**CSCI 470**

**Assignment#3**

**Due 03/26/2023 by 11:59 pm**

**100 Points**

**Exercise 1. (20 Points)**

Write a [program](https://pearson.turingscraft.com/codelab/jsp/core_dhtml.jsp?) with a loop that lets the user enter a series of [integers](https://pearson.turingscraft.com/codelab/jsp/core_dhtml.jsp?). The user should   
enter -99 to signal the end of the series. After all the numbers have been entered, the  
[program](https://pearson.turingscraft.com/codelab/jsp/core_dhtml.jsp?) should display the largest and smallest numbers entered. The [value](https://pearson.turingscraft.com/codelab/jsp/core_dhtml.jsp?) -99 is not to  
be counted as part of the series.

### SAMPLE RUN:

Enter an integer (-99 to stop):1

Enter an integer (-99 to stop):1

Enter an integer (-99 to stop):2

Enter an integer (-99 to stop):3

Enter an integer (-99 to stop):5

Enter an integer (-99 to stop):8

Enter an integer (-99 to stop):13

Enter an integer (-99 to stop):21

Enter an integer (-99 to stop):34

Enter an integer (-99 to stop):55

Enter an integer (-99 to stop):-99

The smallest number was: 1

The largest number was: 55

**RUBRIC:**

Uses loop with sentinel value of -99 (10Points)

Displays largest and smalles correctly. (10 Points)

**Exercise 2. (20 Points)**

Create a class named Apartment that holds an apartment number, number of bedrooms, number of baths, and rent amount. Create a constructor that accepts values for each data field. Also create a get method for each field. Write an application that creates at least five Apartment objects. Then prompt a user to enter a minimum number of bedrooms required, a minimum number of baths required, and a maximum rent the user is willing to pay. Display data for all the Apartment objects that meet the user’s criteria or an appropriate message if no such apartments are available. Save the files as **Apartment.java** and **TestApartments.java**.

Use the following data for objects:

Apartment apt1 = new Apartment(101, 2, 1, 725);

         Apartment apt2 = new Apartment(102, 2, 1.5, 775);

         Apartment apt3 = new Apartment(103, 3, 2, 870);

        Apartment apt4 = new Apartment(104, 3, 2.5, 960);

         Apartment apt5 = new Apartment(105, 3, 3, 1100);

**RUBRIC:**

Correctly defined getAptNumber in Apartment class (2 points)

Correctly defined getBedrooms in Apartment class (2 points)

Correctly defined getBaths in Apartment class (2 points)

Correctly defined getRent in Apartment class (2 points)

Correctly defined constructor for Apartment class (2 points)

Correctly creating objects and testing. (10 points)

**Exercise 3. (20 Points)**

Write an application that displays the factorial for every integer value from 1 to 10. A factorial of a number is the product of that number multiplied by each positive integer lower than it. For example, 4 factorial is 4 \* 3 \* 2 \* 1, or 24. Save the file as **Factorials.java**

**RUBRIC:**

Calculate and display the factorial of each number between **1** and **10 (20 points)**

**Exercise 4. (20 Points)**

Write an application that allows a user to enter any number of student quiz scores until the user enters 99. If the score entered is less than 0 or more than 10, display an appropriate message and do not use the score. After all the scores have been entered, display the number of scores entered, the highest score, the lowest score, and the arithmetic average. Save the file as **QuizScoreStatistics.java**.

**RUBRIC:**

The QuizScoreStatistics program outputs expected results for valid quiz scores. (10 Points)

The QuizScoreStatistics program handles invalid quiz scores. (10 Points)

**Exercise 5. (20 Points)**

1. Create a class named Purchase. Each Purchase contains an invoice number, amount of sale, and amount of sales tax. Include set methods for the invoice number and sale amount. Within the set() method for the sale amount, calculate the sales tax as 5% of the sale amount. Also include a display method that displays a purchase’s details. Save the file as **Purchase.java**.
2. Create an application that declares a Purchase object and prompts the user for purchase details. When you prompt for an invoice number, do not let the user proceed until a number between 1,000 and 8,000 has been entered. When you prompt for a sale amount, do not proceed until the user has entered a nonnegative value. After a valid Purchase object has been created, display the object’s invoice number, sale amount, and sales tax. Save the file as **CreatePurchase.java**.

**RUBRIC:**

Correctly define Purchase class. (3 points)

Correctly define setInvoiceNumber and getInvoiceNumber in the Purchase class. (3 Points)

Correctly define setSaleAmount and getSaleAmount in the Purchase class. (3 Points)

Correctly create the CreatePurchase class. (5 points)

Display the purchase details, including the invoice number, sale amount, and tax. ( 6 Points)