WESTERN UNIVERSITY - CANADA
FACULTY OF ENGINEERING
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

**SE2203b –Software Design**

**Assignment 1**

**Due Date: February 7th , 2020**

This assignment is to be done individually. Cheating is prohibited in this assignment, that is not to show your code to any other student, do not look at any other student's code, and do not put your code in any public domain. However, getting help is easy and available, just ask the instructor or the TAs. At the same time, you are encouraged to discuss your problems (if any) with other students in general terms but the actual program design and code must be your own.

Please note that the consequences of plagiarism are much, much worse than getting a low mark for that particular item of work. The very best case is that you get a zero for the whole thing. Keep in mind that turning in your own work would have at least gotten you a few marks, better than a zero.

# 1 Goals

- Write specification for methods within a Java interface.
- Choose appropriate classes and methods during the design of a program, including classes that might be written already.

# 2 Assignment works

An odometer records a car's mileage. It contains a number of wheels that turn as the car travels. Each wheel shows a digit from 0 to 9. The rightmost wheel turns the fastest and increases by 1 for every mile traveled. Once a wheel reaches 9, it rolls over to 0 on the next mile and increases by 1 the value on the wheel to its left.

You can generalize the behavior of such wheels by giving them symbols other than the digits from 0 to 9. Examples of such wheel counters include

- A binary odometer whose wheels each show either 0 or 1
- A desktop date display with three wheels, one each for year, month, and day
- A dice roll's display whose wheels each show the spots for a single die

## 2.1 Question 1 – Define generic type Java Interface (assesses the ET1 CEAB indicator)

1. Using IntelliJ IDE create a new Java project and name it your*UwoIdAssignment1*. For example, if your email address is aouda@uwo.ca, then name the project as *aoudaAssignment1*.

2. Write a generic type Java interface named Rollable to be used for creating any class that represents **a wheel**. Use the UML class diagram shown in Figure 1 to write your Interface.
3. This interface describe four methods; Reset() to reset the counter to the minimum value, increase() to increase the value on the counter, lastRolledOver() to test whether the last increase caused a rollover, and getValue() to return the value of the counter.
4. The interface Rollable will be implemented later by any class that represents a wheel. The generic data type T represents the type of the wheel's current value. For example, a value's type might be String or Integer.
5. Write a generic type Java interface named ConsoleDisplayInterface for **a general wheel counter** as shown in figure 2.
6. Do not forget to write inline comments and descriptive comments at the top of each method.

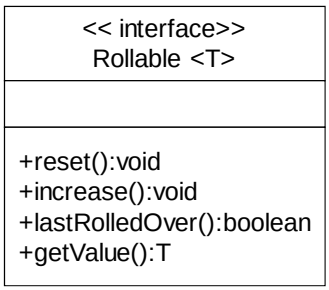


*Figure 1*

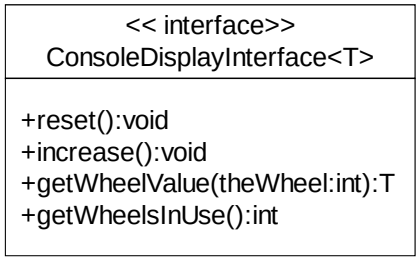


*Figure 2*

**Notice** If you have some questions or if you need to make sure you are doing the right answers, your TAs are willing to help, they are available during the designated Labs hours.

## 2.2 Question 2 – Implement Java Interface (assesses the ET2 CEAB indicator)

1. Write a new class to keep track of a non-negative count (i.e., 0,1,2,…) and name it Counter as shown in Figure 3. This class has increase() method to increase the count by one, decrease() method to decrease the count by one, getCount() method to return the value of the count, and isZero() method to check whether the count is zero.
2. Do not forget to add toString() method to return a string representation of the count.
3. Write a new class named Wheel. This class records a non-negative count value that is between a minimum and a maximum value. Increasing the count beyond its maximum will roll it over to the minimum value. Decreasing the count below its minimum will roll it over to the maximum value.
4. The Wheel class needs to extend the Counter class and implement the Rollable interface, as shown in Figure 4.
5. Write a new class named ConsoleDisplay that implements the ConsoleDisplayInterface, see Figure 5 above. This class maintains a collection of wheels of type Rollable. Note that, the constructors of ConsoleDisplay should be able to instantiate up to 3 wheels, (we can do this by having 3 different constructors).

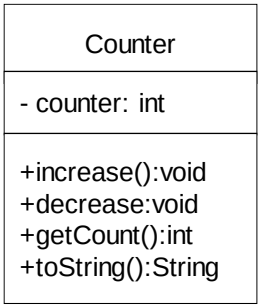


*Figure 3*

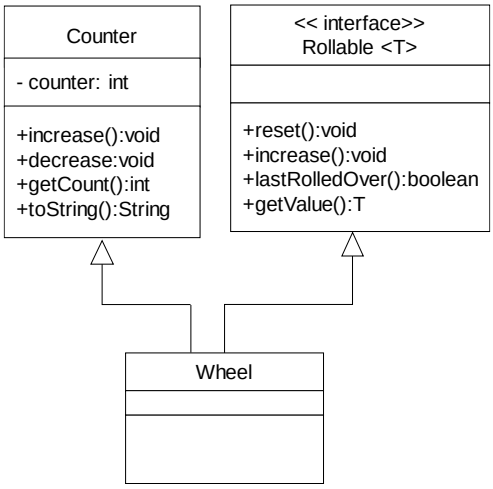


*Figure 4*

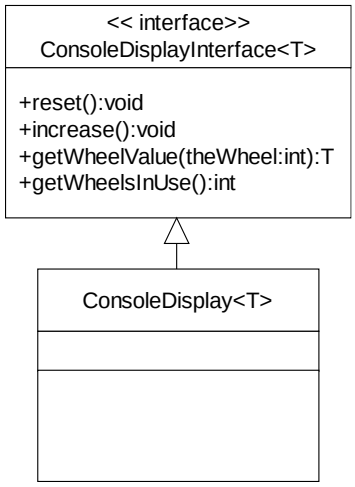


*Figure 5*

6. Do not forget to add toString() method to return a string representation of the value of all wheels.
7. Now, add the given Java class "DigitalClock.java" to your project, then compile and run. Without any changes in this class, you should have an output like the one shown in the given video DigitalClock.mp4.
8. Add inline comments and descriptive comments at the top of each method.

## 2.3 Question 3 – More hand-in experience on classes and objects (assesses the ET2 CEAB indicator)

1. The Feeling Wheels, shown in Figure 6, are designed to aid people in learning to recognize and communicate about their feelings. It consists of an inner wheel with 5 sectors and two outer concentric wheels. The sectors are each labeled with the name of a primary feeling. The outer wheels contain names of secondary feelings related to the primary ones. These wheels have proven useful in assisting clients to learn how to identify, to express, to generate, and to change feelings.
2. The text version of these wheels is given in three txt files; InnerWheel.txt, MiddleWheel.txt, and OuterWheel.txt. The OuterWheel.txt file starts with the word "romantic" and ends with the word "raptured". The words are followed by 1 or 0, 1 means that the middle wheel needs to rollover.

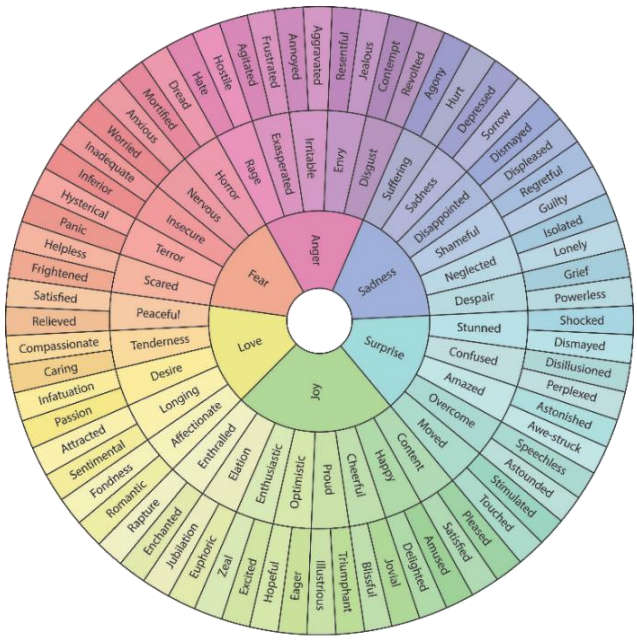


*Figure 6*

3. Similarly, the MiddleWheel.txt file starts with the word "affectionate" and ends with the word "enthralled". These words are followed by 1 or 0, 1 means that the inner wheel needs to rollover.
4. Using your interface classes Rollable<T> develop a new class named "FeelingsWheel" to simulate the operation of these three wheels and the relationship among their sectors.
5. The class diagram shown in Figure 7, will help you to construct your program quickly. Feel free to add any variable you think it is useful.
6. The constructor of the class FeelingsWheel get the name of the txt file, so that getWheelSize and getFeelingsNames method use it upload the file content into an array.
7. Do not forget to write inline comments and descriptive comments at the top of each method.
8. The increase() method needs to be implemented such that the order and the relationship among the three wheels are maintained. For example, when the inner wheel shows the word "Fear" the middle wheel will rollover 5 times and for each of these 5 words, the outer wheel will rollover 2 times. The following represents this relationship for the sector "Fear"

   Fear -> Scared -> Frightened
   Fear -> Scared -> Helpless
   Fear -> Terrified -> Panicked
   Fear -> Terrified -> Hysterical
   Fear -> Insecure -> Inferior
   Fear -> Insecure -> Inadequate
   Fear -> Nervous -> Worried
   Fear -> Nervous -> Anxious
   Fear -> Horrified -> Mortified
   Fear -> Horrified -> Dreadful

9. When finished, add the given Java class "FeelingAndEmotionClinic.java" to your project, then compile and run. Without any changes in this class, you should have an output like the one shown in the given video FeelingAndEmotionClinic.mp4.

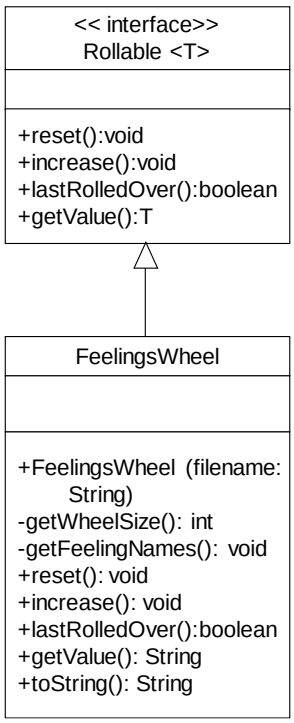


*Figure 7*

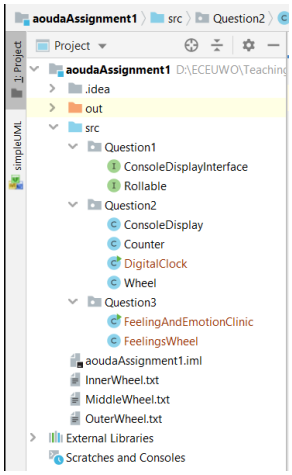


*Figure 8*

## 3 Hand In

You should use IntelliJ IDEA IDE to create, edit, and run the required Java programs.

- The file structure of your project should be created as shown in Figure 8.
- From the IntelliJ main menu, select File → Export to Zip File…, the Save as pop-up window appears, click OK to save your project as yourUwoIdAssignment1.zip
- Submit this zip file using OWL assignment link at the due date mentioned above.