Lab04: Skills - Using a Debugger

***Getting Ready:*** Before going any further, you should:

# Setup your development environment.

1. Download the following files:

PhoneDriver.java

PhoneCard.java

to an appropriate directory/folder. (In most browsers/OSs, the easiest way to do this is by right-clicking/control-clicking on each [of the links above.)](https://users.cs.jmu.edu/bernstdh/web/common/labs/experimenting_unit-testing/using-objects/GiftCard.java)

1. If you don't already have one from earlier in the semester, create a project named eclipseskills.
2. Drag the file PhoneCard.java and PhoneDriver.java into the default package (using the "Copy files" option).
3. Open PhoneCard.java and PhoneDriver.java.

***Part 1.* Review*:*** This part of the lab will review a few topics related to object-oriented programming in Java.

* 1. In the main() method in the PhoneDriver class, what kind of objects are end, now, and start?

**They are all Date objects.**

* 1. In the main() method in the PhoneDriver class, what kind of object is card?

**Card is a PhoneCard object.**

* 1. Where is the code for the PhoneCard class?

**The code for the PhoneCard class is within PhoneCard.java**

* 1. Where is the code for the Date class?

**The code comes from java.util.Date**

* 1. Read the documentation for the Date (<https://docs.oracle.com/javase/7/docs/api/java/util/Date.html>). Make sure you find the documentation for the Date class that is in java.util. (There are several Date classes in the Java library.)
  2. When you construct a Date object using the default constructor (i.e., the constructor that has no parameters), what properties will it have?

**“Allocates a Date object and initializes it so that it represents the time at which it was allocated, measured to the nearest millisecond.”**

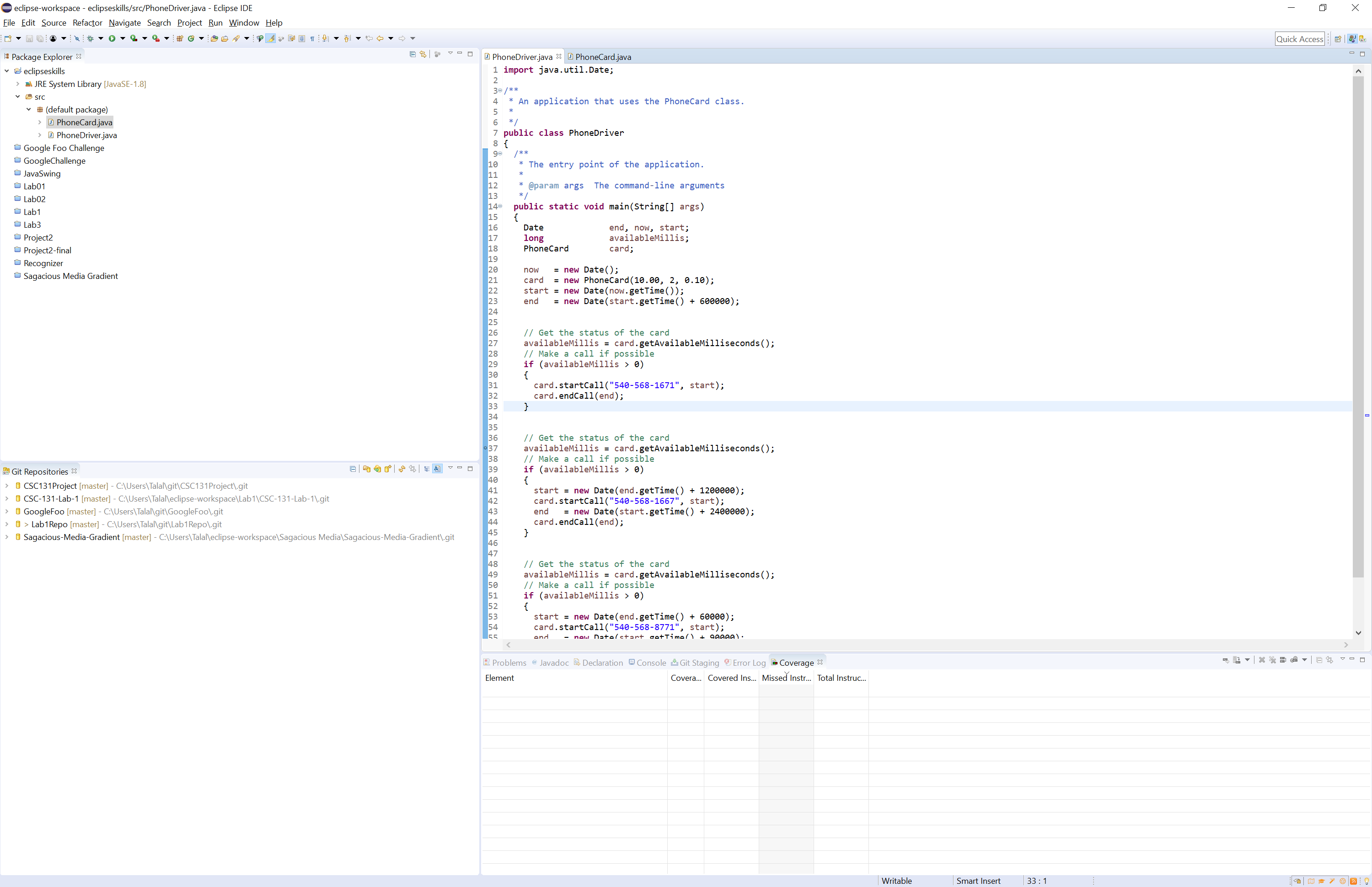
* 1. When you construct a Date object using the default constructor (i.e., the constructor that has no parameters), what properties will it have?

**Allocates a Date object and initializes it so that it represents the time at which it was allocated, measured to the nearest millisecond.**

***Part 2.* Setting a Breakpoint**: One of the nice things about running an application in a debugger is that you can stop the execution at one or more pre-defined locations (called breakpoints). This part of the lab will teach you how.

1. Click on the tab containing PhoneDriver.java to make sure that it has the focus.
2. Right-click in line 33 of PhoneDriver.java and pull down to Toggle Breakpoint.
3. What happened?

**A little blue circle shows up on line 37**



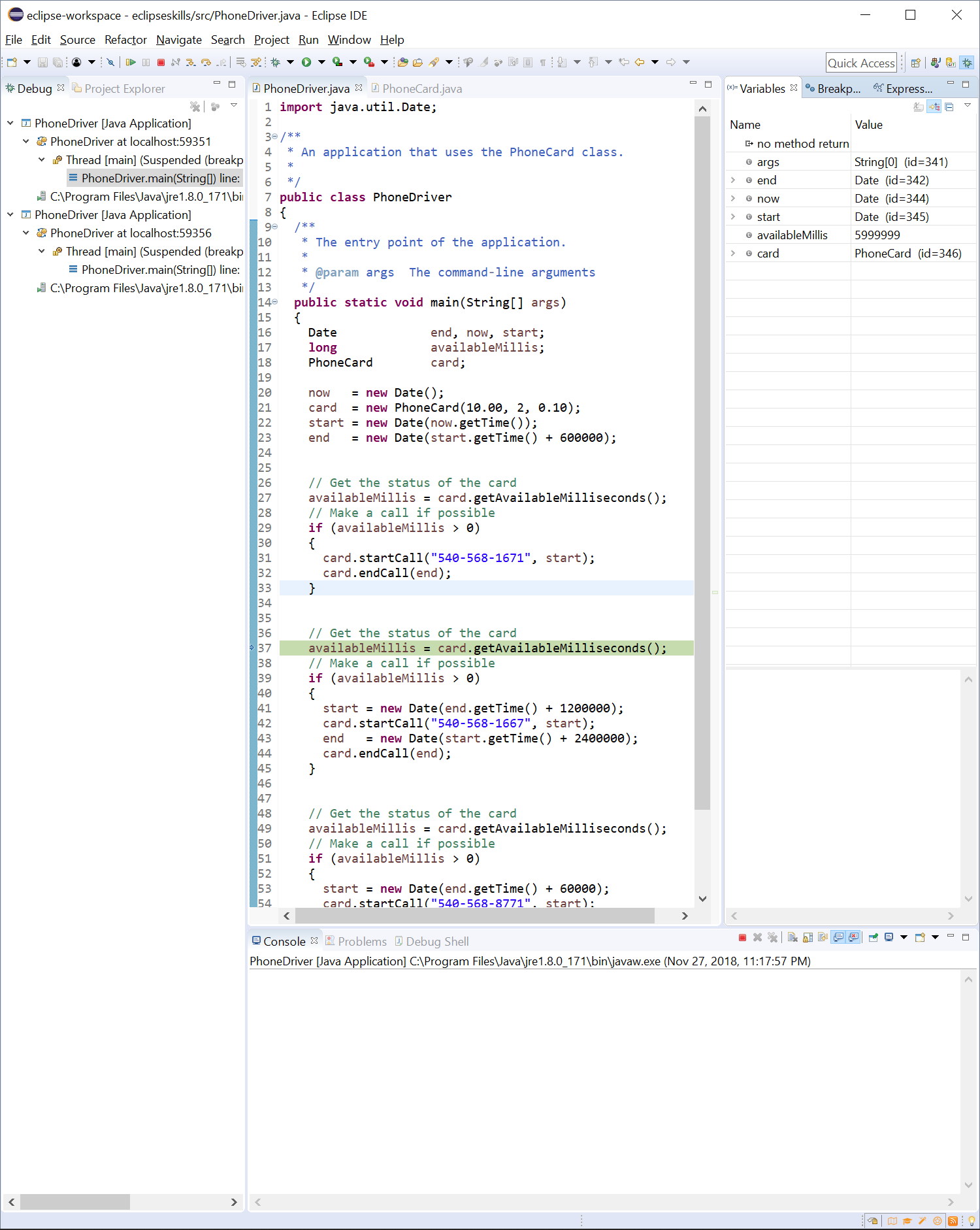
1. Click on debug.png. This will run PhoneDriver and stop the execution at the breakpoint (i.e., line 33). Note: If prompted, allow Eclipse to enter the "Debug Perspective".
2. What happened?

The debug perspective opened up and line 37 is now highlighted in green. Everytime I toggle breakpoint in line 33, it toggles line 37 instead.

***Part 3. Checking State Information:*** Another nice thing about running an application in a debugger is that, once you stop the execution at a breakpoint, you can check state information (e.g., the value of attributes and variables). This part of the lab will teach you how.

1. Click on the "Variables" tab on the left side of the debug window.
2. Click on the "tree icon" next to "Locals" to expand it.

**I don’t see Locals**



1. What is the current value of availableMillis?

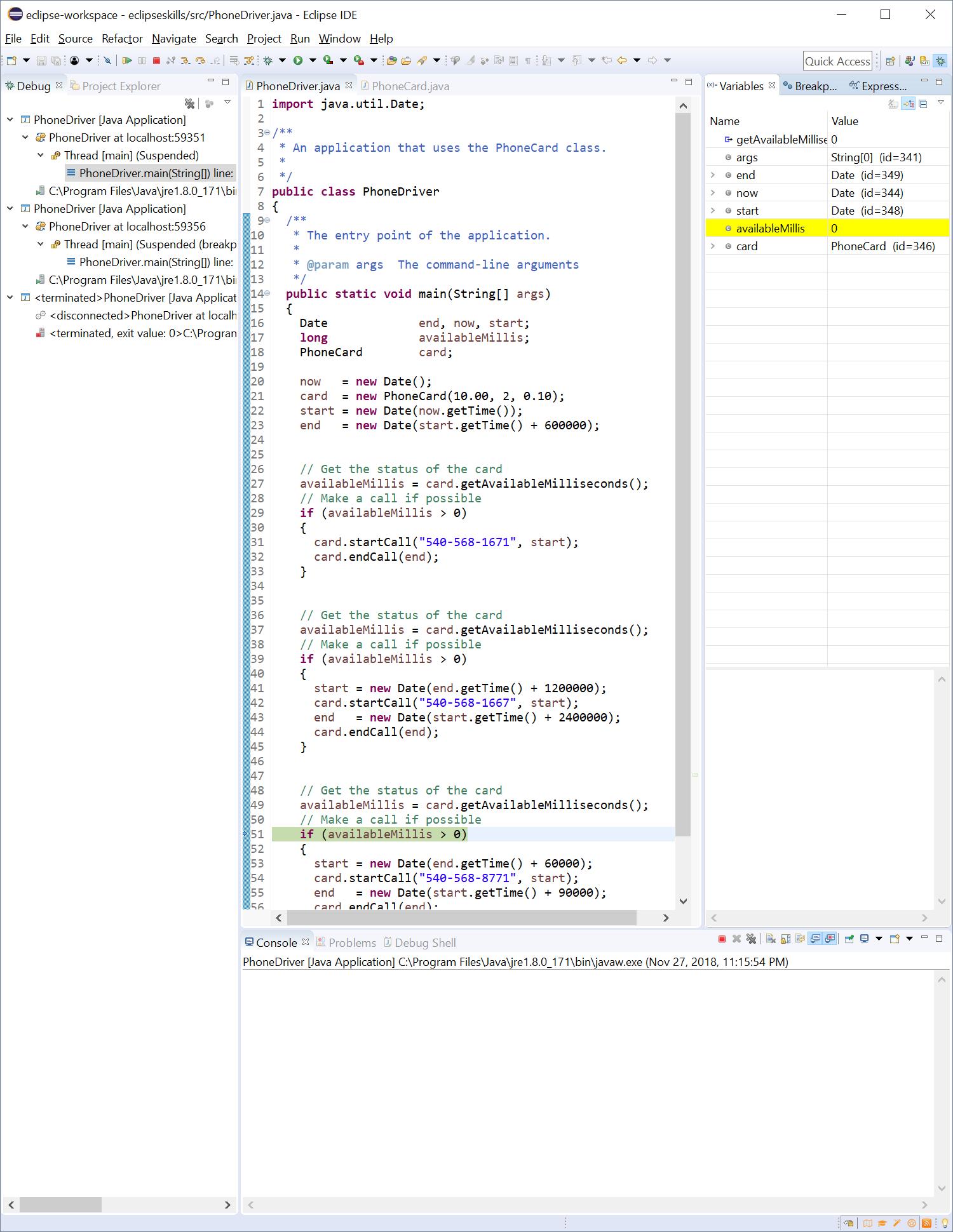
**The current value of availableMillis is 5999999**

***Part 4. Stepping Over Lines***: When running an application in a debugger, once you stop the execution at a breakpoint, you can continue the execution one "step" at a time. This part of the lab will teach you how.

1. Click on debug.png. This will run PhoneDriver again and stop the execution at the breakpoint (i.e., line 33).
2. Click on the step-over.png button.
3. What happened?

availableMillis turned yellow and is now showing 5399999

1. Click on the step-over.png button until the next if statement is highlighted.



1. What is the current value of availableMillis? (Hint: Look in the "Variables" tab. You may beed to scroll.)

**The current value of availableMillis is 0.**

1. Click on the resume.png button to run to the end of the application.

***Part 5: Stepping Into Lines:***So far, all of the "stepping" you have done has been in one method in one class. This is called "stepping over". You can also "step into" a line of code to see what happens there. This part of the lab will teach you how.

1. Click on debug.png. This will run PhoneDriver and stop the execution at the breakpoint (i.e., line 33).
2. Click on the step-into.png button.
3. What happened?

**It jumped to line 70**.

1. Click on the step-into.png button again.
2. What happened?

**It jumped to the if statement in line 71**

1. Look at the call stack in the "Debug" tab. It tells you what class and method you are in and where this method was called from.
2. What method is currently being executed (and what class is it in)?

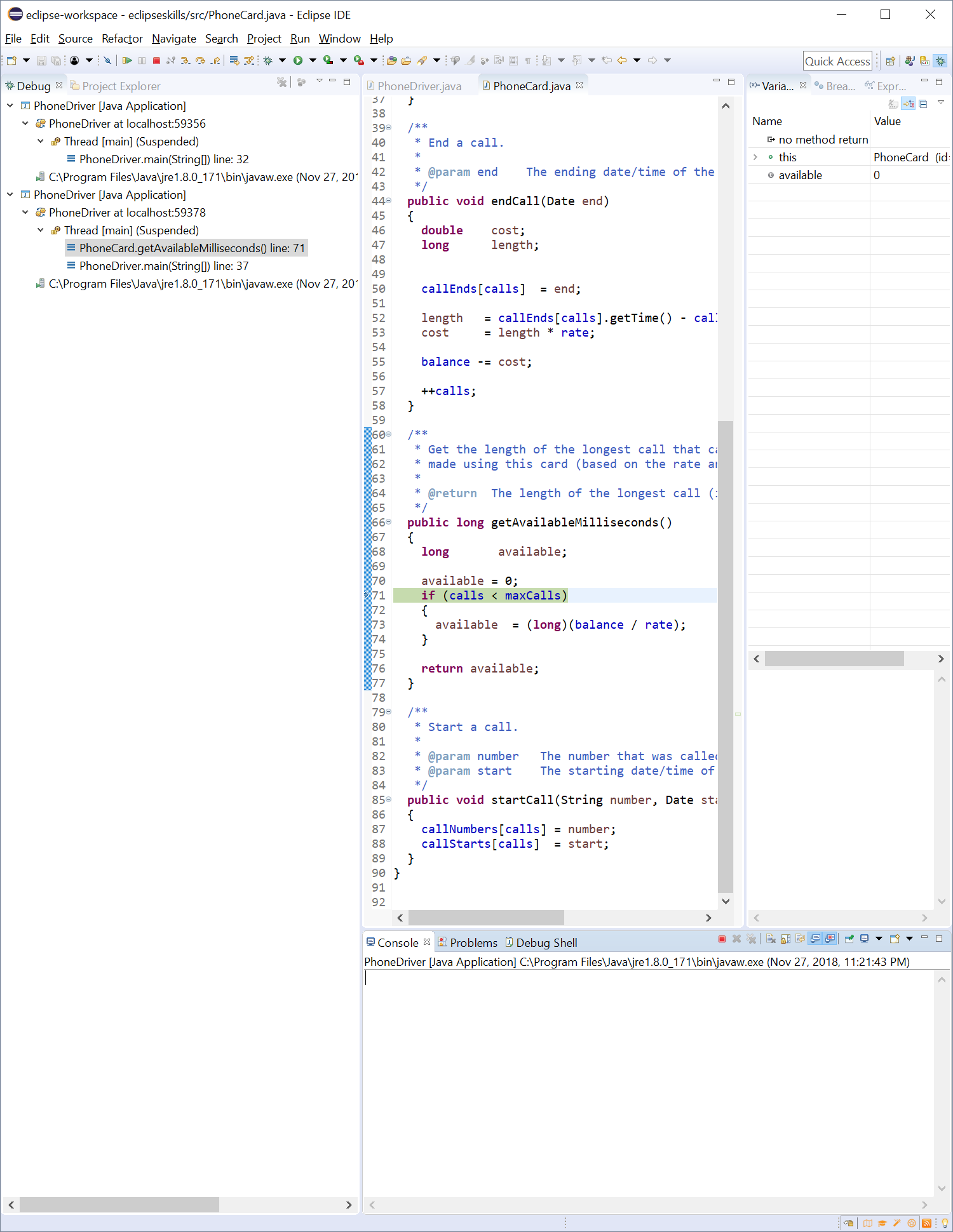
**getAvailableMilliseconds is being executed from Class PhoneCard**

1. What line is currently being executed?

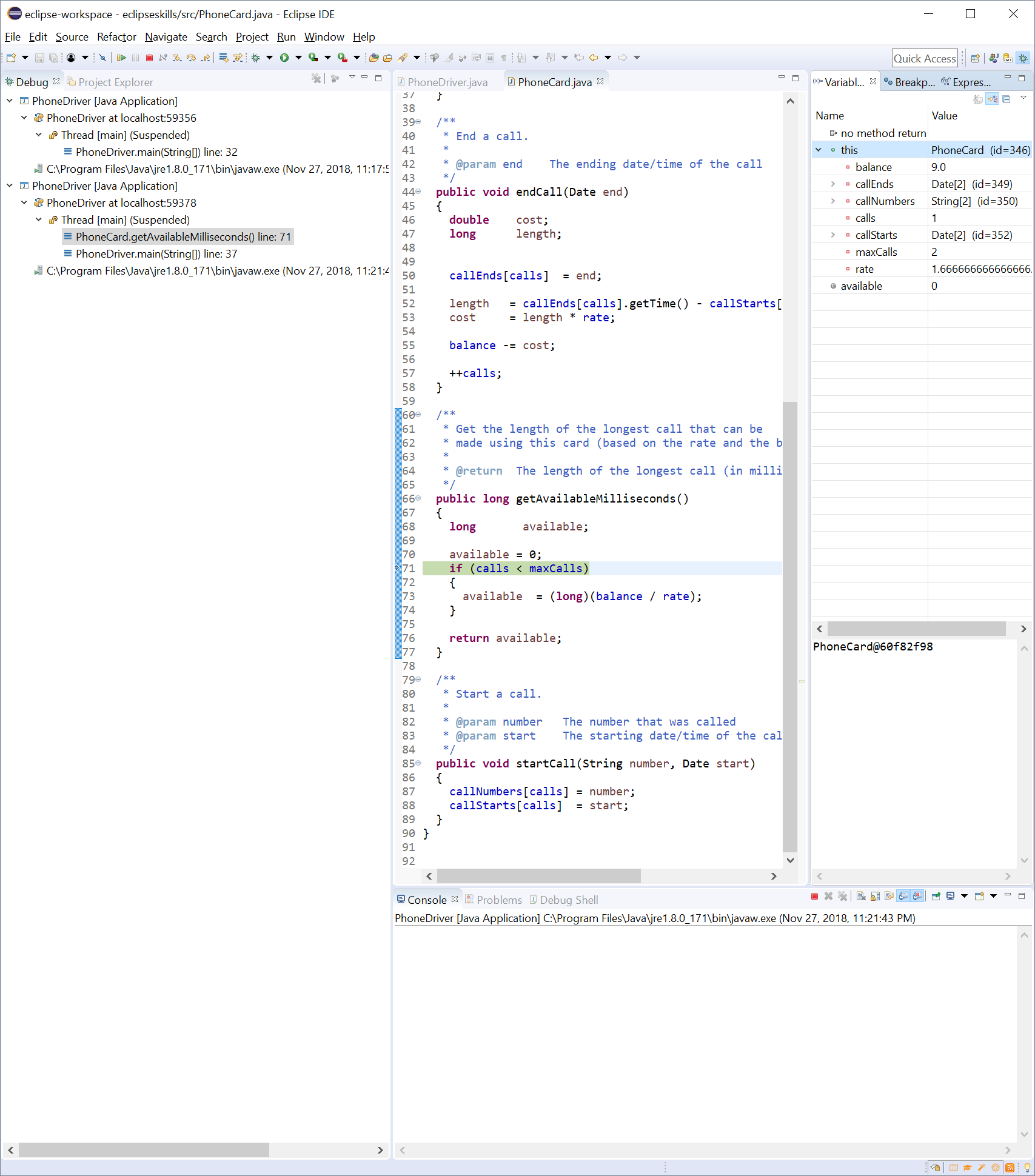
**Line 71 is currently being executed**

1. Where was this method called from?

**This method was called from main**



1. Click on the "triangle icon" next to this to expand it.



1. What is the current value of balance?

**The current value of balance is 9.0**

1. Click on the step-over.png button.
2. Click on the "triangle icon" next to callNumbers to expand it.
3. What is the current value of callNumbers[0]?

**The current value of callNumbers[0] is 540-568-1671**

1. Click on the "triangle icon" next to callStarts to expand it.
2. What is the current value of callStarts[0]?

**The current value of callStarts[0] is “Date”**

1. Why does it have that value?

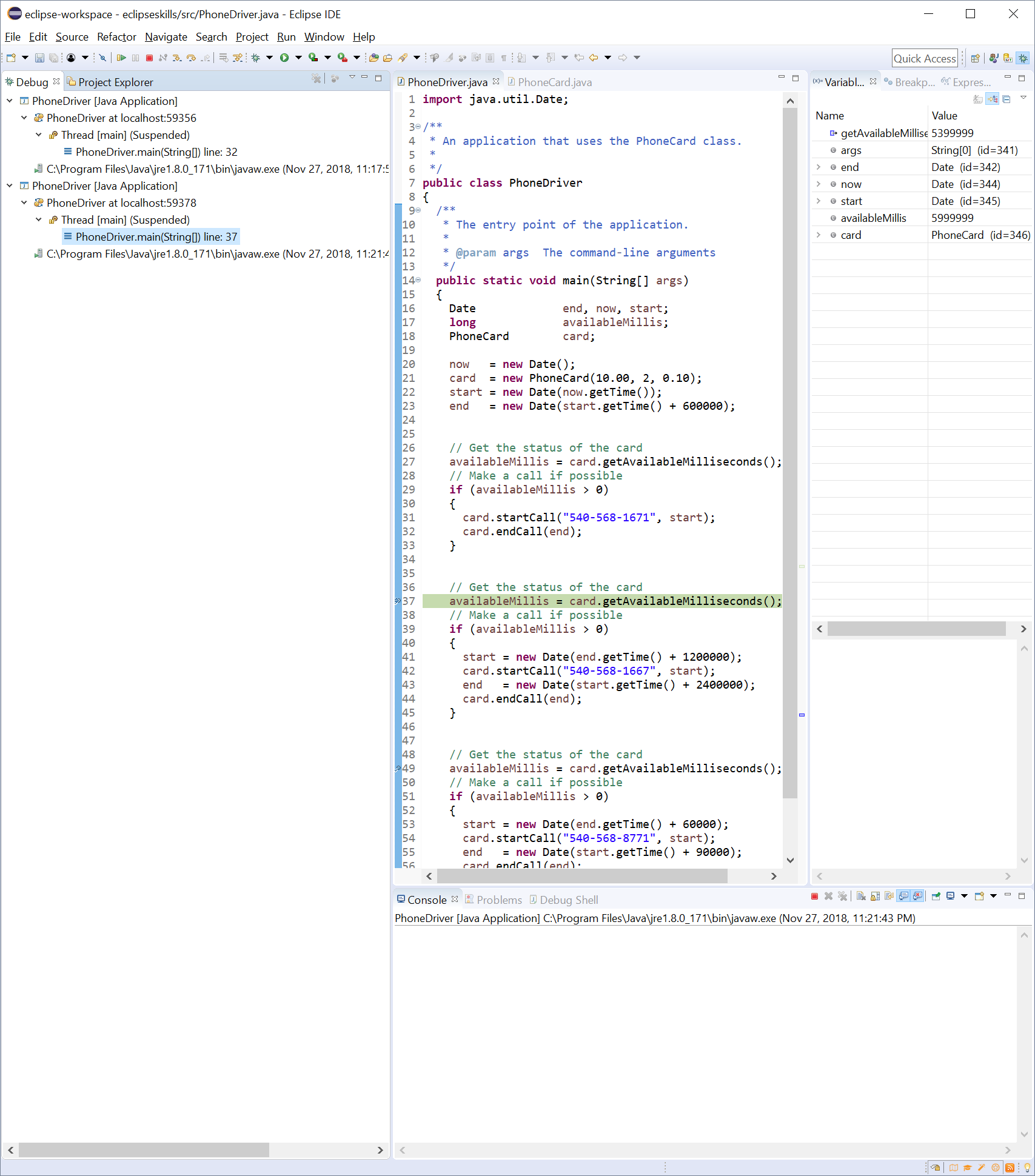
**I’m assuming it has that value because the call should start at the current time?**

1. Click on the step-over.png button twice.
2. What happened?

**Line 37 is now highlighted green**

1. Add a breakpoint at line 46 in PhoneDriver.java (i.e., the line that constructs a Date).
2. Click on the resume.png button. This will run the application to the next breakpoint (i.e., line 46).

**I tried to add a breakpoint to line 46 and it added it to line 49 instead. I think Eclipse hates me.**



1. Click on the step-into.png button.
2. Why didn't the debugger step into the Date constructor?

**The debugger went to line 39. I don’t know why it didn’t step into the Date constructor.**

1. Click on the resume.png button to run to the end.

I don’t think it ran to the end because it went to line 49 and the debugger says “already running.”

1. Click on Window+Perspective+Close Perspective to close the "Debug Perspective".

***Part 6: Advanced Topics:***This part of the lab will help you use the debugger more efficiently.

1. How can you display all of the breakpoints?

**By opening the Breakpoints tab inside of the debug perspective.**

1. What is a conditional breakpoint?

**It will stop at a line only if the Boolean value changes on it.**

1. How can you see variable references while debugging?

**By opening the variables tab inside of the debug perspective.**