|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case #** | **Purpose of the test case** | **Input Data** | **Expected Output** |
| 1 | Test the **add()** method in the ShoppingBag class. If the size of the bag is greater than or equal to its current capacity, the method calls grow(), which increases the length of the array by 5 and adds 5 to capacity. In the Shopping class, the item name is printed out after calling the add() method in ShoppingBag. | “Spider\_Eye”, “0.01” “false”  “Golden\_Apple”, “49.99”, “true”  “Rotten\_Flesh”, “0.03”, “false”  “Enchanted\_Golden\_Apple”, “100.01”, “true” | “·Spider\_Eye: $.01 : tax free  ·Golden\_Apple: $49.99 : is taxable  ·Rotten\_Flesh: $.03 : tax free  ·Enchanted\_Golden\_Apple: $100.01 : is taxable |
| 2 | Test the **remove()** method in the ShoppingBag class. This method calls find() to get the index of the item to be removed, moves the last item in the array into the index of the removed item, and sets the reference of the last array index to null. If an item that is not in the bag is requested to be removed, the method in Shopping prints the appropriate error message.   * Case 2.1: requested item is in bag, item is removed, last item in array is moved to removed index, and last item index is null * Case 2.2: requested item is not found in bag, error message is printed | * Case 2.1:   “Spider\_Eye”  “Rotten\_Flesh”   * Case 2.2:   “Golden\_carrot”  “Apple” | * Case 2.1:   Spider\_Eye removed.  Rotten\_Flesh removed.  Printing contents:  ·Enchanted\_Golden\_Apple: $100.01 : is taxable  ·Golden\_Apple: $49.99 : is taxable   * Case 2.2:   “Unable to remove Golden\_Carrot  Unable to remove Poisonous\_Potato”  Printing contents:  ·Enchanted\_Golden\_Apple: $100.01 : is taxable  ·Golden\_Apple: $49.99 : is taxable |
| 3 | Test the **grow()** method in the ShoppingBag class. This method is called when there is no more space in the bag for another item, and it increases the array size by 5, and increments the capacity variable by 5 as well. The new bag size becomes the newly incremented size of the current size plus 5. Then, the old and new capacities will be printed. | “Spider\_Eye”, “0.01”, “false”  “Golden\_Apple”, “49.99”, “true”  “Rotten\_Flesh”, “0.03”, “false”  “Enchanted\_Golden\_Apple”, “100.01”, “true” | “Old capacity: 5”  “New capacity: 10” |
| 4 | Test the **salesTax()** method in the ShoppingBag class. This method adds on a 6.625% sales tax to any taxable items in the bag. It iterates through the bag and calls the getTaxable() method on each iteration; if this method returns true, then the item’s price is added to an accumulating sum, which after the for loop is done iterating through the bag, the accumulated sum is multiplied by the sales tax of 6.625% and this number is returned rounded to 2 decimal places. | “Enchanted\_Golden\_Apple”, 100.01, false  “Golden\_Apple”, 49.99, false  “Spider\_Eye” .01, true  “Golden\_Apple”, 49.99, true  “Rotten\_Flesh”, .03, true  “Enchanted\_Golden\_Apple”, 100.01, false | “Sales tax: 19.88” |