

**SWE4201 Introduction to software development**

**Object interaction worksheet**

**Main Concepts**

|  |  |  |  |
| --- | --- | --- | --- |
| • | Abstraction | • | Object diagrams |
| • | Modularisation | • | Method calls |
| • | Constructors | • | Debuggers |

* Object creation

Note: some of these exercises require you to modify source code in Eclipse, remember that you must compile the code before you can use it.

1. PRESENTATION (Slides 1 - 12)
   * Download from Moodle the Java project “***ClockDisplay”*** into the “week 4” subfolder of the “OOPworkspace” folder on your M drive (or C drive of your own machine).
   * Unzip the “***ClockDisplay***” folder.
   * Start Eclipse and import java project “***ClockDisplay”.***
   * This project consists of three classes, the **NumberDisplay** class, the **ClockDisplay** class and the **ClockDisplayTester** class.

**Exercise 1: Using the default constructor to create a ClockDisplay object and test its methods**

In the **ClockDisplatTester** class, clock1 object has been created using the default constructor. Run that code and observe what is displayed when using the default constructor.

Answer:

**The output is**

**00:00**

**00:01**

**00:02**

**Exercise 2: Using the parameter constructor to create ClockDisplay object and test its Methods**

In the **ClockDisplatTester** class clock2 object has been created using the parameter constructor. Run that code and observe what is displayed when using the parameter constructor. Also observe the rollover when the minute reaches 59.

**Answer: The output is:**

**23:59**

**00:00**

**00:01**

**00:02**

**Exercise 3:**

In the **ClockDisplayTester** class Create a third object and call it “clock3” using the parameter constructor and set the time to 15:25 as in below:

ClockDisplay clock3 = **new** ClockDisplay(15,25);

A. Razak Page 1 of 3 12/02/2023 21:57



In the table below make a note of the current state of the **hours** and **minutes** objects (by using the printTime() method). Then call setTime with the values indicated and make a note of the new time.

|  |  |
| --- | --- |
| **hours** | **minutes** |

Starting time

**clock3.setTime(-5, 35) 15 35**

**Clock3.setTime(11, 78) 11 35**

What conclusions can we reach from this experiment?

**Answer: The hours in the first clock get deducted by 5 which shouldn’t be happening because the hours or the minutes never go back on a clock**.

Modify the code so that it will display appropriate error messages when an illegal value is entered.

**Exercise 4:**

Rewrite the increment method without the modulo operator using an if statement. Which solution is better?

The following exercise is a real challenge and if you manage to complete it then you are doing really well. If not don’t be disheartened, you can come back to this at a later time and have another go.

1. PRESENTATION (Slides 12 - 35)

**Exercise 5**

In this activity you are going to modify the ***ClockDisplay*** program to work with hours, minutes, and seconds. Before you start, look at the source code and identify which parts of the class declaration you think need to be changed and why. When you think you know what changes are needed then have a go. Test your program, for instance did it roll over from 08:59:59 to 09:00:00?

**Exercise 6: Challenge exercise**

Change the clock from a 24-hour clock to a 12-hour clock.

In a 12-hour clock, the hours after midnight and afternoon are NOT shown as 00:30, but 12:30. Therefore the minutes display shows values from 0 to 59, while the hours shoes values from 1 to 12, and you have to add am and pm.

**Hint:**

There are - at lease – two ways in which you can make a 12-hour clock. One possibility is to just store hour values from 1 to 12. On the other hand, you can simply leave the clock to work internally as a 24-hour clock but change the display string of the clock display to show 6:28 or 6:28pm when the internal value is 18:28.

**Implement both versions. Which option is easier? Which option is better?**

A. Razak Page 2 of 3 12/02/2023 21:57



**Exercise 7: using the Debugger.**

The ability to read and understand source code is one of the most essential skills for a software developer, and we need to apply it in every project we work on. However, sometimes it is beneficial to use additional tools in order to help us gain a deeper understanding about how a program executes. One tool we will look at is a debugger.

A debugger is a program that lets programmers execute an application one step at a time. It typically provides functions to stop and start a program at selected points in the source code, and to examine the values of variables. Eclipse has a built-in debugger which we are going to use.

1. Read and follow the tutorial on Eclipse debugger here: <https://www.vogella.com/tutorials/EclipseDebugging/article.html>
2. Watch the following videos on using Eclipse debuggers:

Lesson 1:

<https://www.youtube.com/watch?v=pYvYSOLPoPE&list=PLUGinutrSzN-ALgaL0mSyPDHi3VhUCcuV&index=2&t=0s>

Lesson 2:

<https://www.youtube.com/watch?v=rFuFJ41uha8&list=PLUGinutrSzN-ALgaL0mSyPDHi3VhUCcuV&index=3&t=0s>

**You must submit the following to the Moodle “Object interaction programs submission” link clearly identified with your name and student ID number:**

* **A zip folder containing your fully commented code for exercise 5.**
* **A zip folder containing your fully commented code for exercise 6.**

**The deadline for submission is 19/02/2022 @ 23.55 pm**

A. Razak Page 3 of 3 12/02/2023 21:57