Institute of Computer Science

CMSC 22: Object-Oriented Programming

CHECKPOINT JOURNAL 04

Instructions: Accomplish* this journal every checkpoint so we can monitor your progress and improve everyone's learning experience. Answer and submit as Google Doc (PDF is only for those who have limitations in working online.)

**Accomplish this when you are done (or almost done) with the lecture and lab requirements for the week.*

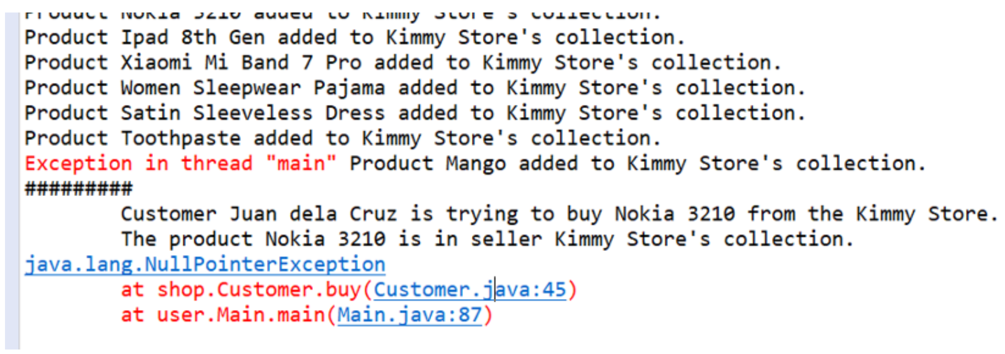
Name:	**Gabinete, Keith Ginoel S.**	Checkpoint Topic/s:	**Polymorphism/Concurrency**
Student Number:	**2020-03670**	Date:	**April 29, 2023**

1. What problem/confusion did you encounter about the lesson/s or requirement/s?

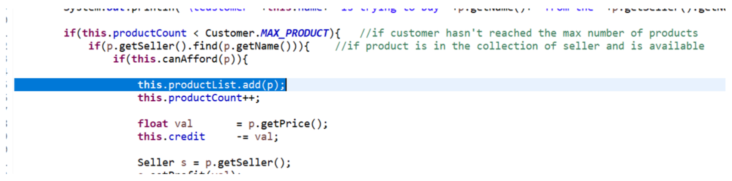
   Explain the specifics of the problem (Minimum of **2** sentences).

   Note that even If no problem was encountered in understanding the lesson, it's certain that at least a minor issue will be faced while doing the requirements, especially the ***lab exercise****. Discuss at least one challenge faced.*

I encountered a NullPointerException error when I try to run my code for exer06. The error was specifically being thrown at line 45 of my code in the Customer class. I couldn't figure out the main reason it was occurring. I know it has something to do with the productList variable (an array using the ArrayList class from the java.util package) but still, I can't figure out how to fix it.



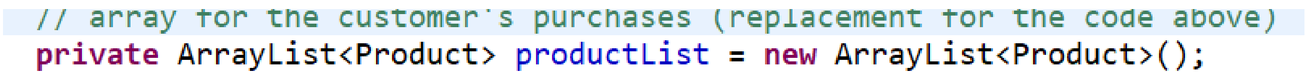
```
Product Ipad 8th Gen added to Kimmy Store's collection.
Product Xiaomi Mi Band 7 Pro added to Kimmy Store's collection.
Product Women Sleepwear Pajama added to Kimmy Store's collection.
Product Satin Sleeveless Dress added to Kimmy Store's collection.
Product Toothpaste added to Kimmy Store's collection.
Exception in thread "main" Product Mango added to Kimmy Store's collection.
#########
        Customer Juan dela Cruz is trying to buy Nokia 3210 from the Kimmy Store.
        The product Nokia 3210 is in seller Kimmy Store's collection.
java.lang.NullPointerException
        at shop.Customer.buy(Customer.java:45)
        at user.Main.main(Main.java:87)
```



```
if(this.productCount < Customer.MAX_PRODUCT){   //if customer hasn't reached the max number of products
    if(p.getSeller().find(p.getName())){    //if product is in the collection of seller and is available
        if(this.canAfford(p)){

            this.productList.add(p);
            this.productCount++;

            float val       = p.getPrice();
            this.credit     -= val;

            Seller s = p.getSeller();
```

2. How did you solve it and what became your solution? **Explain the specific solution found.** Include **references** and **code snippets** when applicable (Minimum of **3** sentences).

This stack overflow thread right here < https://stackoverflow.com/questions/218384/what-is-a-nullpointerexception-and-how-do-i-fix-it> helped me better understand why the NullPointerException is being thrown at on my program. It was stated on the thread that NullPointerException error occurs basically when your reference variable doesn't point to anything. In my case, I found out that my productList array list isn't pointing to anything as I forgot to initialize it (as I'm still not used to utilizing the ArrayList Class). A simple fix to it is just to basically instantiate a new ArrayList of type Product in the declaration of the productList variable.



```
// array for the customer's purchases (replacement for the code above)
private ArrayList<Product> productList = new ArrayList<Product>();
```

3. Choose at least one of the things discussed that you understood the most. Imagine explaining it to a classmate. **Explain it in your own words** (Minimum of **4** sentences - can be 2 sentences for each of the week's topics).

We learned in week 06 that Java couldn't do multiple inheritances, as one Java class can extend only one superclass. Fortunately, Java allows the implementation of one or more interfaces even in one Java class. These so-called java interfaces act like a contract between a Java class and the outside world. Just like a real contract here in the outside world, all conditions/behaviors that were stated in a Java interface should be fulfilled by its contractor / the class implementing it.

We also learned in week 06 that Polymorphism is one of the most beneficial features of an Object-Oriented programming language like Java. Polymorphism allows us to take advantage of an object's flexibility trait as it takes many forms by implementing multiple interfaces or even by extending a superclass. Just like a kid for example. A kid can be a student, an athlete, a celebrity, a sibling or daughter/son . One kid can have different behaviors in different situations . This is what polymorphism is all about.

In week 07, we also learned that multithreading is another useful feature provided by Java. With multithreading, we are able to execute two or more parts of our program concurrently to maximize the utilization of our CPU (especially when its other parts are in idle state while the rest are busy executing tasks). Thread in java can be created either by extending the Thread class or implementing the Runnable Interface.

Please communicate urgent concerns to your instructor via Discord. Do not write them down here so that they can be addressed immediately. (Ex: Installation problems, health concerns).