Due Date

Monday, April 13, by 11:59pm

Submission

1. You must designate a submitter (one of the team members) to submit all required files to Canvas. The comment block on top of each Java file must include the names of all team members.
2. Submit the following files, including
   - *.java files, *.fxml files, the JavaFX package folder (*.jar), and all the image files used in this program [25 points]
3. JUnit test class [5 points]
4. Personal Time logs. This is an individual assignment. [5 points]

Program Description

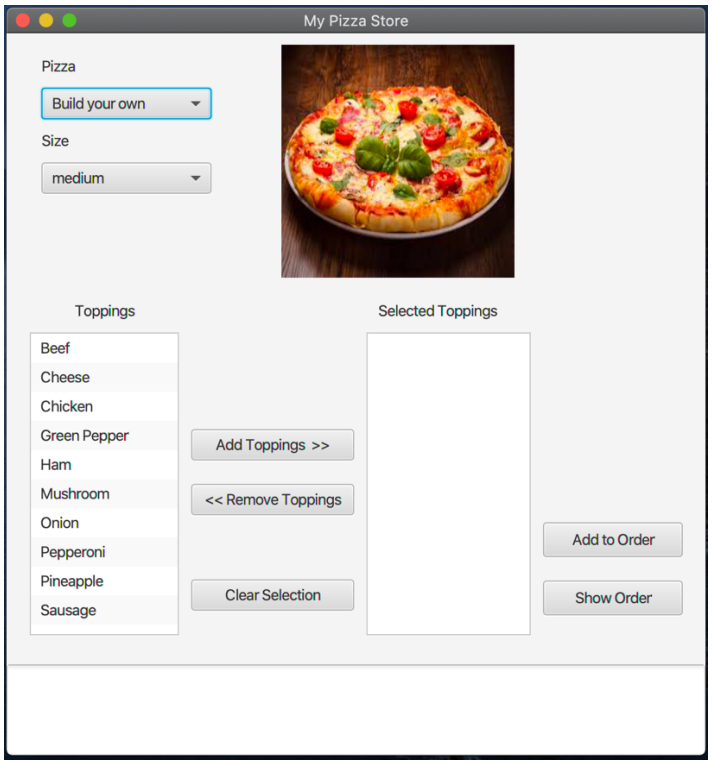
You will be using JavaFX to develop a software system for ordering pizzas. For simplicity, let's abstract away the details of the customer who is ordering the pizza, and abstract away the checkout functionality. However, your software must meet the following functional requirements. **-2 points** for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

1) The system shall provide the options of 2 specialty pizzas, Deluxe and Hawaiian, which have the customized toppings, and Build your own pizza, which allows the customer to customize the toppings.
2) The default pizza selected on the GUI shall be set to Build your own.
3) Upon the selection of the pizza, an image of the pizza shall be displayed on the GUI.
4) For the Deluxe and Hawaiian pizzas, the customer shall not be allowed to customize the toppings.
5) For the Build your own pizza, the system shall provide a list of at least 10 toppings for the customer to choose from. The customer can add or remove the toppings.
6) The customer can choose up to 6 toppings for the Build your own pizza, but at least one topping must be chosen before the pizza is added to the order.
7) When the customer is adding the toppings, the system shall not allow the same toppings to be added more than once.
8) The system shall provide the option of selecting the size of the pizza in either small(10"), medium(12") or large(14").
9) The default pizza size on the GUI shall be set to medium.
10) The customer can order multiple pizzas in an order, by adding one pizza at a time.
11) The system shall be able to show the order details with a list of pizzas added by the customer. For each pizza in the order, the system shall print out the pizza style, size, the list of toppings and the price. At the end of the list, a total amount of the order shall be displayed.
12) The customer can clear the current selection of the pizza, and clear the entire order; that is, remove all the pizzas added on the order.

Program Requirement

1. This is a **group assignment**. You MUST work in pair in order to get the credit for this program. You MUST submit a runnable program to pass this course.
2. You MUST follow the software development ground rules.
3. You are required to log your times working on this project with the template provided. **You will lose up to 5 points** if the log is not submitted or incomplete. The time log is an individual assignment.
4. Each Java class must go in a separate file. **-2 points** if you put more than one Java class into a file.

5. You can use any JavaFX components. However, you MUST include the following JavaFX components, or **-3 points** for each violation. (a) **ComboBox** for the pizza selection; (b) **ImageView** for the pizza images; (c) **TextArea** for displaying the system messages. **NO System.out!!** (d) **ListView** – one for topping options and one for selected toppings.
6. **You can use any container classes in the Java Collections** to hold the pizza order (a list of pizzas).
7. You MUST use a 2nd Stage (window) to show the order details, which is a list of pizzas added to the order. You will lose **10 points** if you don't implement a 2nd stage for the order details. The details of each pizza should include the name of the pizza style, pizza size, the list of toppings and the pizza price. At the end of the pizza list, you must display a running total of the order. **-1 point** for each item of the pizza not properly displayed.
8. You MUST create a second .fxml and a second controller for the 2nd stage, or you will **lose 5 points.**
9. You MUST include a "**Clear**" button on the 2nd stage (the order detail window) to empty the order, and a "**Back**" button to close the 2nd Stage. **-2 points** for each button missing.
10. A sample GUI design for the primary stage is shown below. I encourage you to come up with your own GUI design for the primary Stage, as well as the 2nd stage.



11. You MUST use the abstract class Pizza below, **-10 points** if you don't. You CANNOT add additional data members except the necessary constants. You MUST use the Java container class **ArrayList** to hold the list of toppings. **-3 points** for each violation. You can change the data types of the data members. You can add additional constructors and public methods.

```
public abstract class Pizza {
	protected String style;
	protected String size;
	protected ArrayList<String> toppings;

	public Pizza(String style, String size, ArrayList<String> toppings) { }
	public Pizza(String style, String size) { }

	public abstract int pizzaPrice();
	public String toString() {  }
}
```

12. You must create at least 3 subclasses that extend the Pizza class: **Deluxe**, **Hawaiian** and **BuildYourOwn**. **-5 points** for each subclass missing. Each subclass must implement the **pizzaPrice()** abstract method to return the price for the pizzas, and override the **toString()** method. **-3 points** for each method missing. You can determine the data members, constructors and public methods in the subclasses. The price of a pizza is determined as follows.

Pizza Style	Toppings	Small	Medium	Large
Deluxe	Sausage, Pepperoni, Green Pepper Onion, Mushroom	$14	Small + $2	Small + $4
Hawaiian	Ham, Pineapple	$8	Small + $2	Small + $4
Build your own	Up to 6 toppings, at least 1 topping	$5 + $2/topping	Small + $2 + $2/topping	Small + $4 + $2/topping

13. Create a **JUnit test class** for the **BuildYourOwn class** and test the **pizzaPrice() method**. Use the Black-Box testing technique to design the test cases. You must show all test cases are passed. This part is worth 5 points.
14. **Program Testing.** Your program must meet the requirements and always run in a sane state. You are responsible to thoroughly test your GUI. You will **lose 2 points** for any exception not caught or any malfunction on the GUI, with a **maximum of 10 points off**.