|  | Initials |
| --- | --- |
| **Exercise 2.1** Create a **TicketMachine** object on the object bench and take a look at  its methods. You should see the following: **getBalance**, **getPrice**, **insertMoney**, and **printTicket**. Try out the **getPrice** method. You should see a return value containing the price of the tickets that was set when this object was created. Use the **insertMoney** method to simulate inserting an amount of money into the machine. The machine stores as a balance the amount of money inserted. Use **getBalance** to check that the machine has kept an accurate record of the amount just inserted. You can insert several separate amounts of money into the machine, just like you might insert multiple coins or bills into a real machine. Try inserting the exact amount required for a ticket, and use **getBalance** to ensure that the balance is increased correctly. As this is a simple machine, a ticket will not be issued automatically, so once you have inserted enough money, call the **printTicket** method. A facsimile ticket should be printed in the BlueJ terminal window. |  |
| **Exercise 2.2** What value is returned if you get the machine’s balance after it has printed a ticket? |  |
| **Exercise 2.3** Experiment with inserting different amounts of money before printing tickets. Do you notice anything strange about the machine’s behavior? What happens if you insert too much money into the machine – do you receive any refund? What happens if you do not insert enough and then try to print a ticket? |  |
| **Exercise 2.4** Try to obtain a good understanding of a ticket machine’s behavior by interacting with it on the object bench before we start looking, in the next section, at how the **TicketMachine** class is implemented. |  |
| **Exercise 2.5** Create another ticket machine for tickets of a different price; remember that you have to supply this value when you create the machine object. Buy a ticket from that machine. Does the printed ticket look any different from those printed by the first machine? |  |
| **Exercise 2.6** Write out what you think the outer wrappers of the **Student** and **LabClass** classes might look like; do not worry about the inner part. |  |
| **Exercise 2.7** Does it matter whether we write  **public class TicketMachine**  or  **class public TicketMachine**  in the outer wrapper of a class? Edit the source of the **TicketMachine** class to make the change, and then close the editor window. Do you notice a change in the class diagram? What error message do you get when you now press the *Compile* button? Do you think this message clearly explains what is wrong?  Change the class back to how it was, and make sure that this clears the error when you compile it. |  |
| **Exercise 2.8** Check whether or not it is possible to leave out the word **public** from the outer wrapper of the **TicketMachine** class. |  |
| **Exercise 2.9** Put back the word **public**, and then check whether it is possible to leave out the word **class** by trying to compile again. Make sure that both words are put back as they were originally before continuing. |  |
| **Exercise 2.10** From your earlier experimentation with the ticket machine objects within BlueJ, you can probably remember the names of some of the methods–**printTicket**, for instance. Look at the class definition in Code 2.1 and use this knowledge, along with the additional information about ordering we have given you, to make a list of the names of the fields, constructors, and methods in the **TicketMachine** class. *Hint:* There is only one constructor in the 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 2.14** From what you know about the naming conventions for classes, which of the type names in Exercises 2.12 and 2.13 would you say are class names? |  |
| **Exercise 2.15** In the following field declaration from the **TicketMachine** class  **private int price;**  does it matter which order the three words appear in? Edit the **TicketMachine** class to try different orderings. After each change, close the editor. Does the appearance of the class diagram after each change give you a clue as to whether or not other orderings are possible? Check by pressing the *Compile* button to see if there is an error message.  Make sure that you reinstate the original version after your experiments! |  |
| **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**. |  |