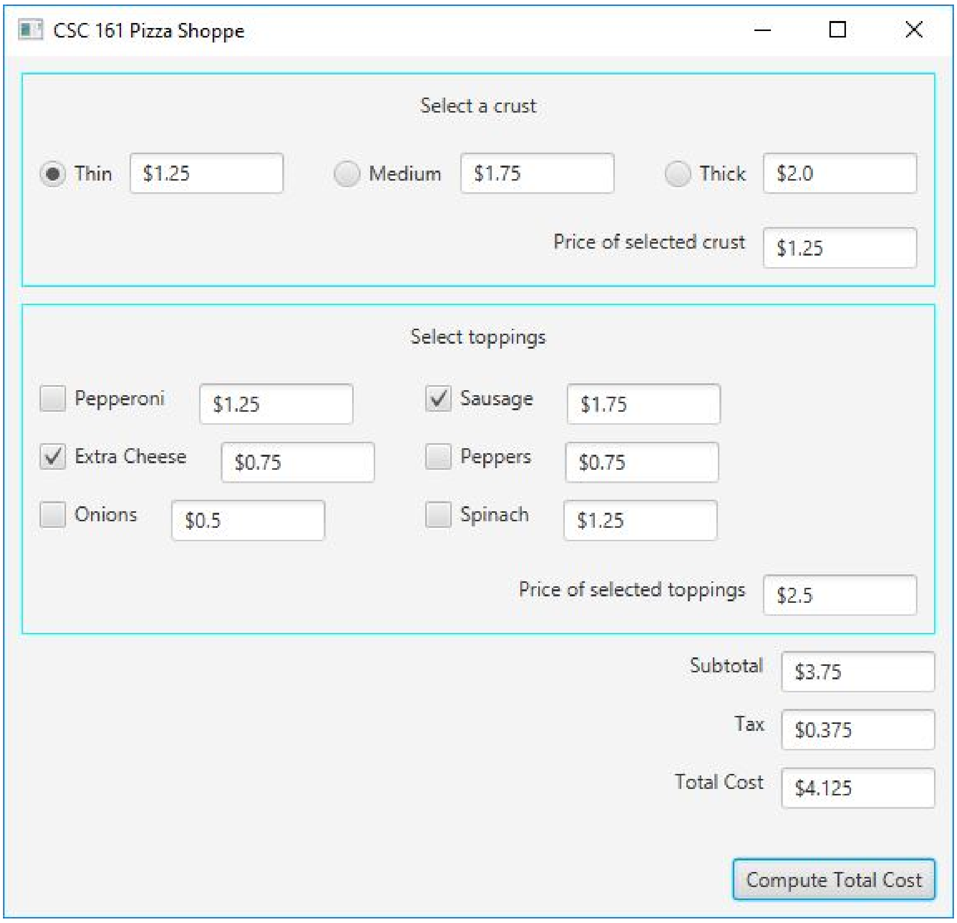
CSC 163 LAB 3-2
PIZZA ORDER GROUP PROJECT

PROFESSOR GODFREY MUGANDA

1. Overview

Write a JavaFX application that implements a program for ordering a pizza. The user interface should look as seen in the following graphic.



CSC 161 Pizza Shoppe

Select a crust

● Thin $1.25   ○ Medium $1.75   ○ Thick $2.0

Price of selected crust $1.25

Select toppings

☐ Pepperoni $1.25   ☑ Sausage $1.75

☑ Extra Cheese $0.75   ☐ Peppers $0.75

☐ Onions $0.5   ☐ Spinach $1.25

Price of selected toppings $2.5

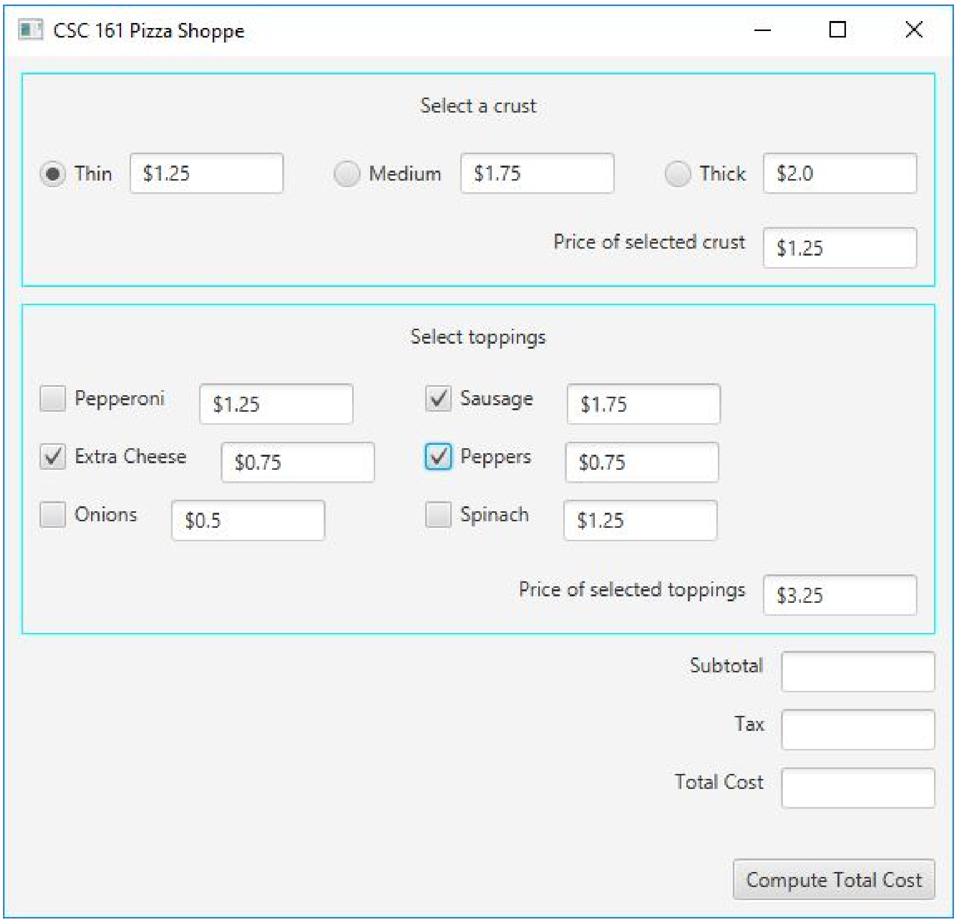
Subtotal $3.75

Tax $0.375

Total Cost $4.125

The graphic shown is right after a user has made some selections of a crust and toppings by using the radio buttons and checkboxes.

Whenever the user clicks on a crust radio button or a toppings check box, the price of the selected crust and / or toppings will be shown in the appropriate text fields, and the three text fields at the botton will be cleared. The text fields at the bottom are only set when the user clicks on the *Compute Total Cost* button.



CSC 161 Pizza Shoppe

Select a crust

● Thin $1.25 ○ Medium $1.75 ○ Thick $2.0

Price of selected crust $1.25

Select toppings

□ Pepperoni $1.25 ☑ Sausage $1.75

☑ Extra Cheese $0.75 ☑ Peppers $0.75

□ Onions $0.5 □ Spinach $1.25

Price of selected toppings $3.25

Subtotal

Tax

Total Cost

The user interface has a panel at the top for selecting a crust. This panel uses radio buttons. There is another panel for selecting toppings that uses check boxes. Both of these panels have a un-editable text field that shows the total cost of the selections in that panel. These text fields will be immediately updated with each change in the selections.

The bottom of the user interface has places for the subtotal, tax (tax is 10 %) of total cost. These three fields are updated when the Calculate Total Cost button at the bottom is clicked.

You should proceed by carefully examining the user interface to see where you can reuse code. For example, note how a label and an un-editable text field appear together 5 times: for the Price of the selected crust, the Price of the selected toppings, the Subtotal, the Tax, and Total Cost. Each time they appear, they are inside of a horizontal box with center-right alignment. To avoid code duplication, you can create a subclass of a horizontal box that takes a string and a text field as constructor parameters and sets up the horizontal box.

You must use a `GridPane` for the toppings check boxes.

2. Working in Groups

This lab is more complex than the others, so you may work in groups of 1, 2, or 3, as you wish. I do recommend working in a group, as it is a good experience.

Students that elect to work in a group must appoint a group leader, and the group leader will submit a list of the group members to me before the end of the lab. The group member will be responsible for convening the group to work on the project outside of lab hours.

Due to the complexity of this lab, it will be worth twice the points of an ordinary lab. It is due Wednesday of Week 4 at midnight. The group leader will be responsible to submit by email, with a CC to all members of the group.

3. Additional Requirements

Be sure to ask for help and hints from the professor and preceptor as you work. Use the following variables (do not change the variable names) to get started.

```
public void start(Stage stage) throws Exception
    {
        // Active user interface components
        final int TF_COLUMN_COUNT = 7;
        // Crusts
        String[] crustNames =
        {
            "Thin", "Medium", "Thick"
        };
        double[] crustPrices =
        {
            1.25, 1.75, 2.0
        };

        ToggleGroup radiosToggleGroup = new ToggleGroup();
        CrustRadioButton[] crustRadioButtons =
             new CrustRadioButton[crustNames.length];
        TextField[] crustPricesTFs = new TextField[crustNames.length];

        for (int k = 0; k < crustRadioButtons.length; k++)
        {
            crustRadioButtons[k] =
                new CrustRadioButton(crustNames[k], crustPrices[k]);
            crustRadioButtons[k].setToggleGroup(radiosToggleGroup);
            crustPricesTFs[k] = new TextField();
            crustPricesTFs[k].setText("$" + crustPrices[k]);
            crustPricesTFs[k].setPrefColumnCount(TF_COLUMN_COUNT);
            crustPricesTFs[k].setEditable(false);
        }
        TextField selectedCrustPriceTF = new TextField();
        selectedCrustPriceTF.setPrefColumnCount(TF_COLUMN_COUNT);
        selectedCrustPriceTF.setEditable(false);

        // Toppings
        String[] toppingsNames =
        {
            "Pepperoni", "Sausage", "Extra Cheese",
            "Peppers", "Onions", "Spinach"
        };
        double[] toppingsPrices =
        {
            1.25, 1.75, 0.75, 0.75, 0.5, 1.25
        };

        ToppingsCheckBox[] toppingsCheckBoxes =
             new ToppingsCheckBox[toppingsNames.length];
        TextField[] toppingsPricesTFs =
             new TextField[toppingsNames.length];
        TextField selectedToppingsPriceTF = new TextField();

        // The three "Cost" text fields at the bottom of the UI
        TextField subTotalTF = new TextField();
        TextField taxTF = new TextField();
        TextField totalCostTF = new TextField();

        // Button at bottom of the UI
        Button totalCostButton = new Button("Compute Total Cost");
```

To simplify things, we have defined a subclass of `RadioButton` that is used to select a crust. This subclass keeps track of the price of a crust. This class should be placed in a separate file:

```
package csc161lab3_2;

import javafx.scene.control.RadioButton;
public class CrustRadioButton extends RadioButton
{
    final double price;
    public CrustRadioButton(String label, double price)
    {
        super(label);
        this.price = price;
    }
    public double getPrice() { return price; }
}
```

You must define a similar class for a toppings check box.

In addition, you must define a subclass of `VBox` for the top pane (crusts) and another subclass for the toppings pane. These will be similar to what you did in Lab 3-1: Ask the instructor and preceptor for help if you need it.

You may find to define other classes as well.

Your user interface must look nice, and must be very close to what is shown here.

Have fun!