Practice problem on defining and using Class types. Part 2.

In this problem, you will be adding code to class Time, developed in the previous practice, and simultaneously testing it using another class. Download Time.java and TestTime.java from the course webpage. The content of these files is also included in this handout for your reference.

Be sure to have your Time and UseTime classes placed in the same project. When running your program in Eclipse, pick UseTime as the main class.

Add the following methods to class Time:

1. A no-argument (0-argument constructor), which would set the calling object’s time to represent time 0:00.
2. A 2-argument (a.k.a. parameter) constructor, which accepts 2 input parameters representing an hour and a minute values. This constructor should call method setTime() to set the instance variable of the object.
3. Instance method toString() which returns a string that contains the description of the calling object in the “hh:mm” format, e.g. “7:15” or “18:00”.
4. Instance method isBefore() which can be used to compare two time objects. The method should accept a single parameter of type Time, and return true if the calling object represents an earlier time point than the parameter, and false otherwise.

As you are adding methods to class Time, simultaneously add code to the main method of class UseTime that will test the methods you have developed.

1. Create two objects of class Time using the constructors you have defined.
2. Print the objects.
3. Compare the two using instance method isBefore(), and print the earlier one.

```java
/* * A class that represents a time value, consisting of hour and minute, based on the 24-hour clock. */

public class Time {
    private int hour;
    private int minute;

    /* Method
       boolean setTime(int hour, int minute )
       * - sets the hour and minute value of the calling object to
       * the parameters, in case params hour and minute* have appropriate values.
       * Returns true, if values were appropriate, false, otherwise. */
    public boolean setTime(int h, int m) {
        boolean ok = true;
        // check if hour is appropriate
        if ( h < 0 || h > 23){
            System.out.println("Invalid value of hour " +h);
            ok = false;
        }
        else this.hour = h;

        // check if minute is appropriate
        if ( m < 0 || m > 59){
            System.out.println("Invalid value of minute " +m);
            ok = false;
        }
        else this.minute = m;

        return ok;
    }
}
```
/* Accessor methods */
public int getHour(){
    return this.hour;
}

public int getMinute() {
    return this.minute;
}

/*  public int toMinutes()
* returns the time of the calling object, represented in minutes only */
public int toMinutes(){
    // compute the calling object in minutes only
    int thisInMinutes = this.hour *60 + this.minute;
    return thisInMinutes;
}

/* public String getAMPMTime()
* Returns a string like "2:35 PM" or "7:15 AM" based on the calling object*/
public String getAMPMTime(){
    String ampm;
    DecimalFormat minFrm = new DecimalFormat("00");
    if (this.hour < 12 )
        ampm = this.hour + ":" + minFrm.format(this.minute) + " AM";
    else if (this.hour == 12)
        ampm = this.hour + ":" + minFrm.format(this.minute) + " PM";
    else // here this.hour > 12
        ampm = (this.hour-12) + ":" + minFrm.format(this.minute) + " PM";
    return ampm;
}

/* public Time timeAfter(int min)
* creates and returns a Time object, representing the calling object's time
* plus min minutes. */
public Time timeAfter(int min){
    int newMinutes = min + this.toMinutes();
    Time newTime = new Time();
    newTime.setTime(newMinutes/60%24, // %24 for wrapping hours in 0..23 range
                     newMinutes%60);
    return newTime;
}
public class UseTime {
    public static void main (String[] args){
    }
}