CIS355A Lab Report

**Your Name: \_\_\_\_\_\_Sarah Bohn\_\_\_\_\_\_**

**Date: \_\_8/20/18\_\_\_\_\_\_\_**

**Lab Week \_\_\_\_\_\_Practice 6\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Testing/Results**

Does your program satisfy all requirements of the lab?  **\_No\_\_\_\_**

|  |
| --- |
| **Screenshots go here:** |

**Objective/Purpose of the program**

Briefly describe the program’s requirements:

**\_\_Add database connectivity to the pizza program\_\_\_**

**Analysis/Design**

Describe the approach/structure of program. What classes/functions were used?

**\_\_The program has many classes and a GUI with arrays, buttons, check boxes, text boxes and subtotals, then we added the database connectivity with MySQL\_\_\_**

**How did you TEST your Lab**? Indicate test cases used, expected values, and *show results with screen shots.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Case** | **Description** | **Test Data** | **Expected Result** | **Actual Result** |
| 1 | The connectivity to the database | None, kept timing out | It to load properly | Timed out many times. |

**Conclusions / Lessons Learned**

What difficult problems did you encounter, and how did you handle them?

This program is very big and its difficult to sometimes add code to a big program without messing up another part, it’s something I really need to work on.

What new concepts did you learn/reinforce with this lab?

Refresher on how to make a database and learning how to use Java to add, update, and delete, data.

Is there anything you would have done differently?

Not broken the program.

**Paste your program here:** ***(Copy and Paste all of your code here. Include lots of comments in your code as a demonstration of understanding.***

*/\* Week 5 File IO \*/*

*//open package for class*

*package business;*

*//import list for class*

*import java.util.ArrayList;*

*import javax.swing.DefaultListModel;*

*import data.\*;*

*import java.io.IOException;*

*import java.sql.SQLException;*

*import java.text.NumberFormat;*

*import java.util.Locale;*

*//creating array for data*

*public class OrderList {*

*private ArrayList<PizzaOrder> orderList;*

*private static final NumberFormat CF = NumberFormat.getCurrencyInstance(Locale.US);*

*private final DataIO dataIO;*

*//getting pizza order info*

*public OrderList() {*

*orderList = new ArrayList<>();*

*dataIO = new DataIO();*

*}*

*private void getList() {*

*orderList.clear();*

*orderList = dataIO.getList();*

*}*

*public boolean add(PizzaOrder aOrder) {*

*boolean success = false;*

*if (aOrder != null) {*

*orderList.add(aOrder);*

*success = dataIO.add(aOrder);*

*}*

*return success;*

*}*

*public boolean remove(PizzaOrder aOrder) throws ClassNotFoundException {*

*boolean success = false;*

*if (aOrder != null) {*

*orderList.remove(aOrder);*

*success = dataIO.delete(aOrder);*

*}*

*return success;*

*}*

*public boolean update(PizzaOrder aOrder) {*

*boolean success = false;*

*if (aOrder != null) {*

*success = dataIO.update(aOrder);*

*}*

*return success;*

*}*

*public int setList(DefaultListModel<PizzaOrder> listModel) {*

*int count = 0;*

*getList();*

*if (!orderList.isEmpty()) {*

*listModel.clear();*

*count = orderList.size();*

*for(PizzaOrder order : orderList) {*

*listModel.addElement(order);*

*}*

*}*

*return count;*

*}*

*public String getOrderSummary() {*

*double totalOrder = 0;*

*double averagePrice = 0;*

*int count = 0;*

*if (!orderList.isEmpty()) {*

*for(PizzaOrder aOrder:orderList) {*

*totalOrder += aOrder.getTotal();*

*count++;*

*}*

*if (count > 0) {*

*averagePrice = totalOrder/count;*

*}*

*}*

*StringBuilder str = new StringBuilder();*

*str.append("Pizza Order Summary:\n");*

*str.append("Number of total orders: ");*

*str.append(count);*

*str.append("\n");*

*str.append("Total Price of Orders: ");*

*str.append(CF.format(totalOrder));*

*str.append("\n");*

*str.append("Average Price of Orders: ");*

*str.append(CF.format(averagePrice));*

*return str.toString();*

*}*

*}*

*//open package for class*

*package business;*

*//start of class for client*

*public class Person {*

*private static final String DEFAULT\_NAME = "Not Given";*

*//variables for client name*

*String firstName;*

*String lastName;*

*//constructors for name*

*public Person()*

*{*

*this.firstName = DEFAULT\_NAME;*

*this.lastName = DEFAULT\_NAME;*

*}*

*public Person(String firstName, String lastName) {*

*setFirstName(firstName);*

*setLastName(lastName);*

*}*

*public String getFirstName() {*

*return firstName;*

*}*

*public final void setFirstName(String firstName) {*

*this.firstName = firstName;*

*}*

*public String getLastName() {*

*return lastName;*

*}*

*public final void setLastName(String lastName) {*

*this.lastName = lastName;*

*}*

*public String getFullName() {*

*return firstName + " " + lastName;*

*}*

*@Override*

*public String toString() {*

*return getFullName();*

*}*

*}*

*//open package for class*

*package business;*

*//import info for class*

*import java.text.NumberFormat;*

*import java.util.Locale;*

*//opening class for the pizza order info*

*public final class PizzaOrder {*

*private static final NumberFormat CF = NumberFormat.getCurrencyInstance(Locale.US);*

*//order prices and choices*

*public enum PizzaSize {*

*Small, Medium, Large*

*}*

*public static final double DEFAULT\_PRICE = 12.99;*

*public static final double MIN\_PRICE = 0;*

*public static final double MAX\_PRICE = 12.99;*

*public static final String DEFAULT\_TYPE = "Item";*

*private static final double SMALL\_PRICE = 6.99;*

*private static final double MEDIUM\_PRICE = 9.99;*

*private static final double LARGE\_PRICE = 12.99;*

*private static final double CHEESE\_COST = 1.50;*

*private static final double SAUSAGE\_COST = 2.25;*

*private static final double HAM\_COST = 2.5;*

*private final Person customer = new Person();*

*private boolean cheeseSelected = false;*

*private boolean sausageSelected = false;*

*private boolean hamSelected = false;*

*private PizzaSize pizzaSize;*

*private double sizeCost;*

*private double total = 0;*

*private boolean overrideTotal = false;*

*private int id =-1;*

*//constructors for the order info*

*public PizzaOrder() {*

*clearOrder();*

*}*

*public PizzaOrder(int id) {*

*this.id= id;*

*}*

*public int getId() {*

*return id;*

*}*

*public void setId(int id) {*

*this.id = id;*

*}*

*public void setFirstName(String firstName){*

*customer.setFirstName(firstName);*

*}*

*public String getFirstName() {*

*return customer.getFirstName();*

*}*

*public void setLastName(String lastName) {*

*customer.setLastName(lastName);*

*}*

*public String getLastName() {*

*return customer.getLastName();*

*}*

*public String getFullName() {*

*return customer.getFullName();*

*}*

*public void setCheese(boolean selected) {*

*cheeseSelected = selected;*

*}*

*public Boolean getCheese() {*

*return cheeseSelected;*

*}*

*public void setSausage(boolean selected) {*

*sausageSelected = selected;*

*}*

*public Boolean getSausage() {*

*return sausageSelected;*

*}*

*public void setHam(boolean selected) {*

*hamSelected = selected;*

*}*

*public Boolean getHam() {*

*return hamSelected;*

*}*

*public void setTotal(double total) {*

*this.total = total;*

*overrideTotal = true;*

*}*

*public double getTotal() {*

*if (!overrideTotal) {*

*calculateTotalCost();*

*}*

*return total;*

*}*

*public String getTotalCost() {*

*calculateTotalCost();*

*return CF.format(total);*

*}*

*public void setPizzaSize(PizzaSize size) {*

*pizzaSize = size;*

*}*

*//switch statement for pizza size*

*public String getPizzaSize() {*

*String str;*

*switch (pizzaSize) {*

*case Small:*

*str = "Small";*

*break;*

*case Medium:*

*str = "Medium";*

*break;*

*case Large:*

*str = "Large";*

*break;*

*default:*

*str = "Large";*

*break;*

*}*

*return str;*

*}*

*private void calculateTotalCost() {*

*total = 0;*

*switch (pizzaSize) {*

*case Small:*

*sizeCost = SMALL\_PRICE;*

*break;*

*case Medium:*

*sizeCost = MEDIUM\_PRICE;*

*break;*

*case Large:*

*sizeCost = LARGE\_PRICE;*

*break;*

*default:*

*sizeCost = LARGE\_PRICE;*

*break;*

*}*

*//if statement for pizza size with toppings*

*total = sizeCost;*

*if (cheeseSelected) {*

*total += CHEESE\_COST;*

*}*

*if (sausageSelected) {*

*total += SAUSAGE\_COST;*

*}*

*if (hamSelected) {*

*total += HAM\_COST;*

*}*

*}*

*public void setPizzaSize(String size) {*

*if(size.compareToIgnoreCase("small") == 0) {*

*pizzaSize = PizzaSize.Small;*

*}*

*else if (size.compareToIgnoreCase("medium") == 0) {*

*pizzaSize = PizzaSize.Medium;*

*}*

*else if (size.compareToIgnoreCase("large") == 0) {*

*pizzaSize = PizzaSize.Large;*

*}*

*else {*

*pizzaSize = PizzaSize.Large;*

*}*

*}*

*public void clearOrder() {*

*setPizzaSize(PizzaSize.Large);*

*setCheese(false);*

*setHam(false);*

*setSausage(false);*

*}*

*//collecting all order info*

*public String getOrderInformation() {*

*StringBuilder str = new StringBuilder();*

*getTotal();*

*str.append("Pizza order for:\n");*

*str.append(customer.getFullName());*

*if (cheeseSelected) {*

*str.append("\nExtra cheese: ");*

*str.append(CF.format(CHEESE\_COST));*

*}*

*if (sausageSelected) {*

*str.append("\nSausage: ");*

*str.append(CF.format(SAUSAGE\_COST));*

*}*

*if (hamSelected) {*

*str.append("\nHam: ");*

*str.append(CF.format(HAM\_COST));*

*}*

*str.append ("\nSize: ");*

*str.append(getPizzaSize());*

*str.append(": ");*

*str.append(CF.format(sizeCost));*

*str.append("\nTotal: ");*

*str.append(CF.format(total));*

*return str.toString();*

*}*

*@Override*

*public String toString() {*

*return customer.toString();*

*}*

*}*

*//package for class*

*package data;*

*//imports for database connections*

*import java.sql.Connection;*

*import java.sql.DriverManager;*

*import java.sql.SQLException;*

*/\*\**

*\* @author Tevis Boulware*

*\* @version 1.0*

*\* <p><b>Class Description</b></p>*

*\* <p>*

*\* This class contains the inner objects load drivers to an MySQL database,*

*\* set the connection, and manage the database connection.*

*\* </p>*

*\* <p>*

*\* Note that this object will store the user credentials in this object and transmit them*

*\* to the database in an unsecure manner. <span style="font-weight:bold; font-style:italic; color:red;">This is an unsecure technique and should only be use for "educational" purposes!</span>*

*\* In addition, the exception handling can be made more robust and do more than just print out log messages.*

*\* </p>*

*\*/*

*public class DBConnect {*

*/\*\**

*\* SQL Connection object*

*\*/*

*private Connection connection = null;*

*/\*\**

*\* string, which is the URL to the database with no user credentials*

*\*/*

*private String strCon;*

*/\*\**

*\* Registered using name of the database*

*\*/*

*private String userName;*

*/\*\**

*\* Password of the registered user <span style="font-weight:bold; font-style:italic; color:red;">this is an unsecure technique and should only be use for "educational" purposes!</span>.*

*\*/*

*private String password;*

*private boolean connected = false;*

*public static int openConnections = 0;*

*/\*\**

*\* Accepts the connection string (with no user credentials).*

*\* @param strCon string, which is the URL to the database*

*\*/*

*public DBConnect(String strCon) {*

*setConnectionString(strCon);*

*}*

*public DBConnect(String strCon, String userName, String password) {*

*setConnectionString(strCon);*

*setUserName(userName);*

*setPassword(password);*

*setConnection();*

*}*

*/\*\**

*\* Sets the database user name*

*\* @param userName string containing the registered user name. this is not a secure method*

*\* and should only be used in "educational" programs.*

*\*/*

*public void setUserName(String userName) throws IllegalArgumentException {*

*if(userName != null && !userName.trim().isEmpty()) {*

*this.userName = userName;*

*}*

*else {*

*this.userName = "";*

*throw new IllegalArgumentException("username cannot be empty");*

*}*

*}*

*/\*\**

*\* Sets the registered users password, this is not a secure method*

*\* and should only be used in "educational" programs.*

*\* @param password string containing the registered users password.*

*\*/*

*public void setPassword(String password) throws IllegalArgumentException {*

*if(password != null && !password.trim().isEmpty()) {*

*this.password = password;*

*}*

*else {*

*this.password = "";*

*throw new IllegalArgumentException("password cannot be empty");*

*}*

*}*

*/\*\**

*\* Validates the connect string is not empty, and then sets the string.*

*\* @param strCon string, which is the URL to the database*

*\*/*

*public final void setConnectionString(String strCon) throws IllegalArgumentException {*

*if(strCon != null && !strCon.trim().isEmpty()) {*

*this.strCon = strCon;*

*}*

*else {*

*this.strCon = "";*

*throw new IllegalArgumentException("connection string cannot be empty");*

*}*

*}*

*/\*\**

*\* @return the connection object.*

*\*/*

*public Connection getConnection() {*

*return connection;*

*}*

*/\*\**

*\* @return boolean value representing if the database connection to database is established and valid.*

*\*/*

*public boolean isConnected() {*

*return connected;*

*}*

*/\*\**

*\* Loads the MySQL drives, and if successful will attempt to make a connection*

*\* to the database with the given connection string, username and password.*

*\* @param conStr the URL to the database*

*\* @param userName string containing the registered user name. this is not a secure method*

*\* @param password string containing the registered users password.*

*\* @return boolean where true = drivers where loaded and connection was successful and user validated, false = means either the drives where not loaded or the connection to the database failed.*

*\* @exception ClassNotFoundException occurs when the drivers are not loaded correctly, typically caused by the my-sql-connector.jar (or other jar) not being added to project library*

*\* @exception SQLException caused by a bad connection URL or the user is not validated.*

*\* @exception Exception general catch all exception*

*\*/*

*private void setConnection() {*

*boolean valid = false;*

*connected = false;*

*if (connection == null) {*

*try {*

*Class.forName("com.mysql.jdbc.Driver");*

*valid = true;*

*}*

*catch (ClassNotFoundException e) {*

*String prompt = "\nDatabase Helpers setConnection: cannot find database drivers\n";*

*System.out.print("\n" + prompt);*

*valid = false;*

*}*

*if(valid)*

*{*

*try {*

*connection = DriverManager.getConnection(strCon, userName, password);*

*String prompt = "\nDatabase Helpers, Connected to OmnyBus\n";*

*System.out.print("\n" + prompt);*

*connected = true;*

*}*

*catch (SQLException e) {*

*String prompt = "\nDatabase Helpers setConnection: Cannot connect to database server\n";*

*System.out.print("\n" + prompt);*

*valid = false;*

*}*

*catch (Exception e) {*

*String prompt = "\nDatabase Helpers setConnection: Cannot connect to database server\n";*

*System.out.print("\n" + prompt);*

*valid = false;*

*}*

*}*

*}*

*}*

*/\*\**

*\* Closes the database connection object.*

*\* @throws SQLException if the connection object is corrupt or null.*

*\*/*

*public void closeConnection() throws SQLException {*

*try {*

*if ((connection != null) && (!connection.isClosed())) {*

*connection.close();*

*connection = null;*

*}*

*}*

*catch (SQLException ex) {*

*System.out.println(ex.getMessage());*

*}*

*}*

*}*

*//package info for class*

*package data;*

*//import data for class with SQL connectivity*

*import java.sql.PreparedStatement;*

*import java.sql.SQLException;*

*import java.sql.Statement;*

*import java.util.ArrayList;*

*import java.sql.ResultSet;*

*import java.sql.ResultSetMetaData;*

*import java.util.Vector;*

*import business.\*;*

*import java.sql.Connection;*

*import java.sql.DriverManager;*

*import javax.swing.JOptionPane;*

*public class DataIO {*

*private final String CONNECTION\_STRING = "jdbc:mysql://devry.edupe.net:8300/PizzaDB\_3503"; //db connector*

*private final String ORDER\_TABLE = "PizzaOrder"; //table name*

*private final String USER\_NAME = "3503"; //user name*

*private final String PASSWORD = "J3ck24681012"; //password*

*private final DBConnect dbConnect;*

*public DataIO() {*

*dbConnect = new DBConnect(CONNECTION\_STRING, USER\_NAME, PASSWORD);*

*}*

*public boolean add(PizzaOrder aOrder) {*

*boolean success = false;*

*PreparedStatement statement;*

*StringBuilder sqlStr = new StringBuilder();*

*int rowCount;*

*if (aOrder != null && dbConnect != null && dbConnect.isConnected()) {*

*try {*

*sqlStr.append("INSERT INTO ");*

*sqlStr.append(ORDER\_TABLE);*

*sqlStr.append(" (firstName, lastName, size, cheese, sausage, ham, total)");*

*sqlStr.append(" VALUES (?,?,?,?,?,?,?)");*

*statement = dbConnect.getConnection().prepareStatement(sqlStr.toString(), Statement.RETURN\_GENERATED\_KEYS);*

*//code to add order to the database. week 6*

*statement.executeUpdate("INSERT INTO PizzaOrder" + "(firstName, lastName, size, cheese, sausage, ham, total)" );*

*rowCount = statement.executeUpdate();*

*if (rowCount == 1) {*

*ResultSet rs = statement.getGeneratedKeys();*

*if(rs.next()) {*

*aOrder.setId(rs.getInt(1));*

*}*

*success = true;*

*}*

*}*

*catch (SQLException e) {*

*String prompt = e.getMessage()*

*+ " cannot save pizza order information for "*

*+ aOrder.getFullName();*

*JOptionPane.showMessageDialog(null, prompt, "SQL Server Error: Insert", JOptionPane.ERROR\_MESSAGE);*

*}*

*}*

*else if (aOrder == null) {*

*throw new NullPointerException("Pizza Order object is null");*

*}*

*else {*

*throw new IllegalStateException("Database is not connected");*

*}*

*return success;*

*}*

*public boolean delete(PizzaOrder aOrder) throws ClassNotFoundException {*

*boolean success = false;*

*PreparedStatement statement = null;*

*StringBuilder sqlStr = new StringBuilder();*

*int rowCount;*

*if(aOrder != null && dbConnect != null && dbConnect.isConnected()){*

*try {*

*//create the SQL statement to delete a record from the database with a given id added week 6*

*Class.forName("com.mysql.jdbc.Driver");*

*Connection connection = DriverManager.getConnection(CONNECTION\_STRING, USER\_NAME, PASSWORD);*

*PreparedStatement st = connection.prepareStatement("DELETE FROM Table WHERE name = ?");*

*st.setString(1,"id");*

*st.executeUpdate();*

*statement = dbConnect.getConnection().prepareStatement(sqlStr.toString());*

*statement.setInt(1, aOrder.getId());*

*rowCount = statement.executeUpdate();*

*if (rowCount == 1) {*

*success = true;*

*}*

*}*

*catch (SQLException e) {*

*String prompt = e.getMessage()*

*+ " cannot delete pizza order information for "*

*+ aOrder.getFullName();*

*JOptionPane.showMessageDialog(null, prompt, "SQL Server Error: Delete", JOptionPane.ERROR\_MESSAGE);*

*}*

*}*

*else if (aOrder == null) {*

*throw new NullPointerException("Pizza Order object is null");*

*}*

*else {*

*throw new IllegalStateException("Database is not connected");*

*}*

*return success;*

*}*

*public boolean update(PizzaOrder aOrder) {*

*boolean success = false;*

*PreparedStatement statement = null;*

*StringBuilder sqlStr = new StringBuilder();*

*int rowCount;*

*if(aOrder != null && dbConnect != null && dbConnect.isConnected()) {*

*try {*

*//SQL and prepared statements to update an order in the database week 6*

*Connection connection = DriverManager.getConnection(CONNECTION\_STRING, USER\_NAME, PASSWORD);*

*PreparedStatement st = connection.prepareStatement("UPDATE PizzaOrder SET id = ?, firstName = ?, lastName = ? WHERE type = ? AND total = ?");*

*rowCount = statement.executeUpdate();*

*}*

*catch (SQLException e) {*

*String prompt = e.getMessage()*

*+ " cannot update pizza order information for "*

*+ aOrder.getFullName();*

*JOptionPane.showMessageDialog(null, prompt, "SQL Server Error: Update", JOptionPane.ERROR\_MESSAGE);*

*}*

*}*

*else if (aOrder == null) {*

*throw new NullPointerException("Pizza Order object is null");*

*}*

*else {*

*throw new IllegalStateException("Database is not connected");*

*}*

*return success;*

*}*

*public ArrayList<PizzaOrder> getList() {*

*ArrayList<PizzaOrder> list = new ArrayList<>();*

*String sqlStr = "Select \* from " + ORDER\_TABLE;*

*Statement statement;*

*ResultSet rs;*

*int id;*

*PizzaOrder aOrder;*

*try {*

*statement = (Statement) dbConnect.getConnection().createStatement();*

*rs = statement.executeQuery(sqlStr);*

*while (rs.next()) {*

*aOrder = new PizzaOrder();*

*aOrder.setId(rs.getInt(1));*

*aOrder.setFirstName(rs.getString(2));*

*aOrder.setLastName(rs.getString(3));*

*//TODO create the code to:*

*//1. set the pizza size*

*aOrder.setPizzaSize(rs.getString(4));*

*//2. set the cheese selection*

*aOrder.setCheese(rs.getBoolean(5));*

*//3. set the sausage selection*

*aOrder.setSausage(rs.getBoolean(6));*

*//4. set the ham selection*

*aOrder.setHam(rs.getBoolean(7));*

*//5. set the total*

*aOrder.setTotal(rs.getDouble(8));*

*//6. add the order to the array list*

*}*

*}*

*catch (SQLException ex) {*

*String prompt = ex.getMessage()*

*+ " cannot retrieve the list from the server.";*

*JOptionPane.showMessageDialog(null, prompt, "SQL Server Error: getList", JOptionPane.ERROR\_MESSAGE);*

*}*

*return list;*

*}*

*}*

*/\**

*\* To change this license header, choose License Headers in Project Properties.*

*\* To change this template file, choose Tools | Templates*

*\* and open the template in the editor.*

*\*/*

*package presentation;*

*import business.OrderList;*

*import business.PizzaOrder;*

*import java.io.IOException;*

*import java.util.Vector;*

*import java.util.logging.Level;*

*import java.util.logging.Logger;*

*import javax.swing.DefaultListModel;*

*import javax.swing.JOptionPane;*

*public class Pizza\_Main extends javax.swing.JFrame {*

*private PizzaOrder aOrder;*

*private final DefaultListModel<PizzaOrder> listModel;*

*private final OrderList orderList;*

*public Pizza\_Main() {*

*initComponents();*

*super.setLocationRelativeTo(null);*

*newOrder();*

*listModel = new DefaultListModel<>();*

*lstOrders.setModel(listModel);*

*orderList = new OrderList();*

*retrieveOrderList();*

*}*

*private void retrieveOrderList() {*

*int count = 0;*

*String message;*

*count = orderList.setList(listModel);*

*message = count + " pizza order records retrieved";*

*JOptionPane.showMessageDialog(null, message, "Pizza Orders Retrieved", JOptionPane.INFORMATION\_MESSAGE);*

*setOrderSummary();*

*}*

*private void setOrderSummary() {*

*txtOrderSummary.setText(orderList.getOrderSummary());*

*}*

*private void showOrderDetails() {*

*if (lstOrders.getSelectedIndex() != -1) {*

*aOrder = lstOrders.getSelectedValue();*

*txtOrderDetails.setText(aOrder.getOrderInformation());*

*}*

*}*

*private void submitOrder() {*

*StringBuilder str = new StringBuilder();*

*boolean success = false;*

*String prompt;*

*if (aOrder != null) {*

*str.append("Order submitted\n");*

*str.append (aOrder.getOrderInformation());*

*listModel.addElement(aOrder);*

*success = orderList.add(aOrder);*

*if (success) {*

*prompt = "Succesfully added " + aOrder.getFullName() + " to the database";*

*JOptionPane.showMessageDialog(null, prompt, "Record added successfully", JOptionPane.INFORMATION\_MESSAGE);*

*setOrderSummary();*

*}*

*else {*

*prompt = "DID NOT successfully add " + aOrder.getFullName() + " to the database";*

*JOptionPane.showMessageDialog(null, prompt, "Record NOT added successfully", JOptionPane.ERROR);*

*}*

*}*

*}*

*private void deleteOrder() throws ClassNotFoundException {*

*int item = lstOrders.getSelectedIndex();*

*if (item != -1) {*

*int response;*

*PizzaOrder theOrder = lstOrders.getSelectedValue();*

*String prompt = "Are you sure you want to delete order for: " + theOrder.toString();*

*response = JOptionPane.showConfirmDialog(null, prompt, "Delete order " + theOrder.toString(),*

*JOptionPane.OK\_CANCEL\_OPTION, JOptionPane.QUESTION\_MESSAGE);*

*if (response == JOptionPane.YES\_OPTION) {*

*listModel.removeElement(theOrder);*

*orderList.remove(theOrder);*

*txtOrderDetails.setText("");*

*setOrderSummary();*

*}*

*}*

*}*

*private void updateOrder() {*

*double value;*

*try {*

*value = Double.parseDouble(txtTotal.getText());*

*aOrder.setTotal(value);*

*txtOrderDetails.setText(aOrder.getOrderInformation());*

*orderList.update(aOrder);*

*setOrderSummary();*

*}*

*catch (NumberFormatException ex) {*

*String message = txtTotal.getText() + " value is an illegal, enter a number greater than 0";*

*JOptionPane.showMessageDialog(null, message, "Illegal total value", JOptionPane.ERROR\_MESSAGE);*

*}*

*}*

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\* existing functions from previous iteration \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

*private void setLastName() {*

*if (aOrder != null) {*

*aOrder.setLastName(txtLastName.getText());*

*}*

*}*

*private void setFirstName() {*

*if (aOrder != null) {*

*aOrder.setFirstName(txtFirstName.getText());*

*}*

*}*

*private void newOrder() {*

*aOrder = new PizzaOrder();*

*clearOrder();*

*}*

*private void clearOrder() {*

*txtFirstName.setText("");*

*txtLastName.setText("");*

*radLarge.setSelected(true);*

*chkExtraCheese.setSelected(false);*

*chkSausage.setSelected(false);*

*chkHam.setSelected(false);*

*lblSubTotal.setText("");*

*aOrder.clearOrder();*

*}*

*private void setSize(PizzaOrder.PizzaSize size) {*

*aOrder.setPizzaSize(size);*

*setSubTotal();*

*}*

*private void setCheese() {*

*if (chkExtraCheese.isSelected()) {*

*aOrder.setCheese(true);*

*}*

*else {*

*aOrder.setCheese(false);*

*}*

*setSubTotal();*

*}*

*private void setSausage() {*

*if (chkSausage.isSelected()) {*

*aOrder.setSausage(true);*

*}*

*else {*

*aOrder.setSausage(false);*

*}*

*setSubTotal();*

*}*

*private void setHam() {*

*if (chkHam.isSelected()) {*

*aOrder.setHam(true);*

*}*

*else {*

*aOrder.setHam(false);*

*}*

*setSubTotal();*

*}*

*private void setSubTotal() {*

*lblSubTotal.setText(aOrder.getTotalCost());*

*}*

*private void terminateApplication() {*

*int response = JOptionPane.NO\_OPTION;*

*String prompt = "Do you want to quit the pizza program?";*

*String title = "Quit Pizza Program";*

*response = JOptionPane.showConfirmDialog(null, prompt, title,*

*JOptionPane.OK\_CANCEL\_OPTION,*

*JOptionPane.QUESTION\_MESSAGE);*

*if(response == JOptionPane.YES\_OPTION) {*

*System.exit(0);*

*}*

*}*

*/\*\**

*\* This method is called from within the constructor to initialize the form.*

*\* WARNING: Do NOT modify this code. The content of this method is always*

*\* regenerated by the Form Editor.*

*\*/*

*@SuppressWarnings("unchecked")*

*// <editor-fold defaultstate="collapsed" desc="Generated Code">*

*private void initComponents() {*

*grpSize = new javax.swing.ButtonGroup();*

*tabPizzaInformation = new javax.swing.JTabbedPane();*

*jPanel3 = new javax.swing.JPanel();*

*pnlOrder = new javax.swing.JPanel();*

*jLabel4 = new javax.swing.JLabel();*

*pnlCustomer = new javax.swing.JPanel();*

*jLabel5 = new javax.swing.JLabel();*

*txtFirstName = new javax.swing.JTextField();*

*jLabel6 = new javax.swing.JLabel();*

*txtLastName = new javax.swing.JTextField();*

*pnlSize = new javax.swing.JPanel();*

*radSmall = new javax.swing.JRadioButton();*

*radMedium = new javax.swing.JRadioButton();*

*radLarge = new javax.swing.JRadioButton();*

*pnlToppings = new javax.swing.JPanel();*

*chkExtraCheese = new javax.swing.JCheckBox();*

*chkSausage = new javax.swing.JCheckBox();*

*chkHam = new javax.swing.JCheckBox();*

*btnSubmitOrder = new javax.swing.JButton();*

*btnNewOrder = new javax.swing.JButton();*

*btnClear = new javax.swing.JButton();*

*lblTotal = new javax.swing.JLabel();*

*lblSubTotal = new javax.swing.JLabel();*

*jPanel1 = new javax.swing.JPanel();*

*jLabel1 = new javax.swing.JLabel();*

*jScrollPane1 = new javax.swing.JScrollPane();*

*lstOrders = new javax.swing.JList<>();*

*jLabel2 = new javax.swing.JLabel();*

*jScrollPane2 = new javax.swing.JScrollPane();*

*txtOrderDetails = new javax.swing.JTextArea();*

*btnDelete = new javax.swing.JButton();*

*btnUpdate = new javax.swing.JButton();*

*txtTotal = new javax.swing.JTextField();*

*jPanel2 = new javax.swing.JPanel();*

*jScrollPane3 = new javax.swing.JScrollPane();*

*txtOrderSummary = new javax.swing.JTextArea();*

*lblWelcome = new javax.swing.JLabel();*

*btnExit = new javax.swing.JButton();*

*mnuBar = new javax.swing.JMenuBar();*

*mnuOrder = new javax.swing.JMenu();*

*mnuNewOrder = new javax.swing.JMenuItem();*

*mnuSubmit = new javax.swing.JMenuItem();*

*mnuClearOrder = new javax.swing.JMenuItem();*

*mnuSummary = new javax.swing.JMenu();*

*mnuDelete = new javax.swing.JMenuItem();*

*setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);*

*setTitle("Best Pizza Ordering");*

*tabPizzaInformation.setForeground(new java.awt.Color(0, 0, 80));*

*tabPizzaInformation.setFont(new java.awt.Font("Tahoma", 3, 12)); // NOI18N*

*pnlOrder.setPreferredSize(new java.awt.Dimension(579, 550));*

*jLabel4.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N*

*jLabel4.setText("Please enter your name and select your ordering options:");*

*pnlCustomer.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "Customer Information", javax.swing.border.TitledBorder.DEFAULT\_JUSTIFICATION, javax.swing.border.TitledBorder.DEFAULT\_POSITION, new java.awt.Font("Tahoma", 1, 11), new java.awt.Color(51, 51, 255))); // NOI18N*

*jLabel5.setText("First Name:");*

*txtFirstName.setToolTipText("Enter your first name");*

*txtFirstName.addFocusListener(new java.awt.event.FocusAdapter() {*

*public void focusLost(java.awt.event.FocusEvent evt) {*

*txtFirstNameFocusLost(evt);*

*}*

*});*

*jLabel6.setText("Last Name:");*

*txtLastName.setToolTipText("Enter your last name");*

*txtLastName.addFocusListener(new java.awt.event.FocusAdapter() {*

*public void focusLost(java.awt.event.FocusEvent evt) {*

*txtLastNameFocusLost(evt);*

*}*

*});*

*javax.swing.GroupLayout pnlCustomerLayout = new javax.swing.GroupLayout(pnlCustomer);*

*pnlCustomer.setLayout(pnlCustomerLayout);*

*pnlCustomerLayout.setHorizontalGroup(*

*pnlCustomerLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(pnlCustomerLayout.createSequentialGroup()*

*.addContainerGap()*

*.addGroup(pnlCustomerLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addComponent(jLabel5)*

*.addComponent(jLabel6))*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)*

*.addGroup(pnlCustomerLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)*

*.addComponent(txtFirstName)*

*.addComponent(txtLastName, javax.swing.GroupLayout.DEFAULT\_SIZE, 399, Short.MAX\_VALUE))*

*.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))*

*);*

*pnlCustomerLayout.setVerticalGroup(*

*pnlCustomerLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(pnlCustomerLayout.createSequentialGroup()*

*.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)*

*.addGroup(pnlCustomerLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)*

*.addComponent(jLabel5)*

*.addComponent(txtFirstName, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)*

*.addGroup(pnlCustomerLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)*

*.addComponent(jLabel6)*

*.addComponent(txtLastName, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)))*

*);*

*pnlSize.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "Pizza Size (please select one)", javax.swing.border.TitledBorder.DEFAULT\_JUSTIFICATION, javax.swing.border.TitledBorder.DEFAULT\_POSITION, new java.awt.Font("Tahoma", 1, 11), new java.awt.Color(51, 51, 255))); // NOI18N*

*grpSize.add(radSmall);*

*radSmall.setText("Small: $6.99 ");*

*radSmall.setToolTipText("Select for small pizza at $6.99");*

*radSmall.addItemListener(new java.awt.event.ItemListener() {*

*public void itemStateChanged(java.awt.event.ItemEvent evt) {*

*radSmallItemStateChanged(evt);*

*}*

*});*

*grpSize.add(radMedium);*

*radMedium.setText("Medium: $9.99 ");*

*radMedium.setToolTipText("Select for medium pizza at $9.99");*

*radMedium.addItemListener(new java.awt.event.ItemListener() {*

*public void itemStateChanged(java.awt.event.ItemEvent evt) {*

*radMediumItemStateChanged(evt);*

*}*

*});*

*grpSize.add(radLarge);*

*radLarge.setSelected(true);*

*radLarge.setText("Large: $12.99");*

*radLarge.setToolTipText("Select for large pizza at $12.99");*

*radLarge.addItemListener(new java.awt.event.ItemListener() {*

*public void itemStateChanged(java.awt.event.ItemEvent evt) {*

*radLargeItemStateChanged(evt);*

*}*

*});*

*javax.swing.GroupLayout pnlSizeLayout = new javax.swing.GroupLayout(pnlSize);*

*pnlSize.setLayout(pnlSizeLayout);*

*pnlSizeLayout.setHorizontalGroup(*

*pnlSizeLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(pnlSizeLayout.createSequentialGroup()*

*.addContainerGap()*

*.addGroup(pnlSizeLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addComponent(radMedium)*

*.addComponent(radSmall)*

*.addComponent(radLarge))*

*.addContainerGap(438, Short.MAX\_VALUE))*

*);*

*pnlSizeLayout.setVerticalGroup(*

*pnlSizeLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(pnlSizeLayout.createSequentialGroup()*

*.addComponent(radSmall)*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)*

*.addComponent(radMedium)*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)*

*.addComponent(radLarge))*

*);*

*pnlToppings.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "Toppings (Select as many as you like)", javax.swing.border.TitledBorder.DEFAULT\_JUSTIFICATION, javax.swing.border.TitledBorder.DEFAULT\_POSITION, new java.awt.Font("Tahoma", 1, 11), new java.awt.Color(51, 51, 255))); // NOI18N*

*chkExtraCheese.setText("Extra Cheese: $1.50 ");*

*chkExtraCheese.setToolTipText("Select to add extra cheese for $1.50");*

*chkExtraCheese.addItemListener(new java.awt.event.ItemListener() {*

*public void itemStateChanged(java.awt.event.ItemEvent evt) {*

*chkExtraCheeseItemStateChanged(evt);*

*}*

*});*

*chkSausage.setText("Sausage: $2.25");*

*chkSausage.setToolTipText("Check to add Sausage for $2.25");*

*chkSausage.addActionListener(new java.awt.event.ActionListener() {*

*public void actionPerformed(java.awt.event.ActionEvent evt) {*

*chkSausageActionPerformed(evt);*

*}*

*});*

*chkHam.setText("Ham: $2.50");*

*chkHam.setToolTipText("Check to add Ham for $2.50");*

*chkHam.addItemListener(new java.awt.event.ItemListener() {*

*public void itemStateChanged(java.awt.event.ItemEvent evt) {*

*chkHamItemStateChanged(evt);*

*}*

*});*

*javax.swing.GroupLayout pnlToppingsLayout = new javax.swing.GroupLayout(pnlToppings);*

*pnlToppings.setLayout(pnlToppingsLayout);*

*pnlToppingsLayout.setHorizontalGroup(*

*pnlToppingsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(pnlToppingsLayout.createSequentialGroup()*

*.addContainerGap()*

*.addGroup(pnlToppingsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addComponent(chkExtraCheese)*

*.addComponent(chkSausage)*

*.addComponent(chkHam))*

*.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))*

*);*

*pnlToppingsLayout.setVerticalGroup(*

*pnlToppingsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(pnlToppingsLayout.createSequentialGroup()*

*.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)*

*.addComponent(chkExtraCheese)*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)*

*.addComponent(chkSausage)*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)*

*.addComponent(chkHam))*

*);*

*btnSubmitOrder.setText("Submit Order");*

*btnSubmitOrder.setToolTipText("Click to show the order total and order summary, and store the order");*

*btnSubmitOrder.addActionListener(new java.awt.event.ActionListener() {*

*public void actionPerformed(java.awt.event.ActionEvent evt) {*

*btnSubmitOrderActionPerformed(evt);*

*}*

*});*

*btnNewOrder.setText("New Order");*

*btnNewOrder.setToolTipText("Click to clear all fields and start a new order.");*

*btnNewOrder.addActionListener(new java.awt.event.ActionListener() {*

*public void actionPerformed(java.awt.event.ActionEvent evt) {*

*btnNewOrderActionPerformed(evt);*

*}*

*});*

*btnClear.setText("Clear Order");*

*btnClear.setToolTipText("Click to clear all fields and start a new order.");*

*btnClear.addActionListener(new java.awt.event.ActionListener() {*

*public void actionPerformed(java.awt.event.ActionEvent evt) {*

*btnClearActionPerformed(evt);*

*}*

*});*

*lblTotal.setFont(new java.awt.Font("Tahoma", 3, 12)); // NOI18N*

*lblTotal.setForeground(new java.awt.Color(0, 0, 80));*

*lblTotal.setText("Subtotal:");*

*lblSubTotal.setFont(new java.awt.Font("Tahoma", 3, 12)); // NOI18N*

*lblSubTotal.setForeground(new java.awt.Color(0, 0, 80));*

*lblSubTotal.setText("$0");*

*javax.swing.GroupLayout pnlOrderLayout = new javax.swing.GroupLayout(pnlOrder);*

*pnlOrder.setLayout(pnlOrderLayout);*

*pnlOrderLayout.setHorizontalGroup(*

*pnlOrderLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(pnlOrderLayout.createSequentialGroup()*

*.addContainerGap()*

*.addGroup(pnlOrderLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addComponent(pnlCustomer, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)*

*.addGroup(pnlOrderLayout.createSequentialGroup()*

*.addComponent(jLabel4)*

*.addGap(0, 206, Short.MAX\_VALUE))*

*.addComponent(pnlToppings, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)*

*.addComponent(pnlSize, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)*

*.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, pnlOrderLayout.createSequentialGroup()*

*.addComponent(lblTotal, javax.swing.GroupLayout.PREFERRED\_SIZE, 67, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)*

*.addComponent(lblSubTotal, javax.swing.GroupLayout.PREFERRED\_SIZE, 67, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)*

*.addComponent(btnNewOrder, javax.swing.GroupLayout.PREFERRED\_SIZE, 114, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)*

*.addComponent(btnClear, javax.swing.GroupLayout.PREFERRED\_SIZE, 114, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)*

*.addComponent(btnSubmitOrder, javax.swing.GroupLayout.PREFERRED\_SIZE, 115, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addGap(4, 4, 4)))*

*.addContainerGap())*

*);*

*pnlOrderLayout.setVerticalGroup(*

*pnlOrderLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(pnlOrderLayout.createSequentialGroup()*

*.addContainerGap()*

*.addComponent(jLabel4)*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)*

*.addComponent(pnlCustomer, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)*

*.addComponent(pnlSize, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addGap(18, 18, 18)*

*.addComponent(pnlToppings, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addGap(18, 18, 18)*

*.addGroup(pnlOrderLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)*

*.addComponent(btnClear)*

*.addComponent(lblTotal)*

*.addComponent(lblSubTotal)*

*.addComponent(btnNewOrder)*

*.addComponent(btnSubmitOrder))*

*.addContainerGap(61, Short.MAX\_VALUE))*

*);*

*javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);*

*jPanel3.setLayout(jPanel3Layout);*

*jPanel3Layout.setHorizontalGroup(*

*jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel3Layout.createSequentialGroup()*

*.addContainerGap(23, Short.MAX\_VALUE)*

*.addComponent(pnlOrder, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addContainerGap())*

*);*

*jPanel3Layout.setVerticalGroup(*

*jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(jPanel3Layout.createSequentialGroup()*

*.addContainerGap()*

*.addComponent(pnlOrder, javax.swing.GroupLayout.PREFERRED\_SIZE, 440, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))*

*);*

*tabPizzaInformation.addTab("Pizza Order Information", jPanel3);*

*jLabel1.setFont(new java.awt.Font("Tahoma", 1, 11)); // NOI18N*

*jLabel1.setForeground(new java.awt.Color(0, 0, 80));*

*jLabel1.setText("Order List");*

*lstOrders.setSelectionMode(javax.swing.ListSelectionModel.SINGLE\_SELECTION);*

*lstOrders.addListSelectionListener(new javax.swing.event.ListSelectionListener() {*

*public void valueChanged(javax.swing.event.ListSelectionEvent evt) {*

*lstOrdersValueChanged(evt);*

*}*

*});*

*jScrollPane1.setViewportView(lstOrders);*

*jLabel2.setFont(new java.awt.Font("Tahoma", 1, 11)); // NOI18N*

*jLabel2.setForeground(new java.awt.Color(0, 0, 80));*

*jLabel2.setText("Order Details");*

*txtOrderDetails.setColumns(20);*

*txtOrderDetails.setRows(5);*

*jScrollPane2.setViewportView(txtOrderDetails);*

*btnDelete.setFont(new java.awt.Font("Tahoma", 1, 11)); // NOI18N*

*btnDelete.setForeground(new java.awt.Color(0, 0, 80));*

*btnDelete.setText("Delete Order");*

*btnDelete.setToolTipText("Click to delete the selected pizza order");*

*btnDelete.addActionListener(new java.awt.event.ActionListener() {*

*public void actionPerformed(java.awt.event.ActionEvent evt) {*

*btnDeleteActionPerformed(evt);*

*}*

*});*

*btnUpdate.setFont(new java.awt.Font("Tahoma", 1, 11)); // NOI18N*

*btnUpdate.setForeground(new java.awt.Color(0, 0, 80));*

*btnUpdate.setText("Update Total");*

*btnUpdate.setToolTipText("Click to update the total cost of the order");*

*btnUpdate.addActionListener(new java.awt.event.ActionListener() {*

*public void actionPerformed(java.awt.event.ActionEvent evt) {*

*btnUpdateActionPerformed(evt);*

*}*

*});*

*txtTotal.setToolTipText("Override calculated total price of the order");*

*jPanel2.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "Order Total Summary", javax.swing.border.TitledBorder.DEFAULT\_JUSTIFICATION, javax.swing.border.TitledBorder.DEFAULT\_POSITION, new java.awt.Font("Tahoma", 1, 11), new java.awt.Color(0, 0, 80))); // NOI18N*

*jPanel2.setForeground(new java.awt.Color(0, 0, 80));*

*txtOrderSummary.setColumns(20);*

*txtOrderSummary.setRows(5);*

*jScrollPane3.setViewportView(txtOrderSummary);*

*javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);*

*jPanel2.setLayout(jPanel2Layout);*

*jPanel2Layout.setHorizontalGroup(*

*jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(jPanel2Layout.createSequentialGroup()*

*.addContainerGap()*

*.addComponent(jScrollPane3)*

*.addContainerGap())*

*);*

*jPanel2Layout.setVerticalGroup(*

*jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addComponent(jScrollPane3, javax.swing.GroupLayout.DEFAULT\_SIZE, 191, Short.MAX\_VALUE)*

*);*

*javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);*

*jPanel1.setLayout(jPanel1Layout);*

*jPanel1Layout.setHorizontalGroup(*

*jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(jPanel1Layout.createSequentialGroup()*

*.addGap(30, 30, 30)*

*.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addComponent(jLabel1)*

*.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 200, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addComponent(btnDelete, javax.swing.GroupLayout.PREFERRED\_SIZE, 115, javax.swing.GroupLayout.PREFERRED\_SIZE))*

*.addGap(68, 68, 68)*

*.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(jPanel1Layout.createSequentialGroup()*

*.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addComponent(jLabel2)*

*.addGroup(jPanel1Layout.createSequentialGroup()*

*.addComponent(btnUpdate, javax.swing.GroupLayout.PREFERRED\_SIZE, 115, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)*

*.addComponent(txtTotal, javax.swing.GroupLayout.DEFAULT\_SIZE, 79, Short.MAX\_VALUE)))*

*.addGap(114, 114, 114))*

*.addGroup(jPanel1Layout.createSequentialGroup()*

*.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 281, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))))*

*.addGroup(jPanel1Layout.createSequentialGroup()*

*.addGap(19, 19, 19)*

*.addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)*

*.addContainerGap())*

*);*

*jPanel1Layout.setVerticalGroup(*

*jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(jPanel1Layout.createSequentialGroup()*

*.addGap(25, 25, 25)*

*.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)*

*.addComponent(jLabel1)*

*.addComponent(jLabel2))*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)*

*.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)*

*.addComponent(jScrollPane1)*

*.addComponent(jScrollPane2))*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)*

*.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)*

*.addComponent(btnDelete)*

*.addComponent(btnUpdate)*

*.addComponent(txtTotal, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))*

*.addGap(18, 18, 18)*

*.addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)*

*.addContainerGap())*

*);*

*tabPizzaInformation.addTab("Order Summary", jPanel1);*

*lblWelcome.setFont(new java.awt.Font("Arial", 3, 24)); // NOI18N*

*lblWelcome.setForeground(new java.awt.Color(0, 0, 255));*

*lblWelcome.setText("Welcome to Best Pizza!");*

*lblWelcome.setVerticalAlignment(javax.swing.SwingConstants.TOP);*

*lblWelcome.setName("lblWelcome"); // NOI18N*

*btnExit.setText("Exit");*

*btnExit.addActionListener(new java.awt.event.ActionListener() {*

*public void actionPerformed(java.awt.event.ActionEvent evt) {*

*btnExitActionPerformed(evt);*

*}*

*});*

*mnuOrder.setText("Pizza Order");*

*mnuNewOrder.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_N, java.awt.event.InputEvent.ALT\_MASK));*

*mnuNewOrder.setMnemonic('n');*

*mnuNewOrder.setText("New order");*

*mnuNewOrder.setToolTipText("Click to create a new order");*

*mnuNewOrder.addActionListener(new java.awt.event.ActionListener() {*

*public void actionPerformed(java.awt.event.ActionEvent evt) {*

*mnuNewOrderActionPerformed(evt);*

*}*

*});*

*mnuOrder.add(mnuNewOrder);*

*mnuSubmit.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_S, java.awt.event.InputEvent.ALT\_MASK));*

*mnuSubmit.setText("Submit Order");*

*mnuSubmit.setToolTipText("Click to submit the order");*

*mnuSubmit.addActionListener(new java.awt.event.ActionListener() {*

*public void actionPerformed(java.awt.event.ActionEvent evt) {*

*mnuSubmitActionPerformed(evt);*

*}*

*});*

*mnuOrder.add(mnuSubmit);*

*mnuClearOrder.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_C, java.awt.event.InputEvent.ALT\_MASK));*

*mnuClearOrder.setText("Clear Order");*

*mnuClearOrder.setToolTipText("Click to clear the order fields");*

*mnuClearOrder.addActionListener(new java.awt.event.ActionListener() {*

*public void actionPerformed(java.awt.event.ActionEvent evt) {*

*mnuClearOrderActionPerformed(evt);*

*}*

*});*

*mnuOrder.add(mnuClearOrder);*

*mnuBar.add(mnuOrder);*

*mnuSummary.setText("Order Summary");*

*mnuDelete.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK\_DELETE, 0));*

*mnuDelete.setText("Delete Order");*

*mnuDelete.addActionListener(new java.awt.event.ActionListener() {*

*public void actionPerformed(java.awt.event.ActionEvent evt) {*

*mnuDeleteActionPerformed(evt);*

*}*

*});*

*mnuSummary.add(mnuDelete);*

*mnuBar.add(mnuSummary);*

*setJMenuBar(mnuBar);*

*javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());*

*getContentPane().setLayout(layout);*

*layout.setHorizontalGroup(*

*layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(layout.createSequentialGroup()*

*.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(layout.createSequentialGroup()*

*.addGap(18, 18, 18)*

*.addComponent(lblWelcome))*

*.addGroup(layout.createSequentialGroup()*

*.addContainerGap()*

*.addComponent(tabPizzaInformation, javax.swing.GroupLayout.PREFERRED\_SIZE, 617, javax.swing.GroupLayout.PREFERRED\_SIZE)))*

*.addContainerGap(19, Short.MAX\_VALUE))*

*.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()*

*.addGap(0, 0, Short.MAX\_VALUE)*

*.addComponent(btnExit, javax.swing.GroupLayout.PREFERRED\_SIZE, 115, javax.swing.GroupLayout.PREFERRED\_SIZE)*

*.addGap(29, 29, 29))*

*);*

*layout.setVerticalGroup(*

*layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)*

*.addGroup(layout.createSequentialGroup()*

*.addContainerGap()*

*.addComponent(lblWelcome)*

*.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)*

*.addComponent(tabPizzaInformation, javax.swing.GroupLayout.PREFERRED\_SIZE, 481, Short.MAX\_VALUE)*

*.addGap(18, 18, 18)*

*.addComponent(btnExit)*

*.addContainerGap())*

*);*

*pack();*

*}// </editor-fold>*

*private void btnExitActionPerformed(java.awt.event.ActionEvent evt) {*

*terminateApplication();*

*}*

*private void mnuNewOrderActionPerformed(java.awt.event.ActionEvent evt) {*

*newOrder();*

*}*

*private void mnuSubmitActionPerformed(java.awt.event.ActionEvent evt) {*

*submitOrder();*

*}*

*private void mnuDeleteActionPerformed(java.awt.event.ActionEvent evt) {*

*try {*

*deleteOrder();*

*} catch (ClassNotFoundException ex) {*

*Logger.getLogger(Pizza\_Main.class.getName()).log(Level.SEVERE, null, ex);*

*}*

*}*

*private void mnuClearOrderActionPerformed(java.awt.event.ActionEvent evt) {*

*clearOrder();*

*}*

*private void btnUpdateActionPerformed(java.awt.event.ActionEvent evt) {*

*updateOrder();*

*}*

*private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {*

*try {*

*deleteOrder();*

*} catch (ClassNotFoundException ex) {*

*Logger.getLogger(Pizza\_Main.class.getName()).log(Level.SEVERE, null, ex);*

*}*

*}*

*private void lstOrdersValueChanged(javax.swing.event.ListSelectionEvent evt) {*

*showOrderDetails();*

*}*

*private void btnClearActionPerformed(java.awt.event.ActionEvent evt) {*

*clearOrder();*

*}*

*private void btnNewOrderActionPerformed(java.awt.event.ActionEvent evt) {*

*newOrder();*

*}*

*private void btnSubmitOrderActionPerformed(java.awt.event.ActionEvent evt) {*

*submitOrder();*

*}*

*private void chkHamItemStateChanged(java.awt.event.ItemEvent evt) {*

*setHam();*

*}*

*private void chkSausageActionPerformed(java.awt.event.ActionEvent evt) {*

*setSausage();*

*}*

*private void chkExtraCheeseItemStateChanged(java.awt.event.ItemEvent evt) {*

*setCheese();*

*}*

*private void radLargeItemStateChanged(java.awt.event.ItemEvent evt) {*

*setSize(PizzaOrder.PizzaSize.Small);*

*}*

*private void radMediumItemStateChanged(java.awt.event.ItemEvent evt) {*

*setSize(PizzaOrder.PizzaSize.Medium);*

*}*

*private void radSmallItemStateChanged(java.awt.event.ItemEvent evt) {*

*setSize(PizzaOrder.PizzaSize.Large);*

*}*

*private void txtLastNameFocusLost(java.awt.event.FocusEvent evt) {*

*setLastName();*

*}*

*private void txtFirstNameFocusLost(java.awt.event.FocusEvent evt) {*

*setFirstName();*

*}*

*/\*\**

*\* @param args the command line arguments*

*\*/*

*public static void main(String args[]) {*

*/\* Set the Nimbus look and feel \*/*

*//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">*

*/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.*

*\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html*

*\*/*

*try {*

*for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {*

*if ("Nimbus".equals(info.getName())) {*

*javax.swing.UIManager.setLookAndFeel(info.getClassName());*

*break;*

*}*

*}*

*} catch (ClassNotFoundException ex) {*

*java.util.logging.Logger.getLogger(Pizza\_Main.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);*

*} catch (InstantiationException ex) {*

*java.util.logging.Logger.getLogger(Pizza\_Main.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);*

*} catch (IllegalAccessException ex) {*

*java.util.logging.Logger.getLogger(Pizza\_Main.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);*

*} catch (javax.swing.UnsupportedLookAndFeelException ex) {*

*java.util.logging.Logger.getLogger(Pizza\_Main.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);*

*}*

*//</editor-fold>*

*//</editor-fold>*

*/\* Create and display the form \*/*

*java.awt.EventQueue.invokeLater(new Runnable() {*

*public void run() {*

*Pizza\_Main mainForm = new Pizza\_Main();*

*mainForm.setVisible(true);*

*mainForm.setLocationRelativeTo(null);*

*}*

*});*

*}*

*// Variables declaration - do not modify*

*private javax.swing.JButton btnClear;*

*private javax.swing.JButton btnDelete;*

*private javax.swing.JButton btnExit;*

*private javax.swing.JButton btnNewOrder;*

*private javax.swing.JButton btnSubmitOrder;*

*private javax.swing.JButton btnUpdate;*

*private javax.swing.JCheckBox chkExtraCheese;*

*private javax.swing.JCheckBox chkHam;*

*private javax.swing.JCheckBox chkSausage;*

*private javax.swing.ButtonGroup grpSize;*

*private javax.swing.JLabel jLabel1;*

*private javax.swing.JLabel jLabel2;*

*private javax.swing.JLabel jLabel4;*

*private javax.swing.JLabel jLabel5;*

*private javax.swing.JLabel jLabel6;*

*private javax.swing.JPanel jPanel1;*

*private javax.swing.JPanel jPanel2;*

*private javax.swing.JPanel jPanel3;*

*private javax.swing.JScrollPane jScrollPane1;*

*private javax.swing.JScrollPane jScrollPane2;*

*private javax.swing.JScrollPane jScrollPane3;*

*private javax.swing.JLabel lblSubTotal;*

*private javax.swing.JLabel lblTotal;*

*private javax.swing.JLabel lblWelcome;*

*private javax.swing.JList<PizzaOrder> lstOrders;*

*private javax.swing.JMenuBar mnuBar;*

*private javax.swing.JMenuItem mnuClearOrder;*

*private javax.swing.JMenuItem mnuDelete;*

*private javax.swing.JMenuItem mnuNewOrder;*

*private javax.swing.JMenu mnuOrder;*

*private javax.swing.JMenuItem mnuSubmit;*

*private javax.swing.JMenu mnuSummary;*

*private javax.swing.JPanel pnlCustomer;*

*private javax.swing.JPanel pnlOrder;*

*private javax.swing.JPanel pnlSize;*

*private javax.swing.JPanel pnlToppings;*

*private javax.swing.JRadioButton radLarge;*

*private javax.swing.JRadioButton radMedium;*

*private javax.swing.JRadioButton radSmall;*

*private javax.swing.JTabbedPane tabPizzaInformation;*

*private javax.swing.JTextField txtFirstName;*

*private javax.swing.JTextField txtLastName;*

*private javax.swing.JTextArea txtOrderDetails;*

*private javax.swing.JTextArea txtOrderSummary;*

*private javax.swing.JTextField txtTotal;*

*// End of variables declaration*

*}*