For this test, you are permitted to use material that you have brought into the test. You are NOT permitted to use work from others or communicate with others in any form. Violation of this will result in an automatic grade of 0 and an academic misconduct being filed.

Your code must align with Java coding best practices. Please read the reverse rubric at the end of this test to ensure you are not violating any of the rules.

The test will require you to create 2 classes. The first class is a “Component” class that represents components in a computer such as a keyboard, memory, monitor, etc... The second class is a “ShoppingCart” class that can hold components. When building the classes, be sure to add USEFUL messages to exceptions that are thrown. A useful message does not just say “error”, it defines what a valid argument is.

A “Main” class and method is provided to test your code and ensure it aligns with the expected output (see page 3 of the test for the expected output)

Throughout the test, you MUST submit your code to GitHub. I have included screen shots on pages 5-7 to help you remember how to use GitHub in IntelliJ and how to submit your repository to D2L.

# Class Diagram

# 

# Create Project in IntelliJ and private repo in GitHub (1 mark)

* Create a project in Intellij using JDK 17 or 21 with the name LHTest1Student1234567. Replace 1234567 with your Lakehead student number.



* Select VCS -> share project on GitHub and create a private repository. (see Helpful hint 1 on page 5)
* Go into the GitHub repository you just created, go to settings -> manage access -> invite a collaborator (green button) and select Rattangeo (see Helpful hint 2 on page 5)

# Component Class (15 marks)

The Component class represents a computer component and must store the component name, manufacturer and price. The methods and validation requirements are noted below:

* 1. The ***constructor*** must accept the component name, manufacturer and price (in that order). All arguments should be validated based on the rules in the set methods below. ***Push to GitHub with the commit message “Question 2.1 complete”.*** (3 marks)
  2. ***setName()*** – this method will receive a String as an argument, validate that it is at least 3 characters long and set the name instance variable. If it is not at least 3 characters long, throw an IllegalArgumentException with a useful message. ***Push to GitHub with the commit message “Question 2.2 complete”*** (3 marks)
  3. ***setPrice()*** – this method will receive a double as an argument. The method should validate that the argument is in the range of 0-10,000 (inclusive) and set the price instance variable. If it is outside the range, an IllegalArgumentException with a useful message should be thrown. ***Push to GitHub with the commit message “Question 2.3 complete”*** (2 marks)
  4. ***setManufacturer()*** – this method will receive a String as an argument. The method should validate that the argument is in the list of "3Com","Acer","Arctic","AMD", "Asus","Apple Inc.","Bose","Cooler Master", "Hitachi", "Intel", "Logitech", "Marvell", "Nvidia", "Qualcomm", "Samsung","Tyan","Fujitsu","MSI","Seagate","Toshiba", "Western Digital", "XTREEM". If the validation passes, set the **category** instance variable. If the validation fails, throw an IllegalArgumentException with a useful message. ***Push to GitHub with the commit message “Question 2.4 complete”*** (3 marks
  5. ***toString() –*** this method should return a String in the format of “[name]-[manufacturer], price: $[price]” where the square brackets indicate instance variables. An example of the toString() method running is: “speakers-Bose, price: $36.99”. Make sure that the price shows exactly 2 decimal places. ***Push to GitHub with the commit message “Question 2.5 complete”*** (2 marks)

# ShoppingCart Class (12 marks)

The ShoppingCart class must store a collection of Component objects in an ArrayList.

* 1. The ***constructor*** should not accept any arguments, but should initialize the ArrayList instance variable to hold Component objects. ***Push to GitHub with the commit message “Question 3.1 complete”.*** (2 marks)
  2. ***addItem()*** – this method should accept a Component object as an argument. The Component should be added to the ArrayList of Component objects. No validation is required. (1 mark)
  3. ***getTotalPrice()*** – this method should sum up the cost of all Component objects in the cart and return it as a double. (2 marks)
  4. ***getComponentNames() –*** this method should return a String object listing each of the Component items separated by a comma. For example, it would return the String “motherboard, processor, case, fan, video card, keyboard, mouse, speakers, NIC, RAM, sound card”. Note that the end of the String does not contain a comma after the last component. (3 marks)
  5. ***toString() –*** This method should return a String object in the format of “The cart has [x] components with a total price of $[XX.YY]”. In this case [x] is the number of Components in the cart and [XX.YY] is the total cost of all purchases. (1 mark)

# Example of the code running with the Main class provided

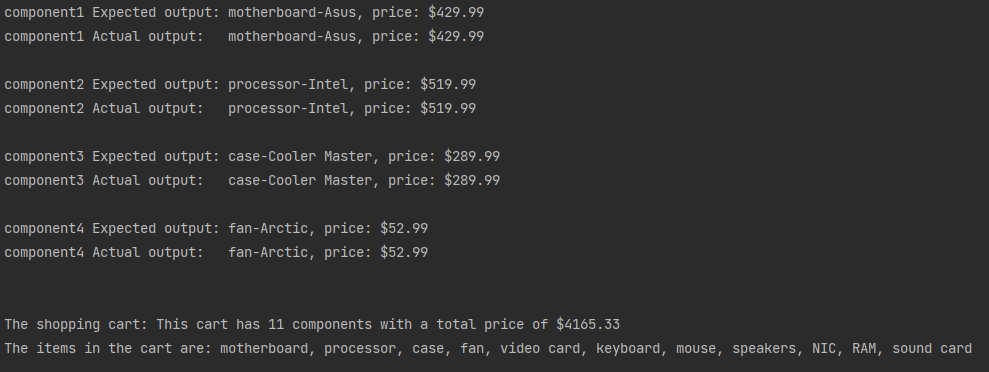
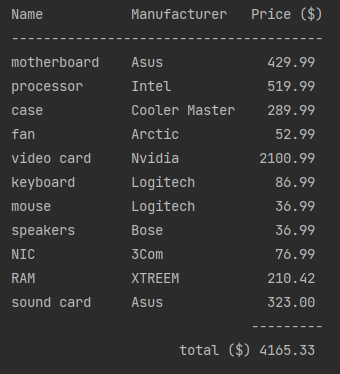
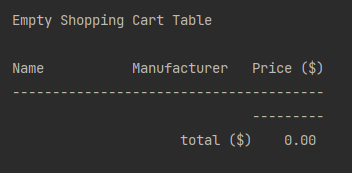


Figure 1 Program running

# Bonus (2 marks)

Create a method in the ShoppingCart class that returns a String with all the cart items in a table. It must work if there are 0 or more items in the ShoppingCart. (1 mark if the information is there, but not lined up. 2 marks if it matches the images below)

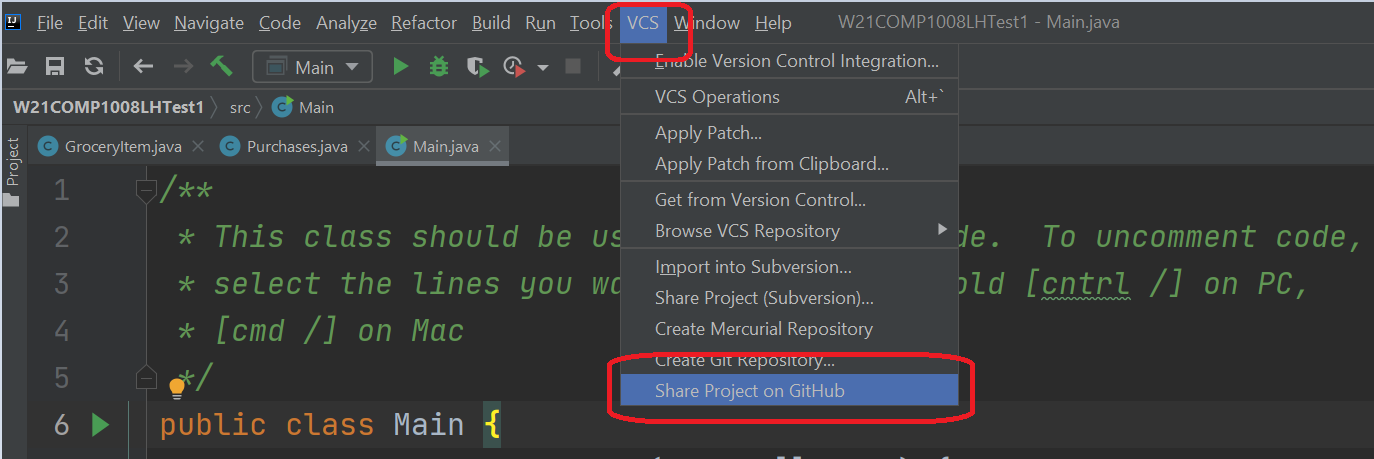
# Reverse Rubric

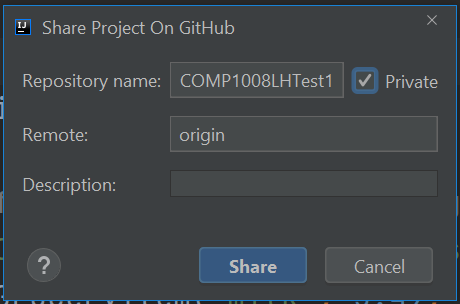
***This section is designed to ensure students adhere to coding best practices. Marks will be removed for code that matches any of the items below:***

* **1 mark will be removed for each line of code I need to change to make the code compile.**
* **1 mark will be removed if there are any variables or methods that do not start with a lower case letter and follow proper naming conventions**

# Helpful Hints

## Initial upload to GitHub





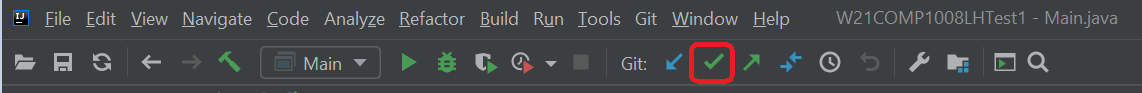
## Adding a Collaborator on a GitHub Repo

Inside your repository, select settings -> manage access -> invite a collaborator (green button) and enter Rattangeo

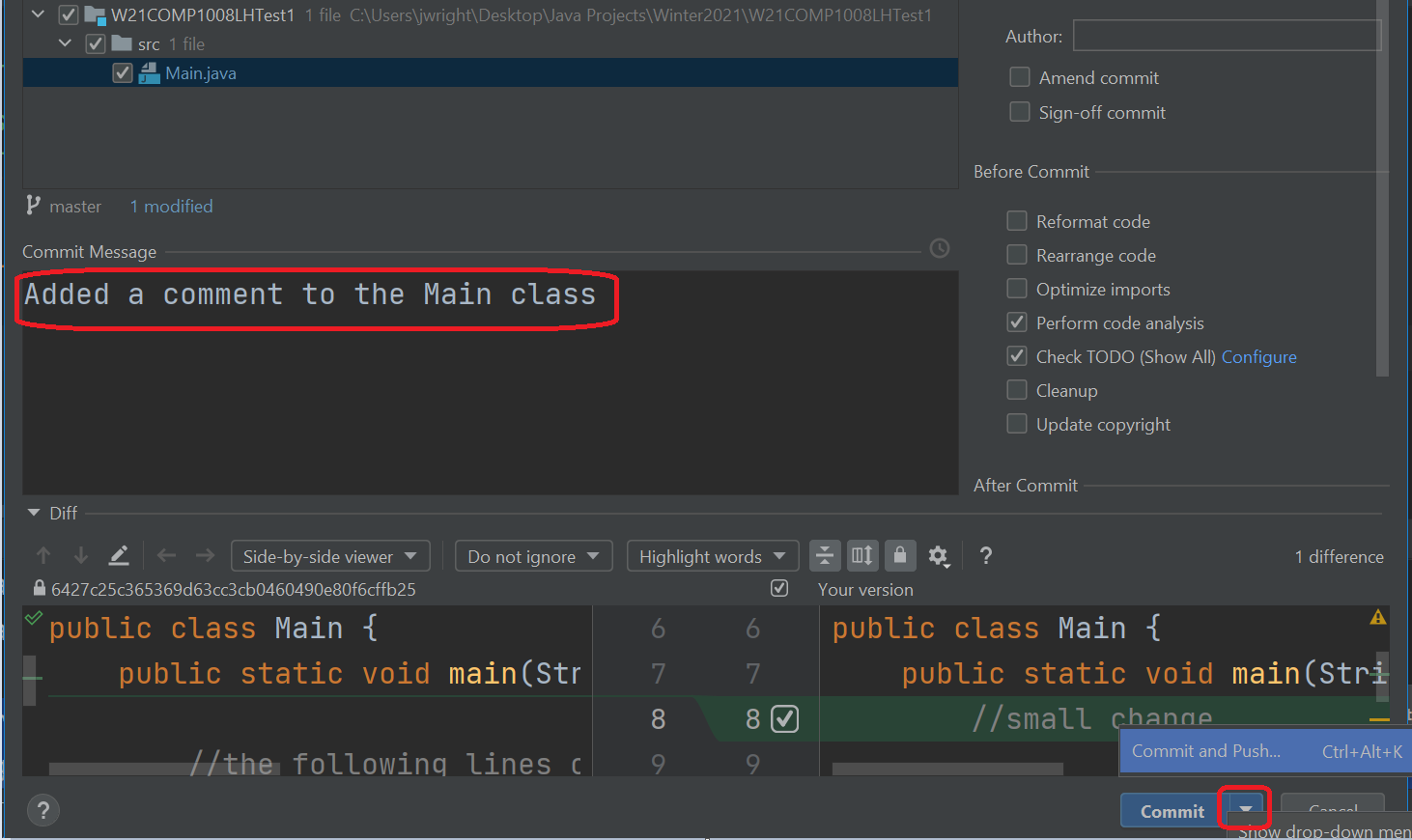
## 

## Commit’s on GitHub

Once you have created a private repository on GitHub, you can commit your code, by selecting the green check box in the Git menu of Intellij



* When the dialog comes up, add a comment as per the test instructions (circled in red).
* When you commit, be sure to select the down arrow and choose “Commit and Push”. If you just select “commit” it will only push to your local repository, not GitHub





## Submitting your repo into D2L

Inside your GitHub private repository for the test, select the green code button, then the clipboard icon. This will copy the URL to your repository into the clip board. In D2L, paste this in as text and you are done!

