Introduction to Computer Science II

Tutorial 3

Objectives

Basic object creation (constructors and instance methods). Looking up Java's Random class.

Grading

Complete the quiz for Tutorial 3 in Brightspace for your grade in this tutorial.

Part One : Money [Constructors]

Modify the `Money` class that is provided. This is a simple class that stores money as dollars and cents. For example, $12.73 will be stored as 12 dollars and 73 cents. The cents value stored should never be greater than 99, so 3 dollars and 164 cents should be stored as 4 dollars and 64 cents.

The class has only one method, toString(), which returns a String representation of the money object. Your first task is to create FOUR constructors for the class as follows:

```
public Money(){...}
   // creates a Money object with zero money

public Money(int dollars, int cents){...}
   // creates a Money object with total value as specified by
   // the input values. Input values are assumed to satisfy
   // dollars >= 0 and cents >= 0.

public Money(int cents){...}
   // creates a Money object with value as specified by
   // the input cents. Input value is assumed to satisfy
   // cents >= 0

pubic Money(int[] coins){...}
   // creates a Money object with value as specified by the input array.
   // The array will have six (6) elements, corresponding to the coins
   // toonies, loonies, quarters, dimes, nickels, pennies (in that order)
   // ($2, $1, $0.25, $0.10, $0.05, $0.01)
```

In all the constructors, be sure that the internal state (dollars and cents) represents the total money and that cents is not greater than 99 after the constructors have terminated.

The `Money` class has a `toString()` method to help test/debug your code. It returns a String representation of the money object. Note that it relies on the state using the attributes as given. So, if you change the attributes your code might not compile.

You are not as asked to use information hiding (encapsulation) for this class. But you should be thinking about how you would do this. Also, you should be thinking about the possible consequences of not doing it.

Part Two : Money [Behaviour]

Add some behaviour to the Money class. Add the following THREE methods:

```
public Money add(int d, int c)
   // adds d dollars and c cents to the current money object
   // (like the constructors, the internal state of the object
   //  after this method ends will always have 0 <= cents <= 99)
   // pre-conditions: d >= 0 and c >= 0.
   // returns a reference to the current object

public boolean remove(int d, int c)
   // removes d dollars and c cents from the current money object
   // if possible and does not change the state if there is not
   // enough money to remove.
   // pre-conditions: d >= 0 and c >= 0.
   // returns true if d dollars and c cents was removed,
   // and returns false otherwise

public boolean transferFrom(int d, int c, Money otherMoney)
   // Transfers d dollars and c cents from otherMoney to this
   // Money object. If otherMoney does not have enough money,
   // nothing is transferred and the method returns false.
   // If otherMoney does have enough money, it is removed from
   // otherMoney and transferred (added) to this Money object.
```

Uncomment the testing code in your `main()` method to test your methods.

What other test cases should you be considering?

Part Three : Reading

Look at the documentation for the `java.util.Random` class and see what methods it provides.