**KV4000 Week 2 Lab\_C. Exercises from the module textbook. Some of these exercises refer to the TicketMachine class.**

**Exercise 2.12** What do you think is the *type* of each of the following fields?

private int count;

private Student representative;

private Server host;

**Exercise 2.13** What are the *names* of the following fields?

private boolean alive;

private Person tutor;

private Game game;

**Exercise 2.16** Is it always necessary to have a semicolon at the end of a field declaration? Once again, experiment via the editor. The rule you will learn here is an important one, so be sure to remember it.

**Exercise 2.17** Write in full the declaration for a field of type int whose name is status.

**Exercise 2.21** Suppose that the class Pet has a field called name that is of type String. Write an assignment statement in the body of the following constructor so that the name field will be initialized with the value of the constructor’s parameter.

public Pet(String petsName)

{

...

}

**Exercise 2.29** Do the insertMoney and printTicket methods have return statements? Why do you think this might be? Do you notice anything about their headers that might suggest why they do not require return statements?

**Exercise 2.33** Complete the body of the following method, whose purpose is to add the value of its parameter to a field named score.

/\*\*

\* Increase score by the given number of points.

\*/

public void increase(int points)

{

...

}

**Exercise 2.35** Can you complete the following method, whose purpose is to subtract the value of its parameter from a field named price?

/\*\*

\* Reduce price by the given amount.

\*/

public void discount(int amount)

{

...

}

**Exercise 2.37** Add a method called prompt to the TicketMachine class. This should have a void return type and take no parameters. The body of the method should print something like:

Please insert the correct amount of money.

**Exercise 2.41** Add a showPrice method to the TicketMachine class. This should have a void return type and take no parameters. The body of the method should print something like:

The price of a ticket is *xyz* pence.

where xyz should be replaced by the value held in the price field when the method is called.

**Exercise 2.42** Create two ticket machines with differently priced tickets. Do calls to their showPrice methods show the same output, or different? How do you explain this effect?

*Before attempting these exercises, be sure that you have a good understanding of*

*how ticket machines behave, and how that behavior is implemented through the*

*fields, constructor, and methods of the class.*

**Exercise 2.43** Modify the constructor of TicketMachine so that it no longer has a parameter. Instead, the price of tickets should be fixed at 1000 pence. What effect does this have when you construct ticket machine objects within BlueJ?

**Exercise 2.44** Give the class two constructors. One should take a single parameter that specifies the price,and the other should take no parameter and set the price to be a default value of your choosing. Test your implementation by creating machines via the two different constructors.

**Exercise 2.45** Implement a method, ‘empty’, that simulates the effect of removing all money from the machine. This method should have a void return type, and its body should simply set the total field to zero.

Does this method need to take any parameters? Test your method by creating a machine, inserting some money, printing some tickets, checking the total, and then emptying the machine. Is this method a mutator or an accessor?