Chris Budway
Prof. Lane
COM 210 S01
2/12/23

Intro Lab Report

Introduction: The purpose of this lab was to experiment with the implementation of java concepts in code. We had four tasks which included the development of an algorithm that would take user input and then output the imputed items in the order they were received and an average of the prices of said inputs. From this point, we had to further develop it to satisfy the growing complexity of the situation.

Method: Starting with the beginning of the algorithm, I opted to use a system that used no loops and followed a simple linear execution.

```java
import java.util.Scanner;
public class COM210LabQ1
{

    Run | Debug
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);
        double x;
        double x1;
        double x2;
        double avg;
        String s;
        String s1;
        String s2;
        System.out.println(x: "List the three items you are purchasing.");
        s = scan.nextLine();
        s1 = scan.nextLine();
        s2 = scan.nextLine();
        System.out.println(x: "What are the prices of these three items?");
        x = scan.nextDouble();
        x1 = scan.nextDouble();
        x2 = scan.nextDouble();
        avg = (x + x1 + x2)/3;
        System.out.println("Item One: " + s + " " + x + ", " + "Item Two: " + s1 + " " + x1 + ", " + "Item Three: " + s2 + " " + x2+ ", " + "The average price of these three products

        //System.out.println(x + " " + x1 + " " + x2);
        //System.out.println(s + " " + s1 + " " + s2);

    }
}
```

It takes in three user inputs for item names and then prices. It takes this data and averages it, then it outputs the information in order. The second iteration uses a conditional statement but utilizes no loops as well. The newly added code is a flag system to detect whether or not “peas” was entered as an input, and if it was, the algorithm would then output the average.

```
if(s.equalsIgnoreCase(peas))
{
    flag = true;
}
else if(s1.equalsIgnoreCase(peas))
{
    flag = true;
}
else if(s2.equalsIgnoreCase(peas))
{
    flag = true;
}
else
{
    flag = false;
}
if(flag)
{
    System.out.println("The average price of these items is: "+ avg);
}
else
{
    System.out.println(x: "No average output.");
}
```

The final iteration of the algorithm was an overhaul of the previous versions. I renamed variables to make them more readable and included loops to reduce clutter.

```
Run | Debug
public static void main(String[] args)
{
Scanner scan = new Scanner(System.in);
int input;

System.out.println(x: "Enter the number of items that you want to process.");
input = scan.nextInt();

String itemNames[] = new String[input];
double itemPrice[] = new double[input];
```

This segment is an initializer for the two arrays that take in the user inputs as well as the size of the class to store these inputs. The next segment is responsible for iterating through the arrays storing the data from the user, the last seen loop then prints these in reverse order due to the for loop iteration from the last known input to the first.

```
for(int i=0;i<input;i++)
{
    System.out.println("Enter Item "+(i+1)+" details");
    System.out.print(s: "Enter Item Name:");
    itemNames[i] = scan.next();
    System.out.print(s: "Enter Item Price:");
    itemPrice[i] = scan.nextDouble();
}

System.out.println(x: "Item details in reverse order:");
System.out.println(x: "Item Name \t Item Price");
for(int i=input-1;i>=0;i--)
{
    System.out.println(itemNames[i]+"\t\t"+itemPrice[i]);
}
```

Throughout the development of the code, I was constantly commenting sections in and out to test if I could remove any unnecessary code or simply to check if specific inputs were working as intended. A big problem for me was running checks for the flagging system to detect whether or not “peas” was being input or not.

Analysis: I do not fully understand how to derive Time Complexity, because of this I will try my best to describe the time complexity of the algorithm. The first algorithm is fully constant but poorly optimized so it would have a lot of unnecessary steps within the algorithm. It is a similar case for the second algorithm with the exception of the conditionals. The final algorithm has much fewer constants due to the introduction of the for loops to make the algorithm run smoother.

Conclusion: Despite not having actual findings on my time complexities for the algorithm, however with what I understand of the concept time complexity is an important aspect of testing algorithms. What would make you ultimately choose between two separate algorithms? How good each is at doing its programmed job and lastly, the time complexity associated with its processing of the algorithm. This lab was a nice introduction back to Java, especially since I have not touched java for nearly a year now. It served as a nice review of the old topics from 152.