**PROJECT SUBMISSION  
Final Project Yummy Catering**

**Each individual java file contains comments explaining what each code block does.  
In these instructions, filenames for the files that I created are in red.   
The instructions to use to run the file are in green.  
Skills I used (from each chapter) are in purple.**

**Ch. 1 - Done**

The case problems in this section introduce two fictional businesses.

Yummy Catering provides meals for parties and special events. Write a program that displays Yummy Catering’s motto, which is Yummy makes the food that makes it a party. Save the file as YummyMotto.java. **To run this program, click on YummyMotto.java and Yummy Catering’s motto will appear in the console.**

Create a second program that displays the motto surrounded by a border composed of asterisks. Save the file as YummyMotto2.java. **To run this program, click on YummyMotto2.java and Yummy Catering’s motto will appear in the console.**

**Skills used -   
Analyze a Java application that produces console output  
Compile a Java class and correct syntax errors   
Run a Java application and correct logic errors  
Add comments to a Java class**

**Ch. 2 - Done**

Yummy Catering provides meals for parties and special events. Write a program that prompts the user for the number of guests attending an event and then computes the total price, which is $35 per person. Display the company motto with the border that you created in the YummyMotto2 class in Chapter 1, and then display the number of guests, price per guest, and total price. Also display a message that indicates true or false depending on whether the job is classified as a large event—an event with 50 or more guests. Save the file as YummyEventPrice.java. **To run this program, click on YummyEventPrice.java and user is asked to enter number of guests. The total price of the event will be generated as well as whether this event is a large event.**

**Skills used -   
Declare and use constants and variables  
Use integer data types  
Use the boolean data type  
Use floating-point data types  
Use the char data type  
Use the Scanner class to accept keyboard input  
Use the JOptionPane class to accept GUI input   
Perform arithmetic using variables and constants**

**Ch3 - Done**

Yummy Catering provides meals for parties and special events. In Chapter 2, you wrote an application that prompts the user for the number of guests attending an event, displays the company motto with a border, and then displays the price of the event and whether the event is a large one. Now modify the program so that the main() method contains only three executable statements that each call a method as follows:

❯ The first executable statement calls a public static int method that prompts the user for the number of guests and returns the value to the main() method.

❯ The second executable statement calls a public static void method that displays the company motto with the border.

❯ The last executable statement passes the number of guests to a public static void method that computes the price of the event, displays the price, and displays whether the event is a large event.

Save the file as YummyEventPriceWithMethods.java. **To run this program, click on YummyEventPriceWithMethods.java and user is asked to enter number of guests. The total price of the event will be generated as well the Yummy moto along with whether this event is a large event. This will output to the console.**

**Skills used -   
Identify the parts of a method   
Add parameters to methods   
Create methods that return values  
Overload a method**

**Ch 4 - Done**

Yummy Catering provides meals for parties and special events. In previous chapters, you have developed an Event class that holds catering event information. Now modify the Event class as follows:

Modify the method that sets the event number in the Event class so that if the argument passed to the method is not a four-character String that starts with a letter followed by three digits, then the event number is forced to A000. If the initial letter in the event number is not uppercase, force it to be so.

Add a contact phone number field to the Event class.

Add a set method for the contact phone number field in the Event class. Whether the user enters all digits or any combination of digits, spaces, dashes, dots, or parentheses for a phone number, store it as all digits. (Assume a phone number is 10 digits and does not require a leading digit that might be needed when dialing.) For example, if the user enters (920) 872-9182, store the phone number as 9208729182. If the user enters a number with fewer or more than 10 digits, store the number as 0000000000.

Add a get method for the phone number field. The get method returns the phone number as a String constructed as follows: parentheses surround a three-digit area code, followed by a space, followed by the three-digit phone exchange, followed by a hyphen, followed by the last four digits of the phone number.

Save the file as Event.java.

**We just compile the file – we do not run it at this point.**

Create an EventDemo application that does the following:

Prompts the user for event numbers and numbers of guests and constructs three Event objects.

Prompts the user for and retrieves a contact phone number for each of the Event objects.

Displays the event number and contact phone number for each Event. Save the file as EventDemo.java

**Ch. 5 - Done**

Yummy Catering provides meals for parties and special events. In Chapter 4, you created an Event class for the company. Now, make the following changes to the class:

❯ Currently, the class contains a constant for the price per guest ($35) that is used for every guest. Replace that constant field with two constant fields—a lower price per guest that is $32 and a higher price per guest that is $35.

❯ Add a new method named isLargeEvent() that returns true if the number of guests is 50 or greater and otherwise returns false.

❯ Modify the method that sets the number of guests so that a large Event (more than 50 guests) uses the lower price per guest to set the field that holds the price per guest and to calculate the total Event price. A small Event uses the higher price. Save the file as Event.java.

a program that demonstrates using the Event class as follows:

❯ Instantiate three Event objects, and prompt the user for values for each object.

❯ Display the details for all three objects. ❯ Create a method that accepts two Event objects and returns the larger one based on the number of guests. (If the Events have the same number of guests, you can return either object.)

❯ Call this method three times—once with each pair of instantiated Events—and display the event number and number of guests for each argument as well as the event number and number of guests for the larger Event. Save the file as EventDemo.java

**To run this program the user is prompted to enter event information eight times. After that is done a menu appears to allow the user to sort and display the info by Event Number or Number of Guests or Event Type or quit.**

**Skills used –**

**Create a class   
Create instance methods in a class   
Declare objects and use their methods   
Compare classes to data types   
Create and use constructors   
Use the this reference   
Use static fields**

**Ch. 6 - Done**

Yummy Catering provides meals for parties and special events. In previous chapters, you developed a class named Event that holds catering event information. Now create an EventDemo application to do the following:

❯ Create three Event objects.

❯ Continually prompt the user for the number of guests for each Event until the value falls between 5 and 100 inclusive.

❯ Display the details for each Event object. ❯ For the Event object with the fewest number of guests, create a loop that displays Please come to my event! as many times as there are guests for the Event.Save the file as EventDemo.java.

**To run this program the user is prompted to enter event information three times. After that is done a message appears on the console listing the events details and prints please come to my event several times.**

**Skills used -   
Create while loops  
Use shortcut arithmetic operators   
Create for loops  
Create do...while loops   
Nest loops   
Improve loop performance**

**Ch. 7 - Done**

Yummy Catering provides meals for parties and special events. In previous chapters, you have developed an Event class that holds catering event information. Now modify the Event class as follows:

❯ Modify the method that sets the event number in the Event class so that if the argument passed to the method is not a four-character String that starts with a letter followed by three digits, then the event number is forced to A000. If the initial letter in the event number is not uppercase, force it to be so.

❯ Add a contact phone number field to the Event class.

❯ Add a set method for the contact phone number field in the Event class. Whether the user enters all digits or any combination of digits, spaces, dashes, dots, or parentheses for a phone number, store it as all digits. (Assume a phone number is 10 digits and does not require a leading digit that might be needed when dialing.) For example, if the user enters (920) 872-9182, store the phone number as 9208729182. If the user enters a number with fewer or more than 10 digits, store the number as 0000000000.

❯ Add a get method for the phone number field. The get method returns the phone number as a String constructed as follows: parentheses surround a three-digit area code, followed by a space, followed by the three-digit phone exchange, followed by a hyphen, followed by the last four digits of the phone number.

Save the file as Event.java.

**We just compile the file – we do not run it at this point.**

Create an EventDemo application that does the following:

❯ Prompts the user for event numbers and numbers of guests and constructs three Event objects.

❯ Prompts the user for and retrieves a contact phone number for each of the Event objects.

❯ Displays the event number and contact phone number for each Event. Save the file as EventDemo.java.

**Now the EventDemo application prompts the user for event numbers, numbers of guests, and contact phone numbers for three Event objects and displays the event number and contact phone number for each Event.**

**Skills used -   
Identify string data problems   
Use Character class methods   
Declare and compare String objects   
Use a variety of String methods   
Use the StringBuilder and StringBuffer classes**

**Ch. 8 - Done**

Now modify the Event class to include an integer field that holds an event type. Add a final String array that holds names of the types of events that Yummy caters—wedding, baptism, birthday, corporate, and other. Include get and set methods for the integer event type field. If the argument passed to the method that sets the event type is larger than the size of the array of String event types, then set the integer to the element number occupied by other. Include a get method that returns an event’s String event type based on the numeric event type. Save the file as **Event.java**.

**Now, the Event class includes the event type field and the necessary get and set methods to manage it. If the argument passed to the set method is larger than the size of the event type names array, it defaults to "other."**

Create an EventDemo class that uses an array of eight Event objects. Get data for each of the objects, including an integer for the event type. Prompt the user to choose an option to sort Events in ascending order by event number, number of guests, or event type. Display the sorted list of Events, including Event number, number of guests, price per guest, total price, phone number, Event type number, and Event type String. Continue to prompt the user for sorting options and display the requested lists until the user enters a sentinel value. Save the file as **EventDemo.java**.

**Now, the EventDemo class creates an array of eight Event objects, allows the user to choose sorting options, and displays the sorted lists based on the chosen option. The user can exit the program by selecting option 4.**

**Skills used -   
Declare an array   
Initialize an array   
Use variable subscripts with an array   
Declare and use arrays of objects   
Search an array and use parallel arrays   
Pass arrays to and return arrays from methods   
Sort array elements   
Use two-dimensional and other multidimensional arrays   
Use the Arrays class   
Create enumerations**

**Ch. 9 - Done**

In Chapter 8, you created an Event class for Yummy Catering. Now extend the class to create a DinnerEvent class. In the extended class, include four new integer fields that represent numeric choices for an entrée, two side dishes, and a dessert for each DinnerEvent object. Also include three final arrays that contain String menu options for entrées, side dishes, and desserts, and store at least three choices in each array. Create a DinnerEvent constructor that requires arguments for an event number and number of guests, and integer menu choices for one entrée, two side dishes, and one dessert. Pass the first two parameters to the Event constructor, and assign the last four parameters to the appropriate local fields. Also include a getMenu() method that builds and returns a String including the Strings for the four menu choices. Save the file as **DinnerEvent.java**.  
In Chapter 8, you also created an EventDemo program for Yummy Catering. The program uses an array of Event objects and allows the user to sort Events in ascending order by event number, number of guests, or event type. Now modify the program to use an array of four DinnerEvent objects. Prompt the user for all values for each object, and then allow the user to continually sort the DinnerEvent descriptions by event number, number of guests, or event type. Save the file as **DinnerEventDemo.java.**

**Skills used -   
Override superclass methods   
Call constructors during inheritance   
Access superclass methods   
Employ information hiding   
Describe which methods you cannot override   
Create and use abstract classes   
Use dynamic method binding   
Create arrays of subclass objects   
Use the Object class and its methods   
Create and use interfaces   
Use records, anonymous inner classes, and lambda expressions**