LAB ASSIGNMENT 1

DUE TO NEXT LAB

1. Open the *naive-ticket-machine* project in chapter02 in BlueJ. Replace the constructor of the `TicketMachine` class with the following:

   ```
   public TicketMachine (int ticketCost)
   {
      int price = ticketCost;
      balance = 0;
      total = 0;
   }
   ```

   A. What is wrong with it?

2. Implement what we started in class: Add a `showPrice` method to the `TICKETMACHINE` class. This should have a void return type and take no parameters.

   A. The body of the method should print:
   the price of a ticket is xyz cents.
   Where xyz should be replaced by the value held in the field `price` when the method is called.

   B. Now 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?

3. Fix the TicketMachine using if / else

   A. Fix: No checks on the amounts entered.

      i. check if the inserted amount is not negative

      ii. only issue a ticket if enough money was inserted

   B. Fix: No checks for a sensible initialization.

      i. check for a sensible price given to the constructor

   C. Fix: No refunds.

      i. balance should not be set to 0 after ticket is printed

4. If the name of `getBalance` is changed to `getAmount`, does the return statement in the body of the method also need to be changed for the code to compile?
   A. Try it out within BlueJ.
   B. What does this tell you about the name of an accessor method and the name of the field associated with it?

5. Write an accessor method `getTotal` in the `TicketMachine` class. The new method should return the value of the field `total`.

6. Try removing the return statement from the body of `getPrice`.
   A. What error message do you see now when you try compiling the classes?

7. 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)
{
...
}
```

8. Add a method called prompt to the `TicketMachine` class. This should have a void return type and take no parameters.
   A. The body of the method should print the following single line of output: please insert the correct amount of money.

9. Add the possibility to count the number of tickets sold. Include a method for outputting how many tickets have been sold.

10. 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.

A. 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.

B. Is the `empty` method a mutator or an accessor?

11. Add on (for the bored): can you make the `better-ticket-machine` give proper change with a minimal amount of euro coins?