the list of pizzas with selected toppings and the order total.

Project Requirements

1. This is a **group assignment**. You must work with your designated partner in order to get the credit for this project.
2. You MUST follow the software development ground rules, or **you will lose points** for not having a good programming style.
3. All methods should not exceed 40 lines, or **-2 points** for each violation, **maximum 5 points off**.
4. Each Java class must go in a separate file. **-2 points** if you put more than one Java class into a file.
5. Your software must provide 4 GUIs listed below, **-5 points** for each GUI missing. The user can navigate between the GUIs to add/remove pizzas to/from an order, review/place the order, check all orders, and cancel an order.
   - **Main Menu**, which shall include the options of ordering different flavors of pizzas, checking the current order, and managing the store orders placed by the user.
   - **Pizza customization**, which displays the image of the chosen pizza flavor, sizes to choose from, a list of preset toppings, a list of additional toppings, the running total of the pizza while the user is customizing the toppings. You MUST use this GUI for all pizza flavors. If you create a GUI for each pizza flavor, you will **lose 10 points.**
   - **Current order**, which displays the phone number, the list of pizzas with selected toppings, subtotal, sales tax, and order total. All dollar amounts must be displayed with 2 decimal places. The user can review the order, remove the selected pizza, and place the order.
   - **Store orders**, which displays all the orders placed so far. The GUI shall list the detail of each order and display the total amount of each order, with 2 decimal places. The user can select an order and cancel the order. The GUI shall display the remaining orders after the cancellation. The user can also export the store orders to a text file, which shall include the details of every order, consistent with the details displayed on the GUI.
6. For each GUI, you must create a .fxml file and its associated controller class. That is, you will have 4 .fxml files and 4 controller.java files. **-5 points** if this is not done properly.
7. When the user is navigating between the GUIs, the primary stage (main menu) must remain visible, **or -5 points.**
8. You can use any JavaFX UI controls or containers to design your GUIs. However, you MUST use ComboBox, ImageView and ListView, or **-3 points** for each violation.
9. **You are NOT ALLOWED to use System.out** in ALL CLASSES, **or you will lose 10 points**. Use either a TextArea or Alert pop-ups to display messages.
10. You can use any Java library classes. However, there are restrictions listed below.
    - All the instance variables defined in the controller classes must be "private", **-2 points** for each violation.
    - Must include the **abstract Pizza class** below. You can add "static final" data items ONLY or **lose 2 points**. You can add additional methods if necessary. All flavors of pizzas should be a subtype of Pizza class; for example, `public class Deluxe extends Pizza`. You should have 3 subclasses extending Pizza class, including Deluxe, Hawaiian and Pepperoni. **-5 points** for each subclass not implemented or not used.

```java
public abstract class Pizza {
    protected ArrayList<Topping> toppings = new ArrayList<Topping>();
    protected Size size;
    public abstract double price();
}
```

- Must include the PizzaMaker class below to create an instance of Pizza. This class should be used in the controller classes to create an instance of Pizza class based on a chosen flavor. That is, this is the ONLY class that can "new" an instance of the subclasses. This is to hide the details of creating the pizza objects and allow flexibility to add new pizza flavors in the future, similar to a **design pattern** called **"Factory Method".**

```
//create an instance of subclasses based on the chosen flavor
public class PizzaMaker {
    public static Pizza createPizza(String flavor) {
       //write the code for creating different instances of subtypes of pizza
    }
}
```

  You CANNOT add any additional instance variables, constants, or other methods to this class, or you will **lose 3 points.** You CANNOT use the subclasses, such as Deluxe, Hawaiian and Pepperoni classes, in ALL other Java classes except the PizzaMaker class. **-5 points** for each violation. You should only use the Pizza class as the data type in all other Java classes.

- Must include an **Order class**. An instance of this class has a unique phone number and keeps the list of instances of Pizza class. **-5 points** if the class is not implemented or not used. All instance variables must be private or **lose 2 points**.
- Must include an **StoreOrders class**. An instance of this class keeps the list of orders placed by the user. **-5 points** if the class is not implemented or not used. All instance variables must be private or **lose 2 points**. You should include an **export() method** in this class to save the store orders to an external text file, **-2 points** if this is not done in this class.

11. Create a JUnit test class for one of your subclass of the Pizza class, such as Deluxe class, to **unit testing the price() method**. Use the Black-Box testing technique to design the test cases and ensure the price() method performs properly with various numbers of toppings and different sizes. The JUnit test class and test cases are **worth 10 points**. You MUST comment your test cases, but DO NOT need to generate Javadoc for the JUnit test class.
12. You are required to **generate the Javadoc** after you properly commented your code. Your Javadoc must include the documentations for the constructors, private methods and public methods of all Java classes (*.java files.) You **must comment ALL Java classes, including Main.java and ALL controller classes.** They must be included in the Javadoc. DO NOT include the *.fxml files, which are NOT java files. DO NOT include the JUnit test class. Generate the Javadoc in a single folder and include it in your project folder to be submitted to Canvas. You are responsible to **double check** your Javadoc after you generated them. The grader will navigate the Javadoc with the "index.html". You will **lose 5 points** for not including the Javadoc, OR, the grader cannot navigate your Javadoc through the "index.html".
13. Create a Class diagram for this project to document your design. You must include all Java classes in this project. DO NOT include the library classes from Java or JavaFX. DO NOT include the JUnit test class and .fxml files. The class diagram **is worth 10 points**.

System Testing

1. You are responsible to thoroughly test your software and ensure your software is meeting the requirements listed above. You will **lose 2 points** for each incorrect implementation, incorrect amount or incorrect information shown on the GUIs.
2. Your software must always run in a sane state and **should not crash in any circumstances**. You must catch all Java Exceptions. Your program shall continue to run until the user stops the program execution or closes the window. **You will lose 2 points** for each exception not caught.