

**SWE4201 Introduction to Software Development**

**Selection worksheet**

**Main Concepts**

* Fields
* Methods (mutators and accessors)
* Constructors
* Assignment and
* Parameters
* Conditional statements

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 - 9)

**Exercise 1: The better ticket machine**

* Download from Moodle the Java subfolder of the “**OOPworkspace”**

project “***BetterTcketMachine”*** into the “**week 3”** folder on your M drive (or C drive of your own machine).

* Unzip the “***BetterTcketMachine”*** folder.
* Start Eclipse and import java project “***BetterTcketMachine”.***
* There are two classes in **TicketMachineTester** class.differ from last week.

this project. The **TicketMachine** class and the Study the coding of the TicketMachine class and see how it

The TicketMachine tester class creates a TicketMachine object (with a ticket price of 200 Pounds). Use the **insertMoney** to answer the following questions.

1. What happens to the value in balance when we use the **insertMoney** methods with negative values?

**It prints a message saying that the user needs to give a positive value.**

1. What happens if we do not insert enough money (<200) before asking for a ticket? ***Note:*** *I have provided the coding for (a and b in the tester class)*

**The program prints the following:**

**# Ticket cost is: 200 Pounds.**

**# In order to buy a ticket, you must insert at least: 100 more pounds.**

1. What happens to **total** and **balance** when we enter exactly the ticket price (200) before asking for a ticket?

**It prints a message saying that the user has bought a ticket for 200.**

1. What happens to **total** and **balance** when we enter more than the ticket price (>200) before asking for a ticket?

**The balance is raised up to 300.**

How does this differ from the BasicTicketMachine of last week machine?

Compare the source code for both machines and make sure you understand why they work differently.

**The week 2 ticket machine prints the balance after the ticket is bought whereas the ticket machine of week 3 does not print the balance after the user has bought the ticket.**

A. Razak Page 1 of 4 05/02/2023 20:54



1. PRESENTATION (Slides 10 - end)

**Exercise 2: Local variables**

Add a new method called **emptyMachine**, that is designed to simulate someone emptying the machine. It should return both the value of total and reset it to zero.

Is this method an accessor or mutator or both?

**It is a mutator method because it sets a value and it’s an accessor at the same time because it returns a value. Also, a new print emptyMachine method needs to be created in order for the result to show.**

**Exercise 3: Local variables**

Rewrite the **printTicket** method so that it declares a local variable, **amountLeftToPay**. This should be intialised to contain the difference between **balance** and **price**. Rewrite the test condition to check the value of **amountLeftToPay**. If its value <= 0, a ticket should be printed, otherwise an error message should be printed stating the amount still required. Test your version to ensure that it behaves in exactly the same way as the original.



The following exercises all make use of the ***book-exercise*** project so open it now. Examine the source code for the **Book** class, you should find two fields and one constructor, you will be adding more methods and fields to this class. After each exercise you should compile and test your class by creating object of type Book in the BookTester class and call (invoke) the appropriate methods on the object. In the comments insert your name and date where indicated.

**Exercise 4:**

Add two methods **printAuthor** and **printTitle** to the Book class. These should print the **author** and **title** fields, respectively, to the terminal window.

**Exercise 5:**

Add a further field called **pages** to the **Book** class to store the number of pages. This should be of type **int**, and its initial value should be passed to the single constructor after the two other parameters already there. Include an accessor method called **getPages** for this field.

**Exercise 6:**

Add a method, **printDetails**, to the **Book** class. This should print details of the **author**, **title** and **pages** to the terminal window. You should decide on how the details are to be displayedbut you might use **printTicket** method as a guide.

**Exercise 7:**

Add a further field called **refNumber** of type **String.** This should be initialised to the zero-length string (“”) in the constructor by adding the line

refNumber = “”;

to the body of the constructor. There is no need to add another parameter to the constructor.

Add a mutator method called **setRefNumber** and an accessor method called **getRefNumber** to set and get this reference number.

A. Razak Page 2 of 4 05/02/2023 20:54



**Exercise 8:**

Modify the **printDetails** method to include printing the **refNumber** but if it has not been set, it should print **”???”** instead.

Hint: the **String** class has a method called length which returns the number of characters in the string (see “Example on the length() method of the String class” in Moodle) so the statement

refNumber.length()

will return zero if it has not been set and >0 if it has been set.

**Exercise 9:**

Modify the mutator method for **refNumber** (i.e., the **setRefNumber** method) so that it is only set if the parameter is a string of more than 3 characters. If it is less than 3 then print an error message.

**Exercise 10:**

Add another integer field called **borrowed** which keeps track of how many times a book has been borrowed, it should be initialised to zero in the constructor. Add a mutator method called **setBorrowed** which increments the field (i.e., add one to the field) every time it is called. Add anaccessor method called **getBorrowed** which returns the value sored in the **borrowed** field.

Finally modify **printDetails** method to include the value of this field along with some explanatory text.



**In the following exercises you will create a new project and class from scratch.**

**Exercise 11:**

In Eclipse create a new project called ***heater-exercise***. In this project, create a new class called **Heater**

1. Modify the class so that it has 4 integer fields called **min**, **max**, **temperature** and **increment**.
2. Modify the constructor so it has 2 parameters called **pMin** and **pMax**. Inside the constructor use these parameters to set the values of **min** and **max**.
3. In the constructor set **increment** to **5** and set the field **temperature** to (**min + 10)**.
4. Create a mutator method called **warmer** which increments the **temperature** by the amount stored in **increment**. Use a conditional statement so the temperature cannot go above **max**.
5. Create a mutator method called **cooler** which decrements the **temperature** by the amount stored in **increment**. Use a conditional statement so the temperature cannot go below **min**.
6. Add a mutator method for the field **increment**. Use a conditional statement that only changes the **increment** if the actual parameter value >0 (greater than zero)
7. Add a **printDetails** method that will display the object’s state in the terminal window.
8. Add an accessor method for each field.

Now test your program – by creating a HeaterTester class and testing all methods. Does it behave as we expect?

A. Razak Page 3 of 4 05/02/2023 20:54



**Exercise 12:**

In Eclipse create a new project and call it ***person-exercise***. In this project create a new class called **Person**

The class should have

* two fields **name** and **age** (you decide on appropriate data types),
* a constructor that has two parameters to initialise the fields,
* accessor and mutator methods for the two fields and
* a **printDetails** method to show the values stored in the fields.

Now test your program – by creating a PersonTester class and test all methods. Does it behave as we expect?

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

1. **A zip folder containing your code for all the exercises in the book-exercise project (exercise 4 to exercise 10)**
2. **A zip folder containing your code for exercise 11**
3. **A zip folder containing your code for exercise 12**

**The deadline for submission is 12/02/2023 @ 23.55 pm.**

A. Razak Page 4 of 4 05/02/2023 20:54