Rochester Institute of Technology
National Technical Institute for the Deaf
Information and Computing Studies Department

Name: Edward Riley

NACA.161 Programming Fundamentals II

Practice Exercise #29 – Modularization

Objective

To make the classes that get user input and access other objects more manageable through the use of modularization.

Getting Started

Download the Table.java file from the conference.

Order Class

1) Create a class named **Modular** with a main method.

2) Create a method named **askForMoreTables** that returns a boolean. Inside this method, ask the user if they want to order a new table and get the input from the user. If the user enters 'n' return false and if they enter 'y' return true. This will be used to decide if the user wants to order another table.

3) Add exception handling around the user input (Scanner class). Make sure to catch the exact exception that the class you are using throws. If an exception is thrown. If an exception is thrown, return false. This will keep the program running.

   What exception did you catch?

   invalid options

4) The "trick" to making methods work inside a class that contains the main method is to create an object of the class and call the methods of this object. So create an object of the ~~Order~~ Modular class inside the main method.

5) Inside the main method, create a loop that will keep running while the call to the **askForMoreTables method** returns a true value. Make sure you use the order object that you created to call this method.

   What was the code you used to call the method?

   Order. Aske for MoreTables (

6) Compile and run the program to make sure everything works before continuing.

7) Add another method named **changeColor** that displays the color choices and the numbers to enter for each color, prompts the user to enter a color for the table, and gets the user input. Compile before continuing.

8) Catch the exception that the user input can throw. Display an error message and exit the program if an exception is thrown.

) Add another method named **changeNumLegs** that displays the number of legs choices, the numbers to enter for each leg option, prompts the user to enter the number of legs for the table, and gets the user input. Compile before continuing.

0)Catch the exception that the user input can throw. Display an error message and exit the program if an exception is thrown.

ethods should also set the attribute by calling the mutator. the mutator of the same object. Where [illegible]

6) Compile and run the program to make sure everything works before continuing.

7) Add another method named **changeColor** that displays the color choices and the numbers to enter for each color, prompts the user to enter a color for the table, and gets the user input. Compile before continuing.

8) Catch the exception that the user input can throw. Display an error message and exit the program if an exception is thrown.

9) Add another method named **changeNumLegs** that displays the number of legs choices, the numbers to enter for each leg option, prompts the user to enter the number of legs for the table, and gets the user input. Compile before continuing.

10) Catch the exception that the user input can throw. Display an error message and exit the program if an exception is thrown.

11) The 2 previous methods should also set the attribute by calling the mutator. However, both methods need to call the mutator of the same object. Where can we define an object so it is accessible inside 2 different methods?

before____________________

So define a Table object outside all of the methods and compile the code.

12) Inside the **changeColor** method add code that continues to call the mutator until the user enters a proper value. Compile the code before continuing.

13) Inside the **changeNumLegs** method add code that continues to call the mutator until the user enters a proper value. Compile the code before continuing.

14) Create another method named **getAttributeOption**. Make this method display a menu that lists the available attributes they can set, and if they want to add the order. Get a number from the user and return the number they entered.

15) In the above method, catch the exception that the user input can throw. Display an error message and exit the program if an exception is thrown.

16) Create a method named **displayAllTables** that displays the attribute values for every object in the collection

Now you have all the methods you need to create the code in the main method using the methods you have created. You need to follow the following **rules** when you perform the remaining steps:

- No methods can be defined as static (the main method is the only one that needs to be defined as static)
- When calling the methods you created, you need to use the object you created not just call the methods directly (bad things will happen if you do)
- Compile and run the code after every step

For steps 17 to 19, add the following code inside the loop you created in the main method. At some point you will need to add code that is not explicitly mentioned.

As you work to combine all the steps in the following steps,

17)Create the ArrayList and Table objects.

Since you need to access the ArrayList and Table objects in multiple methods, where do you need to declare the ArrayList and the Table object?

before main

Also, since the main method will be accessing these 2 objects and the main method is declared as static, what access modifier do you need to add to the declaration of these objects?

public Static

18)Display all the options to the user

19)Get user input from the user and catch the exception

20)Do the following while they don't ask to add the order to the collection:
- Run the proper method for the option they choose
- When they ask to add the object, add it to the collection and get out of this loop

21)At the end of the main method, display all the attributes for each object by calling the method you created

22)Make sure that you get the proper attributes displaying for each object you created.

Signoff

When you complete all of the steps successfully and answer all of the questions, contact your instructor to check if your application(s) executes correctly and to review your code. We will initial the line below.

______Successful execution of code

If you do not finish the program during the class period, contact your instructor to check to review your code and initial below.

______Code not completed during lab time

You may then submit your work at the start of next class. You may not use the work period of the next class to complete this assignment. If you do not have a signature, then you cannot receive any points for this assignment.