|  |  |
| --- | --- |
| **Overview** |  |
| The below document compares requirements from both the assignment text and the assignment PDF against the operations of the code I wrote.  First, I walk through testing the individual classes to document that all requested methods are correct and errors are thrown when necessary. Next, I walk through testing the application as a whole by recreating example orders from both the text and PDF.  This time I tested the code via a Groovy console. After successful operations, Groovy will either repeat the last method executed or call toString() on the last object instantiated.  The below code covers all requirements/errors from the assignment. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | | **Requirement** | **Evidence** |
| Item | | Item(name, price)  Constructor that takes a name and a price as arguments. The name will be a String and the price will be a double. Should throw an IllegalArgumentException is price is negative. |  |
| Item | Item(name, price, bulk quantity, bulk cost)  Constructor that takes a name and a single-item price and a bulk quantity and a bulk price as arguments. The name will be a String and the quantity will be an integer and the prices will be doubles. Should throw an IllegalArgumentException if any number is negative. | |  |

|  |  |  |
| --- | --- | --- |
| Item | priceFor(quantity): Returns the price for a given quantity of the item (taking into account bulk price, if applicable). Quantity will be an integer. Should throw an IllegalArgumentException if quantity is negative.  **Non-bulk item test case** |  |

|  |  |  |
| --- | --- | --- |
| Item | priceFor(quantity): Returns the price for a given quantity of the item (taking into account bulk price, if applicable). Quantity will be an integer. Should throw an IllegalArgumentException if quantity is negative.  **Bulk item test case** |  |

|  |  |  |
| --- | --- | --- |
| Item | priceFor(quantity): Returns the price for a given quantity of the item (taking into account bulk price, if applicable). Quantity will be an integer. Should throw an IllegalArgumentException if quantity is negative.  **Negative quantity for both non-bulk and bulk item.** |  |
| Item | toString(): Returns a String representation of this item: name followed by a comma and space followed by price. If this has a bulk price, then you should append an extra space and a parenthesized description of the bulk pricing that has the bulk quantity, the word “for” and the bulk price. |  |

|  |  |  |
| --- | --- | --- |
| Catalog | Catalog(name): Constructor that takes the name of this catalog as a parameter. The name will be a String. |  |

|  |  |  |
| --- | --- | --- |
| Catalog | add(item): Adds an Item at the end of this list.  size(): Returns the number of items in this list.  get(index): Returns the Item with the given index (0-based).  getName(): Returns the name of this catalog. |  |

|  |  |  |
| --- | --- | --- |
| ItemOrder | ItemOrder(item, quantity): Constructor that creates an item order for the given item and given quantity. The quantity will be an integer.  getPrice(): Returns the cost for this item order.  getItem(): Returns a reference to the item in this order. |  |

|  |  |  |
| --- | --- | --- |
| ShoppingCart | ShoppingCart() Constructor that creates an empty list of item orders.  add(ItemOrder) Adds an item order to the list, replacing any previous order for this item with the new order. The parameter will be of type ItemOrder.  setDiscount(value)  Sets whether or not this order gets a discount (true means there is a discount, false means no discount).  getTotal() Returns the total cost of the shopping cart. |  |

|  |  |  |
| --- | --- | --- |
| Shopping Cart | “Most of these methods are fairly simple to write, but notice that when you add an ItemOrder to a ShoppingCart, you have to deal with replacing any old order for the item. A user at one time might request 3 of some item and later change the request to 5 of that item. The order for 5 replaces the order for 3. The user isn’t requesting 8 of the item in making such a change. The add method might be passed an item order with a quantity of 0. This should behave just like the others, replacing any current order for this item or being added to the order list.” |  |

|  |  |  |
| --- | --- | --- |
| Overall application | For example, the user is ordering 12 buttons that cost $0.99 each but can be bought in bulk 10 for $5.00. The first 10 are sold at that bulk price ($5.00) and the two extras are charged at the single item price ($0.99 each) for a total of $6.98. |  |

|  |  |  |
| --- | --- | --- |
| Overall application | Screenshot from PDF, no discount: | Recreated in my app, total amount of $491.20 ties: |

|  |  |  |
| --- | --- | --- |
| Overall application | Screenshot from PDF with discount: | Recreated in my app, total amount of $442.08 ties: |

|  |  |  |
| --- | --- | --- |
| Overall application | From assignment text:  One silly putty cost $3.95 and 11 are $23.94 |  |

|  |  |  |
| --- | --- | --- |
| Overall application | From assignment text:  One silly putty cost $3.95 and 11 are $23.94, plus one of everything else on the list totals $552.87 |  |

|  |  |  |
| --- | --- | --- |
| Overall application | From assignment text:  One silly putty cost $3.95 and 11 are $23.94, plus one of everything else on the list totals $552.87, but $497.58 if I check the discount, and $479.59 if I remove the Barbie. |  |