

# TITLE PAGE

**Course:** CS1073

**Section:** FR03B

**Assignment number:** 5

**Name:** Zohaib Hassan Khan

**UNB student number:** 3740572

## a) Graphics.java:

```
/**
 * This class represents a gpa calculator.
 * @author Zohaib Khan 3740572.
 */
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
import javafx.scene.control.Button;
import javafx.scene.text.Text;
import javafx.scene.layout.FlowPane;
import javafx.geometry.Pos;
import javafx.event.ActionEvent;
import java.text.NumberFormat;

public class Graphics extends Application {
    private TextField letterField;
    private TextField letterField2;
    private Text pointsResult;
    private Text gpaResult;
    private double totalPoints = 0;
    private int totalCreditHours = 0;
    private double points = 0;
    private double gpa = 0;

    public void start (Stage primaryStage) {
        primaryStage.setTitle ("My GPA Calculator");
        Label fieldLabel = new Label ("Course letter grade:");
        Label fieldLabel2 = new Label ("Course credit hours:");

        letterField = new TextField ();
        letterField2 = new TextField ();

        letterField.setPrefWidth (50);
        letterField2.setPrefWidth (50);

        letterField.setOnAction(this::addRequest);
        letterField2.setOnAction(this::addRequest);

        Button addButton = new Button ("Add to GPA");
        Button clearButton = new Button ("Clear GPA");

        addButton.setOnAction (this::addRequest);
        clearButton.setOnAction (this::clearRequest);

        pointsResult = new Text ("Welcome to my GPA calculator!");
    }
}
```

```

gpaResult = new Text ("Enter your 1st grade & credit hrs.");

FlowPane pane = new FlowPane (fieldLabel, letterField,
                                fieldLabel2, letterField2,
                                addButton, clearButton,
                                pointsResult, gpaResult);

pane.setAlignment(Pos.CENTER);
// pane.setStyle("-fx-background-color:palevioletred");
pane.setHgap (10);
pane.setVgap (20);

Scene scene = new Scene (pane, 250, 300);
primaryStage.setScene (scene);
primaryStage.show ();

}

public void addRequest (ActionEvent event) {
    String grade = letterField.getText();
    int creditHours = Integer.parseInt(letterField2.getText());

    NumberFormat formatter = NumberFormat.getNumberInstance();
    formatter.setMaximumFractionDigits(1);
    formatter.setMinimumFractionDigits(1);

    switch (grade) {
        case "A+": points = Double.parseDouble(formatter.format
                                                    (4.3*creditHours));
                    pointsResult.setText ("Points for this course: " +
                                            formatter.format(points));
                    totalPoints += points;
                    totalCreditHours += creditHours;
                    break;

        case "A": points = Double.parseDouble(formatter.format
                                                    (4.0*creditHours));
                    totalPoints += points;
                    pointsResult.setText ("Points for this course: " +
                                            formatter.format(points));
                    totalCreditHours += creditHours;
                    break;

        case "A-": points = Double.parseDouble(formatter.format
                                                    (3.7*creditHours));
                    totalPoints += points;
                    pointsResult.setText ("Points for this course: " +
                                            formatter.format(points));
                    totalCreditHours += creditHours;
                    break;
    }
}

```

```

case "B+": points = Double.parseDouble(formatter.format
                                     (3.3*creditHours));
    totalPoints += points;
    pointsResult.setText ("Points for this course: " +
                          formatter.format(points));
    totalCreditHours += creditHours;
    break;

case "B": points = Double.parseDouble(formatter.format
                                     (3.0*creditHours));
    totalPoints += points;
    pointsResult.setText ("Points for this course: " +
                          formatter.format(points));
    totalCreditHours += creditHours;
    break;

case "B-": points = Double.parseDouble(formatter.format
                                     (2.7*creditHours));
    totalPoints += points;
    pointsResult.setText ("Points for this course: " +
                          formatter.format(points));
    totalCreditHours += creditHours;
    break;

case "C+": points = Double.parseDouble(formatter.format
                                     (2.3*creditHours));
    totalPoints += points;
    pointsResult.setText ("Points for this course: " +
                          formatter.format(points));
    totalCreditHours += creditHours;
    break;

case "C": points = Double.parseDouble(formatter.format
                                     (2.0*creditHours));
    totalPoints += points;
    pointsResult.setText ("Points for this course: " +
                          formatter.format(points));
    totalCreditHours += creditHours;
    break;

case "D": points = Double.parseDouble(formatter.format
                                     (1.0*creditHours));
    totalPoints += points;
    pointsResult.setText ("Points for this course: " +
                          formatter.format(points));
    totalCreditHours += creditHours;
    break;

```

```

        case "F": points = Double.parseDouble(formatter.format
                                                    (0.0*creditHours));
            totalPoints += points;
            pointsResult.setText ("Points for this course: " +
                                formatter.format(points));
            totalCreditHours += creditHours;
            break;

        case "WF": points = Double.parseDouble(formatter.format
                                                    (0.0*creditHours));
            totalPoints += points;
            pointsResult.setText ("Points for this course: " +
                                formatter.format(points));

            totalCreditHours += creditHours;
            break;

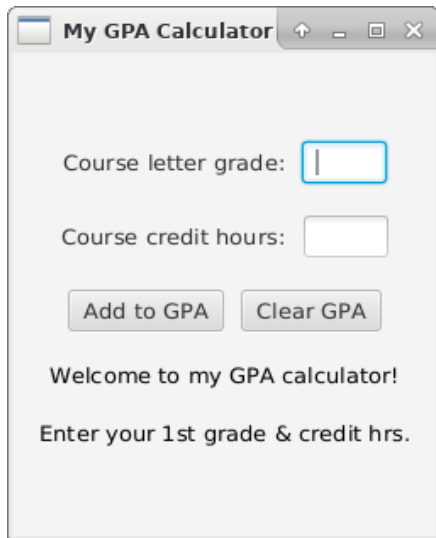
        default: pointsResult.setText ("Invalid grade - GPA not
                                        changed.");
    }

    gpaResult.setText("Your Cumulative GPA is: " +
                    formatter.format(totalPoints/totalCreditHours));
}

public void clearRequest (ActionEvent event) {
    pointsResult.setText ("Total points are reset");
    letterField.setText("");
    letterField2.setText("");
    gpaResult.setText("Enter your 1st grade & credit hrs.");
    totalPoints = 0;
    totalCreditHours = 0;
}
}

```

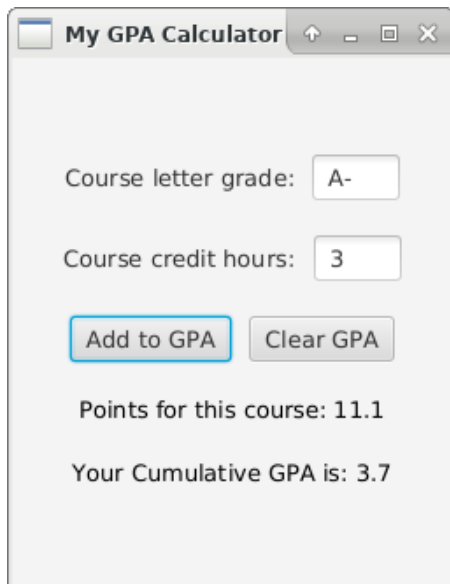
B)



A screenshot of a web browser window titled "My GPA Calculator". The window has a light gray background and a standard browser address bar. Inside the window, there are two input fields: "Course letter grade:" with a text box containing a single character "I", and "Course credit hours:" with a text box containing the number "3". Below these fields are two buttons: "Add to GPA" and "Clear GPA". At the bottom of the window, there is a welcome message: "Welcome to my GPA calculator!" and a prompt: "Enter your 1st grade & credit hrs."

**As5\_Q1\_Output1.png:**

GPA calculator before  
any user input.



A screenshot of the same "My GPA Calculator" web browser window. The "Course letter grade:" field now contains "A-" and the "Course credit hours:" field contains "3". The "Add to GPA" button is highlighted with a blue border. Below the input fields, the text "Points for this course: 11.1" is displayed. At the bottom, the text "Your Cumulative GPA is: 3.7" is displayed.

**As5\_Q1\_Output2.png:**

GPA calculator output  
after adding all of John  
Doe's information.

A screenshot of a Java Swing window titled "My GPA Calculator". It contains two text input fields: "Course letter grade:" with the value "C-" and "Course credit hours:" with the value "3". Below these are two buttons: "Add to GPA" (highlighted with a blue border) and "Clear GPA". The output area shows the message "Invalid grade - GPA not changed." and "Your Cumulative GPA is: 3.7".

**As5\_Q1\_Output3.png:**

GPA calculator output after invalid grade entry.

A screenshot of the "My GPA Calculator" window after the "Clear GPA" button has been pressed. The input fields for "Course letter grade:" and "Course credit hours:" are now empty. The buttons "Add to GPA" and "Clear GPA" (highlighted with a blue border) are still present. The output area displays "Total points are reset" and "Enter your 1st grade & credit hrs.".

**As5\_Q1\_Output4.png:**

GPA calculator output after pressing "Clear GPA" button.

### c) ResortBooking.java:

```
/**
 * This abstract class represents a resort booking.
 * @author Zohaib Khan 3740572.
 */
public abstract class ResortBooking {

    /**
     * The guest's name.
     */
    private String guestName;

    /**
     * The number of meals the guest plans on taking in à la Carte.
     */
    private int numMeals;

    /**
     * The number of spas the guest plans on taking.
     */
    private int numSpas;

    /**
     * This constructor method initialises the instance variables.
     * @param name the name of the guest.
     * @param numMeals the number of meals the guest will eat at à la Carte.
     * @param numSpas the number of spas the guest plans on taking.
     */
    public ResortBooking (String guestName, int numMeals, int numSpas){
        this.guestName = guestName;
        this.numMeals = numMeals;
        this.numSpas = numSpas;
    }

    /**
     * This method gets the number of meals the guest plans on taking in
     * à la Carte.
     * @return the number of meals the guest on taking in à la Carte.
     */
    public int getMeals (){
        return numMeals;
    }
}
```



```
/**
    The method gets the number of spas the guest plans on taking.
    @return the number of spas the guest plans on taking.
*/
public int getSpas (){
    return numSpas;
}

/**
    This abstract method returns the total amount the guest has to
    pay.
*/
public abstract double getCost ();
}
```

## TouristPackageBooking.java:

```
/**
 * This sub-class represents a tourist package booking.
 * @author Zohaib Khan 3740572.
 */

public class TouristPackageBooking extends ResortBooking {

    /**
     * This constructor method initialises the instance variables.
     * @param name the name of the guest.
     * @param numMeals the number of meals the guest will eat at à la Carte.
     * @param numSpas the number of spas the guest plans on taking.
     */
    public TouristPackageBooking (String name, int numMeals, int
                                   numSpas) {
        super (name, numMeals, numSpas);
    }

    /**
     * This method calculates the total price that the guest has to pay.
     * @return the total price that the guest has to pay.
     */
    public double getCost () {

        if (super.getSpas() == 1) {
            return 1475 + (35 * super.getMeals()) + (125);
        }
        else if (super.getSpas() > 1) {
            return 1475 + (35 * super.getMeals()) + (125) +
                (100 * (super.getSpas() - 1));
        }
        else {
            return 1475 + (35 * super.getMeals());
        }
    }

    /**
     * This method returns the building number that the guest is
     * assigned.
     * @return the building number that the guest is assigned.
     */
    public int getBuilding () {
        return 2 + (int) (Math.random() * 4);
    }
}
```

## ElitePackageBooking.java:

```
/**
 * This sub-class represents the elite package booking.
 * @author Zohaib Khan 3740572.
 */

public class ElitePackageBooking extends ResortBooking {

    /**
     * This constructor method initialises the instance variables.
     * @param name the name of the guest.
     * @param numMeals the number of meals the guest will eat at à la Carte.
     * @param numSpas the number of spas the guest plans on taking.
     */

    public ElitePackageBooking (String name, int numMeals, int
                                numSpas){
        super (name, numMeals, numSpas);
    }

    /**
     * This method calculates the total price that the guest has to pay.
     * @return the total price that the guest has to pay.
     */
    public double getCost (){
        if(super.getMeals() > 3){
            return 2250 + (75* super.getSpas()) + ((super.getMeals() - 3)
                * 35);
        }
        else{
            return 2250 + (75 * super.getSpas());
        }
    }

    /**
     * This method returns the building number that the guest is
     * assigned.
     * @return the building number that the guest is assigned.
     */
    public int getBuilding (){
        return 1;
    }
}
```

#### d) Booking.java:

```
/**
 * This class represents a resort booking application.
 * @author Zohaib Khan 3740572.
 */
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
import javafx.scene.control.Button;
import javafx.scene.text.Text;
import javafx.scene.layout.FlowPane;
import javafx.geometry.Pos;
import javafx.event.ActionEvent;
import java.text.NumberFormat;

public class Booking extends Application {
    private TextField letterField;
    private TextField letterField2;
    private TextField letterField3;

    private Text buildingResult;
    private Text priceResult;

    public void start (Stage primaryStage) {

        primaryStage.setTitle ("Package Calculator");

        Label fieldLabel = new Label ("Guest Name:");
        Label fieldLabel2 = new Label ("Number of à la Carte Meals:");
        Label fieldLabel3 = new Label ("Number of Spa Visits:");

        letterField = new TextField ();
        letterField2 = new TextField ();
        letterField3 = new TextField ();

        letterField.setPrefWidth (130);
        letterField2.setPrefWidth (50);
        letterField3.setPrefWidth (50);

        letterField.setOnAction(this::touristRequest);
        letterField2.setOnAction(this::touristRequest);
        letterField3.setOnAction(this::touristRequest);

        letterField.setOnAction(this::eliteRequest);
        letterField2.setOnAction(this::eliteRequest);
        letterField3.setOnAction(this::eliteRequest);
    }
}
```

```

Button touristButton = new Button ("Tourist");
Button eliteButton = new Button ("Elite");
Button resetButton = new Button ("Reset");

touristButton.setOnAction (this::touristRequest);
eliteButton.setOnAction (this::eliteRequest);
resetButton.setOnAction (this::resetRequest);

buildingResult =
new Text ("Welcome to Paradise Palms!");

priceResult =
new Text ("Enter your booking information");

//start method continued on the next slide...

FlowPane pane = new FlowPane
(fieldLabel, letterField, fieldLabel2, letterField2,
 fieldLabel3, letterField3, touristButton, eliteButton,
 resetButton, buildingResult, priceResult);

pane.setAlignment(Pos.CENTER);

pane.setHgap (10);
pane.setVgap (20);
Scene scene = new Scene (pane, 270, 300);
primaryStage.setScene (scene);
primaryStage.show ();

}

public void touristRequest (ActionEvent event) {
    String name = letterField.getText();
    int meals = Integer.parseInt(letterField2.getText());
    int spas = Integer.parseInt(letterField3.getText());

    NumberFormat formatter = NumberFormat.getCurrencyInstance();

    TouristPackageBooking tourist =
    new TouristPackageBooking (name, meals, spas);

    buildingResult.setText("Building Number: " +
                           tourist.getBuilding());

    priceResult.setText("Total price for this package: " +
                        formatter.format(tourist.getCost()));
}

public void eliteRequest (ActionEvent event) {
    String name = letterField.getText();

```

```
int meals = Integer.parseInt(letterField2.getText());
int spas = Integer.parseInt(letterField3.getText());

NumberFormat formatter = NumberFormat.getCurrencyInstance();

ElitePackageBooking elite =
new ElitePackageBooking (name, meals, spas);

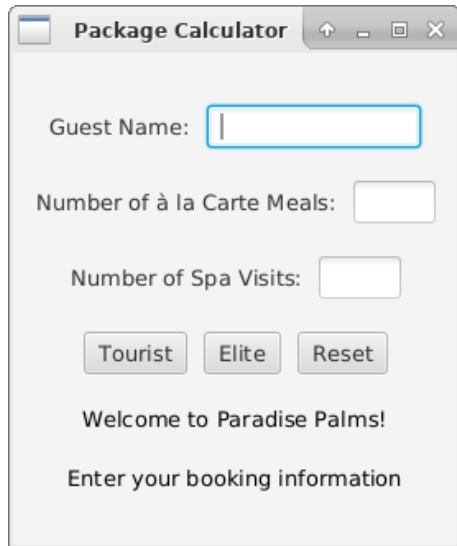
buildingResult.setText("Building Number: " +
                        elite.getBuilding());

priceResult.setText("Total price for this package: " +
                    formatter.format(elite.getCost()));

}

public void resetRequest (ActionEvent event) {
    buildingResult.setText ("Welcome to Paradise Palms!");
    letterField.setText("");
    letterField2.setText("");
    letterField3.setText("");
    priceResult.setText("Enter your booking information");
}
} //end class
```

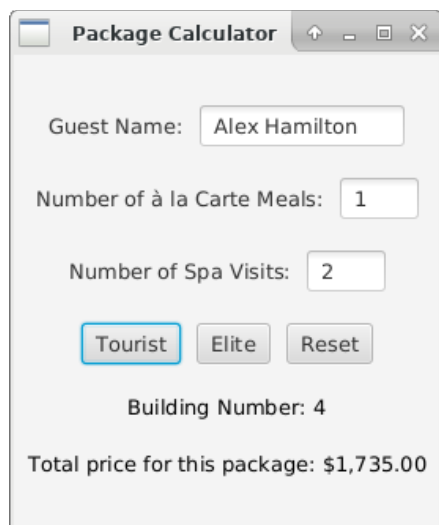
E)



The screenshot shows a window titled "Package Calculator" with standard window controls. It contains three input fields: "Guest Name:" (empty), "Number of à la Carte Meals:" (empty), and "Number of Spa Visits:" (empty). Below these are three buttons: "Tourist", "Elite", and "Reset". At the bottom, there is a welcome message "Welcome to Paradise Palms!" and a prompt "Enter your booking information".

**As5\_Q2\_Output1.png:**

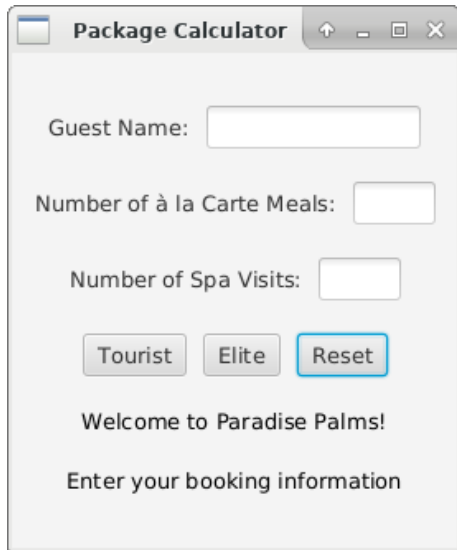
Package Calculator  
before any user input.



The screenshot shows the same "Package Calculator" window after user input. The "Guest Name:" field now contains "Alex Hamilton". The "Number of à la Carte Meals:" field contains "1". The "Number of Spa Visits:" field contains "2". The "Tourist" button is now highlighted with a blue border. Below the buttons, the text "Building Number: 4" is displayed. At the bottom, the text "Total price for this package: \$1,735.00" is shown.

**As5\_Q2\_Output2.png:**

Package Calculator  
output using tourist  
package.



Package Calculator

Guest Name:

Number of à la Carte Meals:

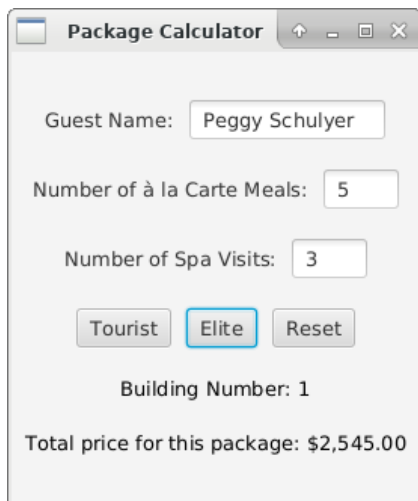
Number of Spa Visits:

Welcome to Paradise Palms!

Enter your booking information

**As5\_Q2\_Output3.png:**

Package Calculator  
output after pressing  
the “Reset” button.



Package Calculator

Guest Name:

Number of à la Carte Meals:

Number of Spa Visits:

Building Number: 1

Total price for this package: \$2,545.00

**As5\_Q2\_Output4.png:**

Package Calculator  
output using the elite  
package.