University of Windsor SCHOOL OF COMPUTER SCIENCE	**COMP-2120, Fall 2024** **Object Oriented Programming Using Java** **LAB-2** **TOTAL MARKS: 10**

LAND ACKNOWLEDGEMENT
The School of Computer Science at the University of Windsor *sits on the Traditional Territory of the Three Fires Confederacy of First Nations*. We acknowledge that this is the beginning of our journey to understanding the Significance *of the history of the Peoples of the Ojibway, the Odawa, and the Pottawatomie.*

SUBMISSION DEADLINE: OCT-4, 2024

GENERAL INFORMATION

Welcome to the Lab Session of COMP-2120 where you will learn object-oriented programming (OOP) using Java. Please, arrive early to the lab and get settled to start working on the lab. Arriving 20 minutes after the start of the lab will be considered late and can affect your lab participation point. There are eight lab exercises you need to complete throughout the course. Each week on Thursday morning we will publish the lab exercise. You need to upload your code by Thursday, 10 AM in the next week.

LAB GRADING AND ASSESSMENT

You need to explain your code to TAs in the lab to receive the marks. YOU MUST demonstrate your results and explain code to the TAs to receive the marks. Thus, attending the lab sessions is mandatory. Note that your TAs are here in the Lab to help you learn. Feel free to ask questions and talk to them. Remember, the total Labs is worth 16% of your final grade.

Lab-2 Tasks

1. Create a class Time that has three private int instance variables: **hour**, **minute** **and** **second**. **(1 Mark)**
2. The Time class has two constructors. The first one is a zero-argument constructor that initialize all instance variables to zero. The second constructor takes three int parameters as input and use them to set the instance variables. **(1 Mark)**
3. Create accessor and mutator methods for all three instance variables. Follow appropriate naming scheme. **(1 Mark)**
4. Write a public Method **toUniversalString** that takes no arguments and returns a String in universal-time format, consisting of two digits each for the hour, minute and second. For example, if the time were 1:30:07 PM, the method would return 13:30:07. **(1 Mark)**

5. Now, write another public method **toString** that takes no arguments and returns a String in standard-time format, consisting of the hour, minute and second values separated by colons and followed by AM or PM (e.g., 1:27:06 PM). Like method toUniversalString, method toString uses static String method **format** to format the minute and second as two-digit values, with leading zeros if necessary. **(1 Mark)** See here to learn more: https://stackoverflow.com/questions/6431933/how-to-format-strings-in-java
   and here: https://docs.oracle.com/javase/8/docs/api/?java/lang/String.html

6. Now create another class **TimeTest** to test the **Time** class. For the purpose of testing, create two objects of the **Time** class using zero-argument and three argument constructors, respectively. Now, call the **toString** and **toUniversalString** methods and verify that they are working by printing their return values on the screen. You need to do that for both objects. **(1 Mark)**
7. Modify class **Time** to include a **tick** method that increases the time stored in a Time object by one second. Provide a method **incrementMinute** that increase minute by one. Provide another method **incrementHour** that increase hour by one. Be sure to test the following cases and make sure your program is working correctly:
   a) incrementing into the next minute,
   b) incrementing into the next hour and
   c) incrementing into the next day (i.e., 11:59:59 PM to 12:00:00 AM).
   (2 Marks)
8. Modify class **Time** to include a method **isEqual** that takes another **Time** object as parameter and return a Boolean value depending on whether they are equal or not. Make sure you compare the **Time** objects you created earlier in the **TimeTest** class by calling the **isEqual** method and print the comparison result on the screen. **(1 Mark)**
9. Your program should check whether the input parameters are within the range (For example, minute cannot be less than zero or more than 60). If the values are not in the correct range, print an appropriate error message and quit the program by calling **System.exit(0)** method. See here: https://stackoverflow.com/questions/30898773/java-system-exit-how-do-i-use-it
   to learn more about the exit method. **(1 Mark)**

WHAT DO YOU NEED TO DO?

1. Complete the program.
2. Attend the lab. Explain the code to TAs.
3. TAs can modify input to your program.