Comp 311
Functional Programming

Nick Vrvilo, Two Sigma Investments
Robert "Corky" Cartwright, Rice University

September 6, 2018
Supplementary Slides

Using DrScala

DrScala

- Available from the course homepage:

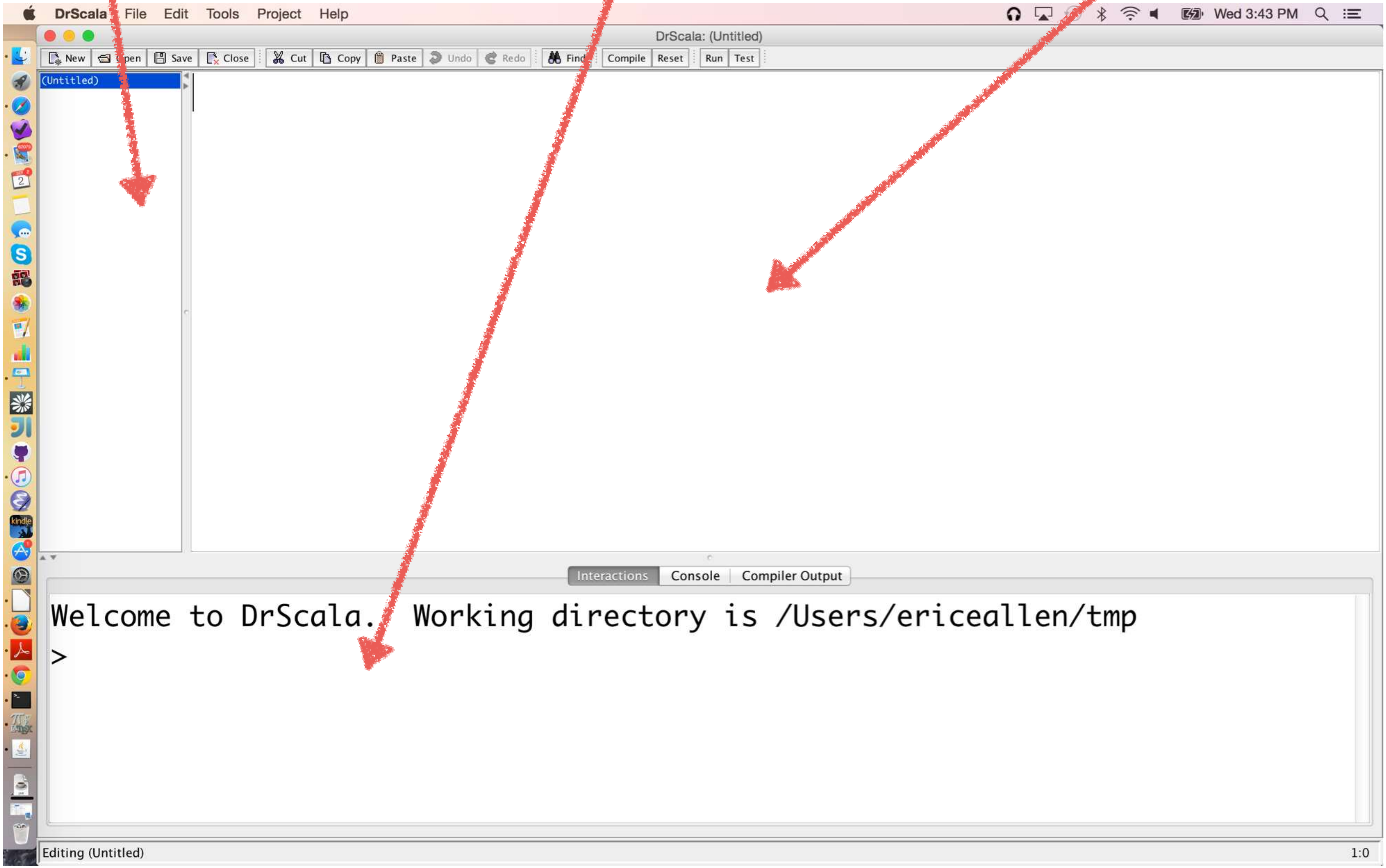
  https://comp311.rice.edu

- A lightweight development environment well-suited to the exercises we will do in this class

Open Files

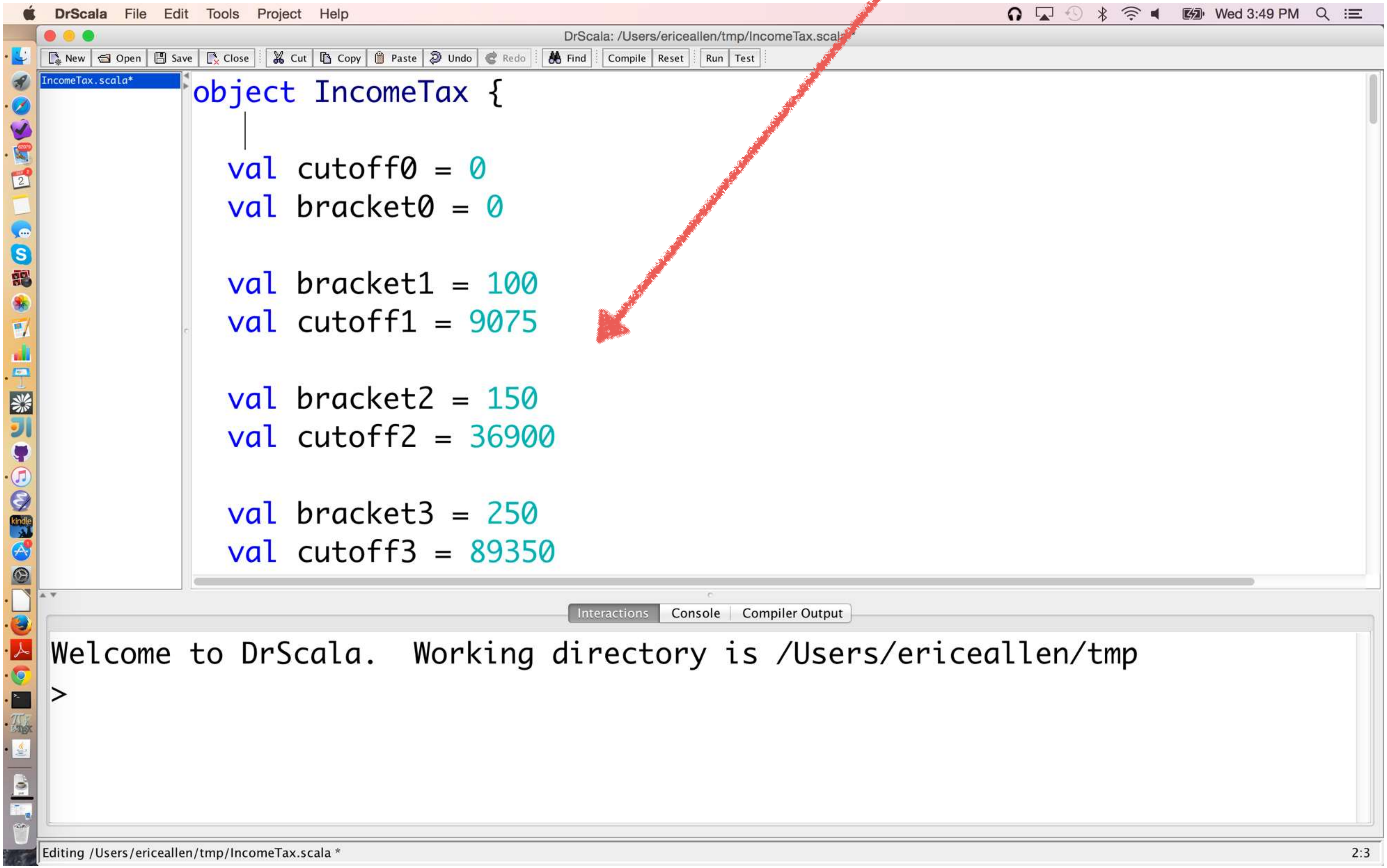
Interactions Pane

Definitions Pane



```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
>
```

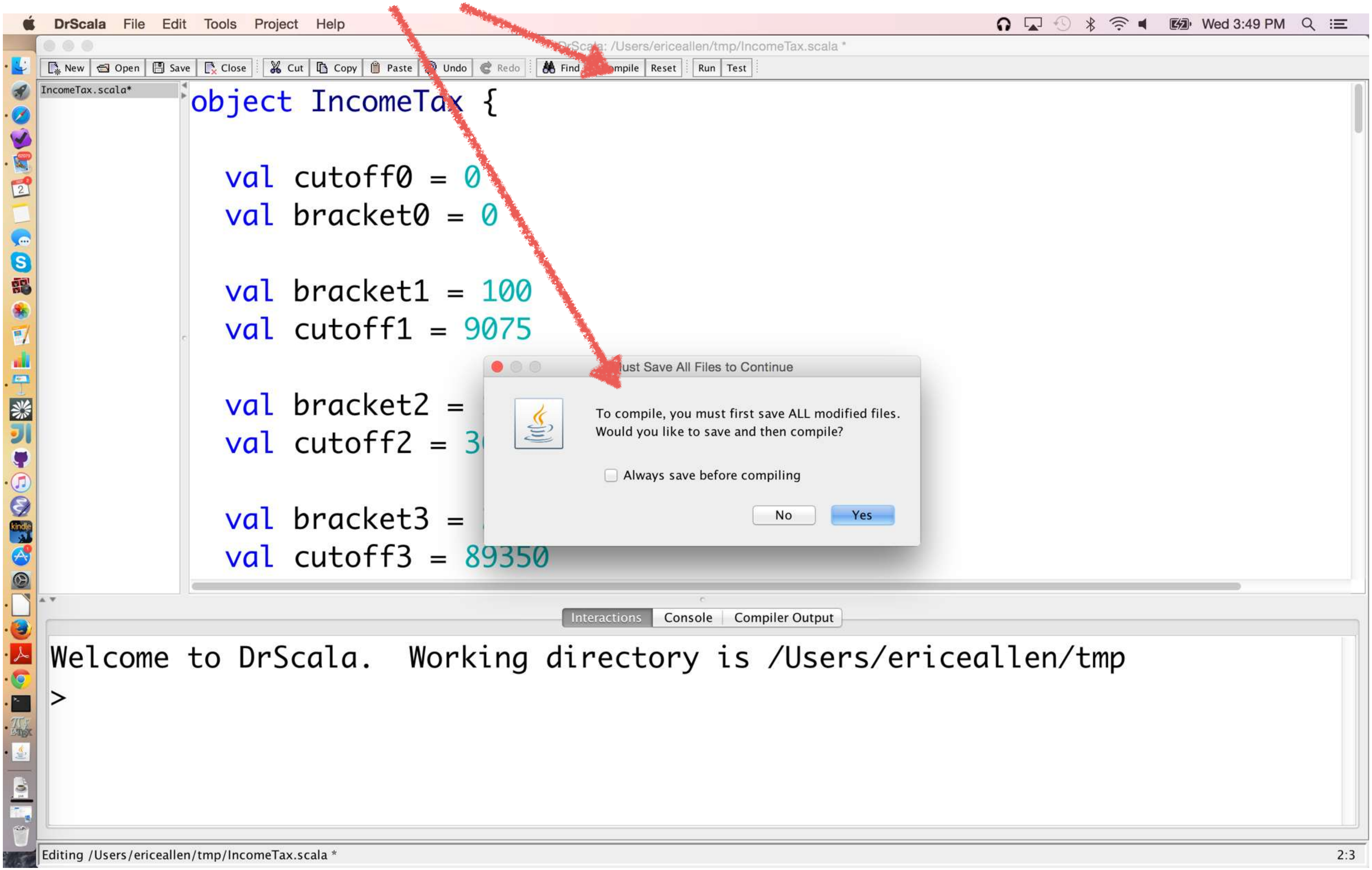
Define your program in the definitions pane


DrScala File Edit Tools Project Help
Wed 3:49 PM
DrScala: /Users/ericeallen/tmp/IncomeTax.scala*
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test
IncomeTax.scala*
Interactions Console Compiler Output
Editing /Users/ericeallen/tmp/IncomeTax.scala *
2:3

```
object IncomeTax {

  val cutoff0 = 0
  val bracket0 = 0

  val bracket1 = 100
  val cutoff1 = 9075

  val bracket2 = 150
  val cutoff2 = 36900

  val bracket3 = 250
  val cutoff3 = 89350
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
>
```

A prompt to save your program after hitting the Compile button



DrScala: /Users/ericeallen/tmp/IncomeTax.scala *

```
object IncomeTax {

  val cutoff0 = 0
  val bracket0 = 0

  val bracket1 = 100
  val cutoff1 = 9075

  val bracket2 =
  val cutoff2 = 3

  val bracket3 =
  val cutoff3 = 89350
```

ust Save All Files to Continue

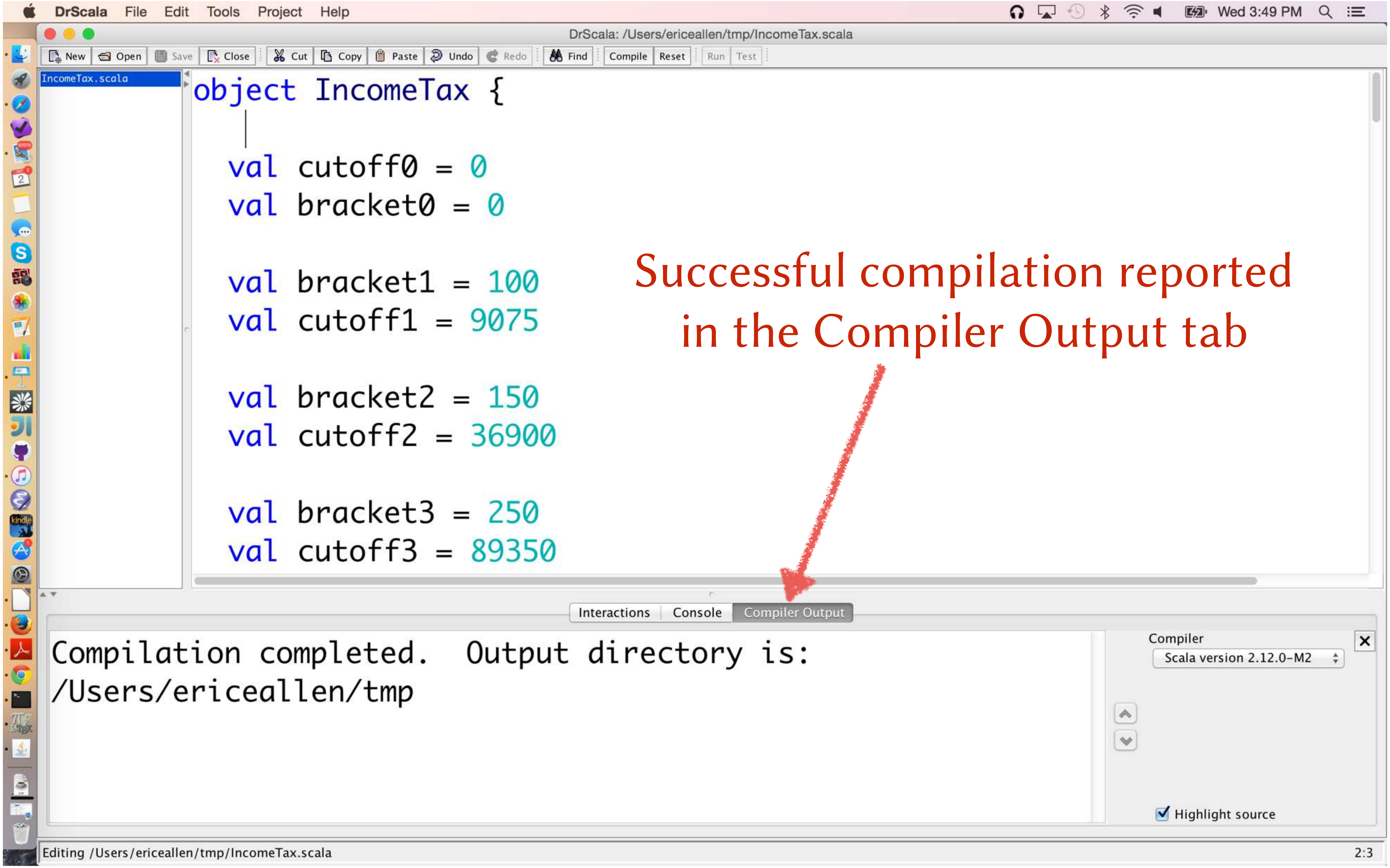
To compile, you must first save ALL modified files.
Would you like to save and then compile?

□ Always save before compiling

No Yes

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
>
```

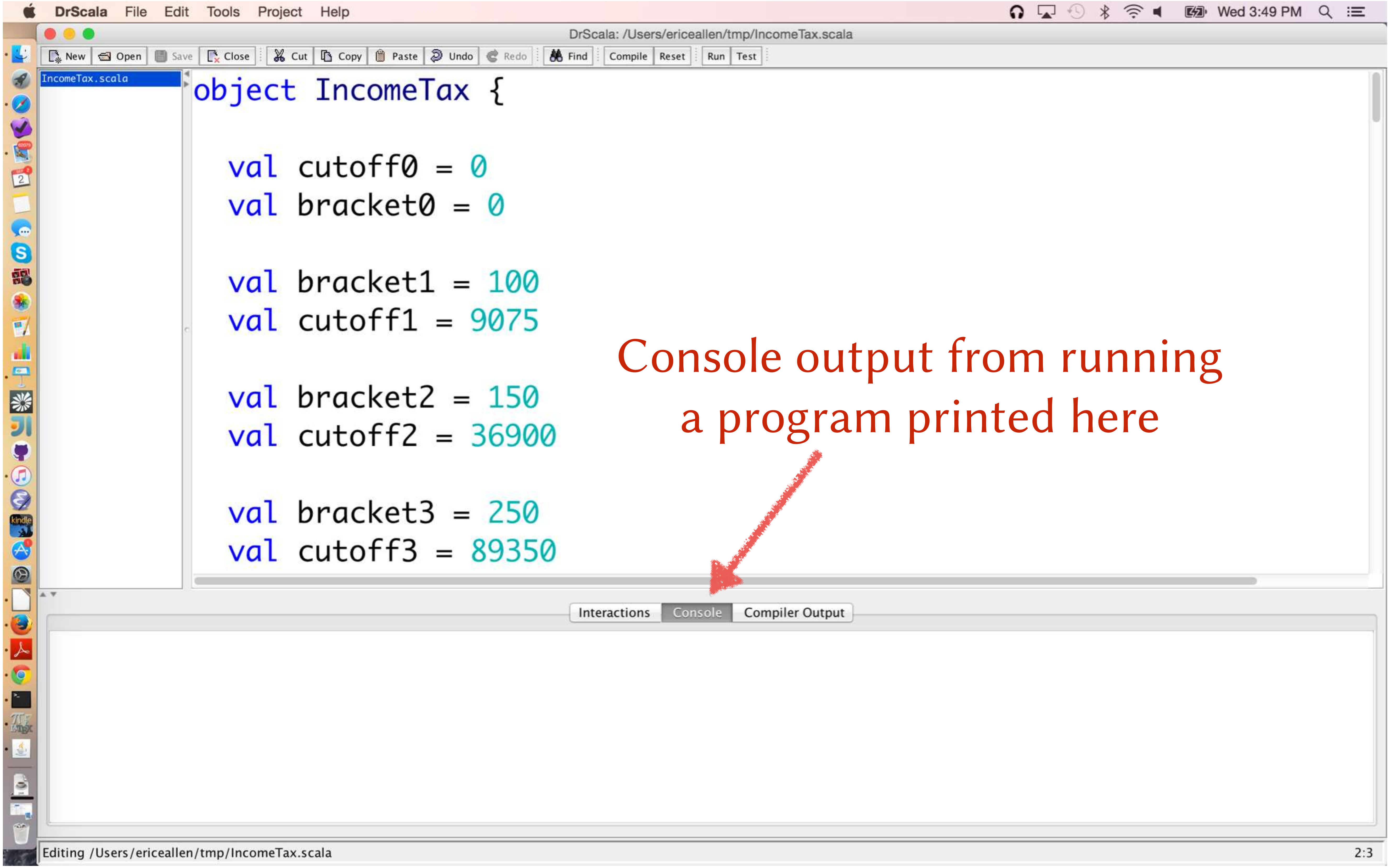
Editing /Users/ericeallen/tmp/IncomeTax.scala * 2:3



```
object IncomeTax {

  val cutoff0 = 0
  val bracket0 = 0

  val bracket1 = 100
  val cutoff1 = 9075

  val bracket2 = 150
  val cutoff2 = 36900

  val bracket3 = 250
  val cutoff3 = 89350
```

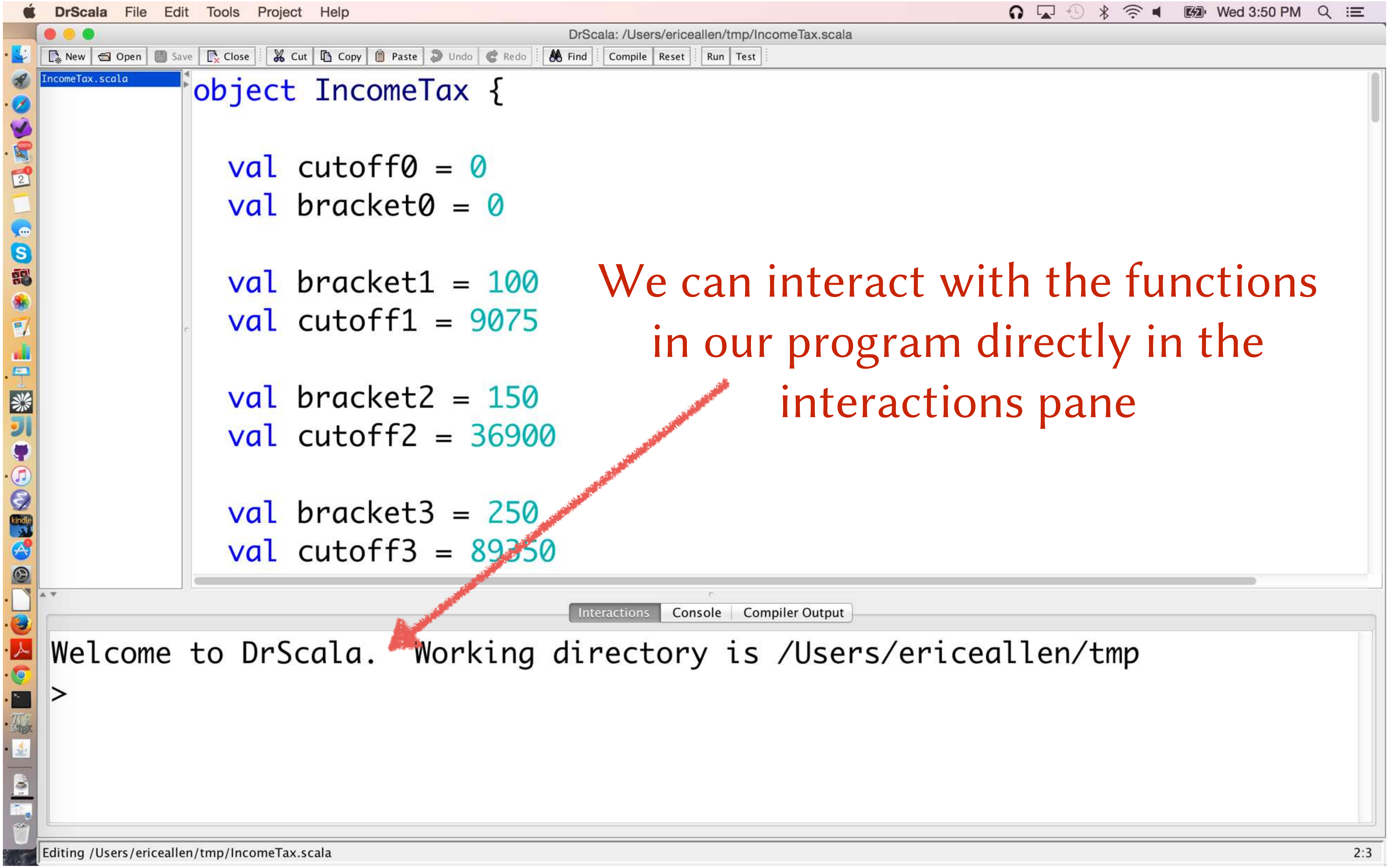
Successful compilation reported in the Compiler Output tab

```
Compilation completed.  Output directory is:
/Users/ericeallen/tmp
```



```scala
object IncomeTax {

  val cutoff0 = 0
  val bracket0 = 0

  val bracket1 = 100
  val cutoff1 = 9075

  val bracket2 = 150
  val cutoff2 = 36900

  val bracket3 = 250
  val cutoff3 = 89350
```

Console output from running
a program printed here

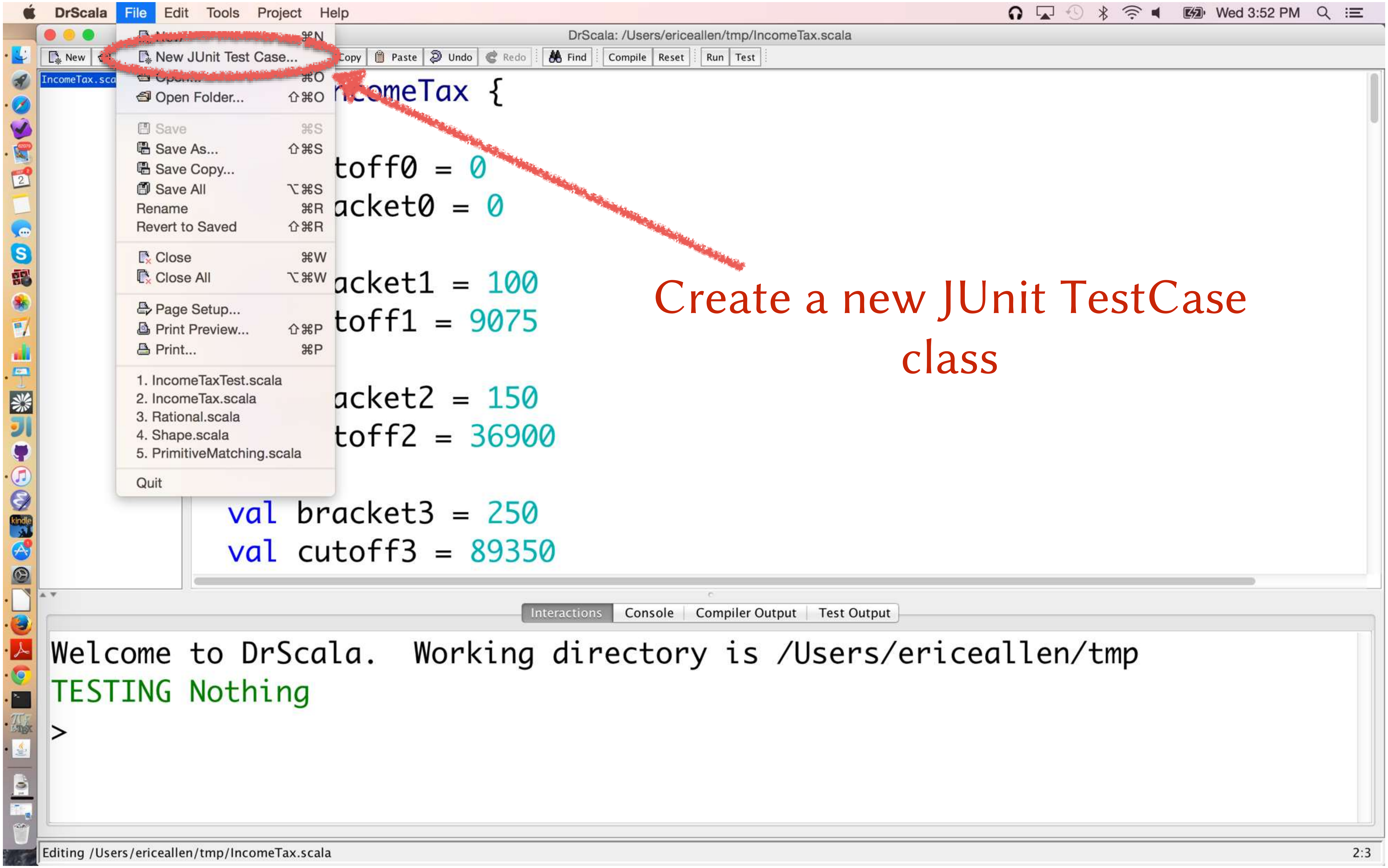


```
object IncomeTax {

  val cutoff0 = 0
  val bracket0 = 0

  val bracket1 = 100
  val cutoff1 = 9075

  val bracket2 = 150
  val cutoff2 = 36900

  val bracket3 = 250
  val cutoff3 = 89350
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
>
```

Editing /Users/ericeallen/tmp/IncomeTax.scala 2:3

We can interact with the functions in our program directly in the interactions pane

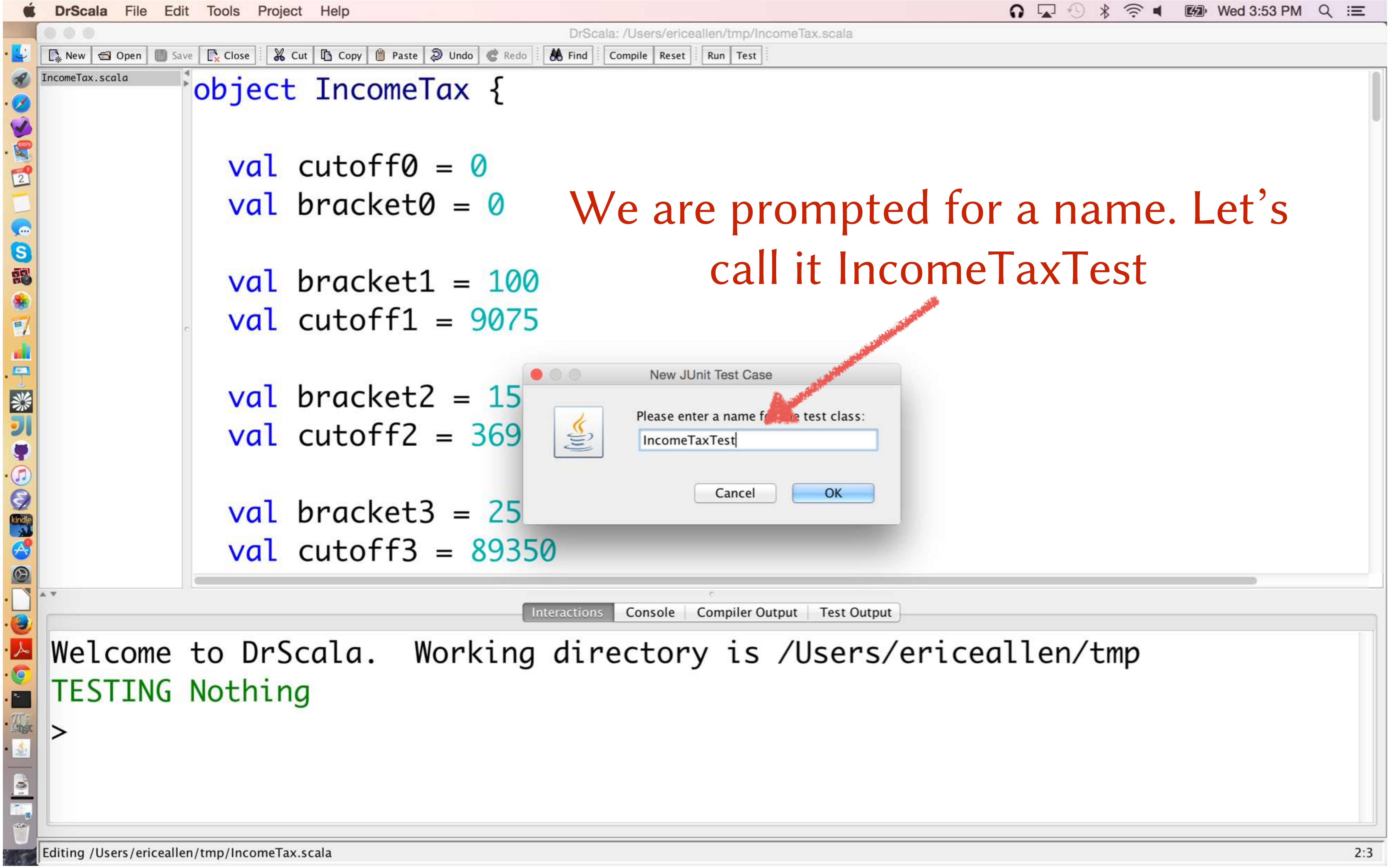


```
ncomeTax {

toff0 = 0
acket0 = 0

acket1 = 100
toff1 = 9075

acket2 = 150
toff2 = 36900

val bracket3 = 250
val cutoff3 = 89350
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
TESTING Nothing
>
```

Editing /Users/ericeallen/tmp/IncomeTax.scala 2:3

Create a new JUnit TestCase class



```
object IncomeTax {

  val cutoff0 = 0
  val bracket0 = 0

  val bracket1 = 100
  val cutoff1 = 9075

  val bracket2 = 15
  val cutoff2 = 369

  val bracket3 = 25
  val cutoff3 = 89350
```

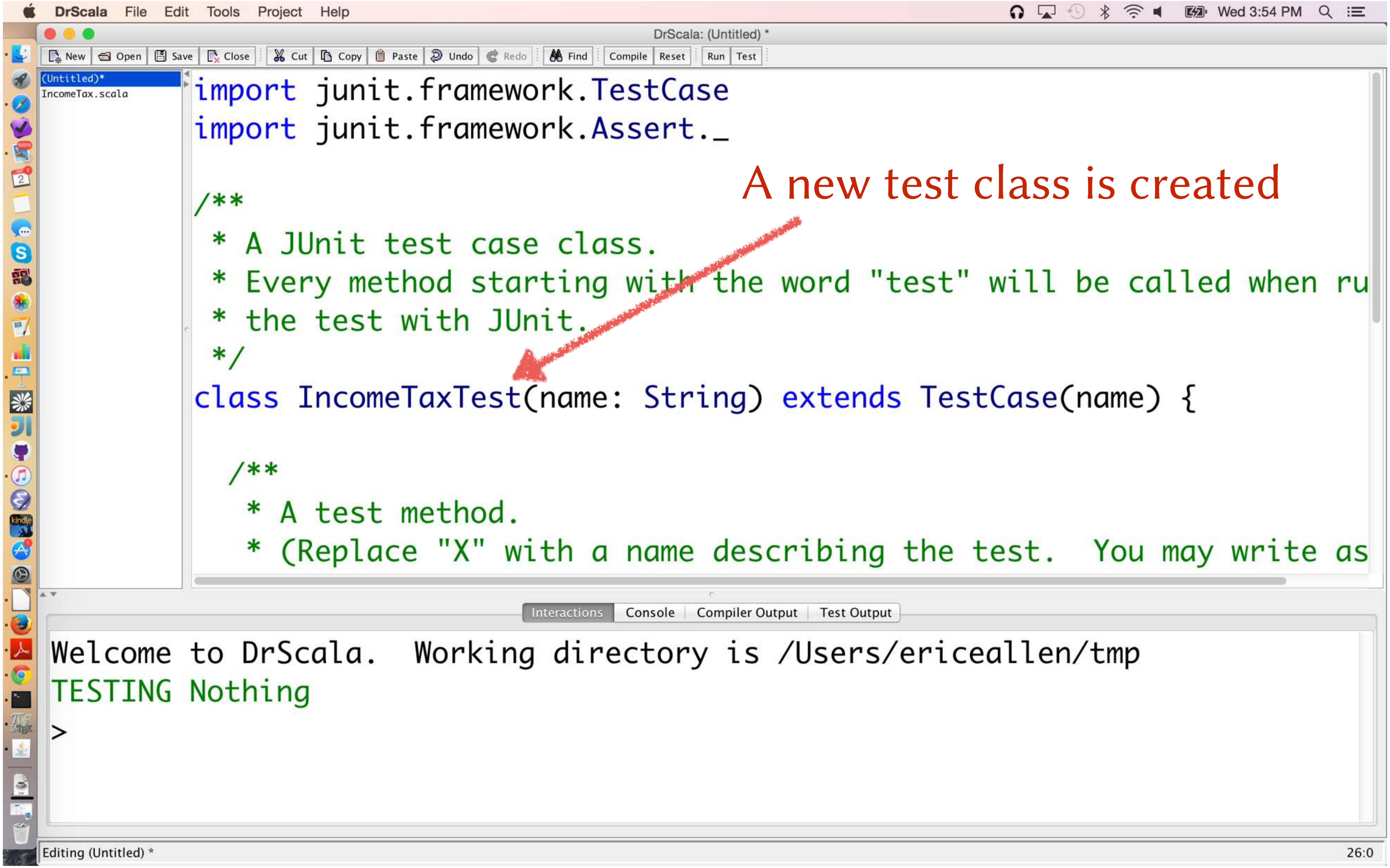
Please enter a name for the test class:

IncomeTaxTest

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
TESTING Nothing
>
```

Editing /Users/ericeallen/tmp/IncomeTax.scala 2:3

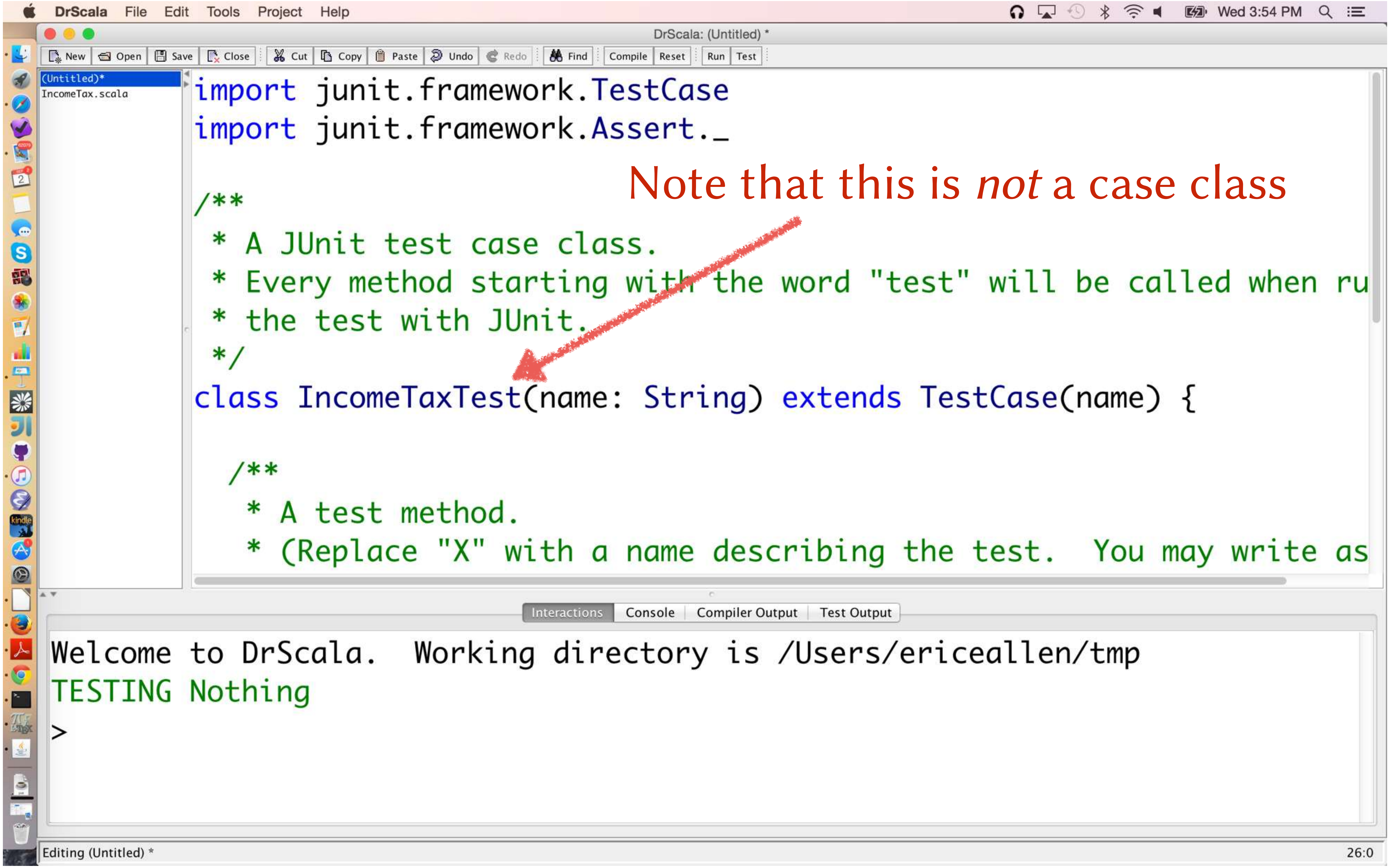
We are prompted for a name. Let’s call it IncomeTaxTest



```
import junit.framework.TestCase
import junit.framework.Assert._

/**
 * A JUnit test case class.
 * Every method starting with the word "test" will be called when ru
 * the test with JUnit.
 */
class IncomeTaxTest(name: String) extends TestCase(name) {

  /**
   * A test method.
   * (Replace "X" with a name describing the test.  You may write as
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
TESTING Nothing
>
```

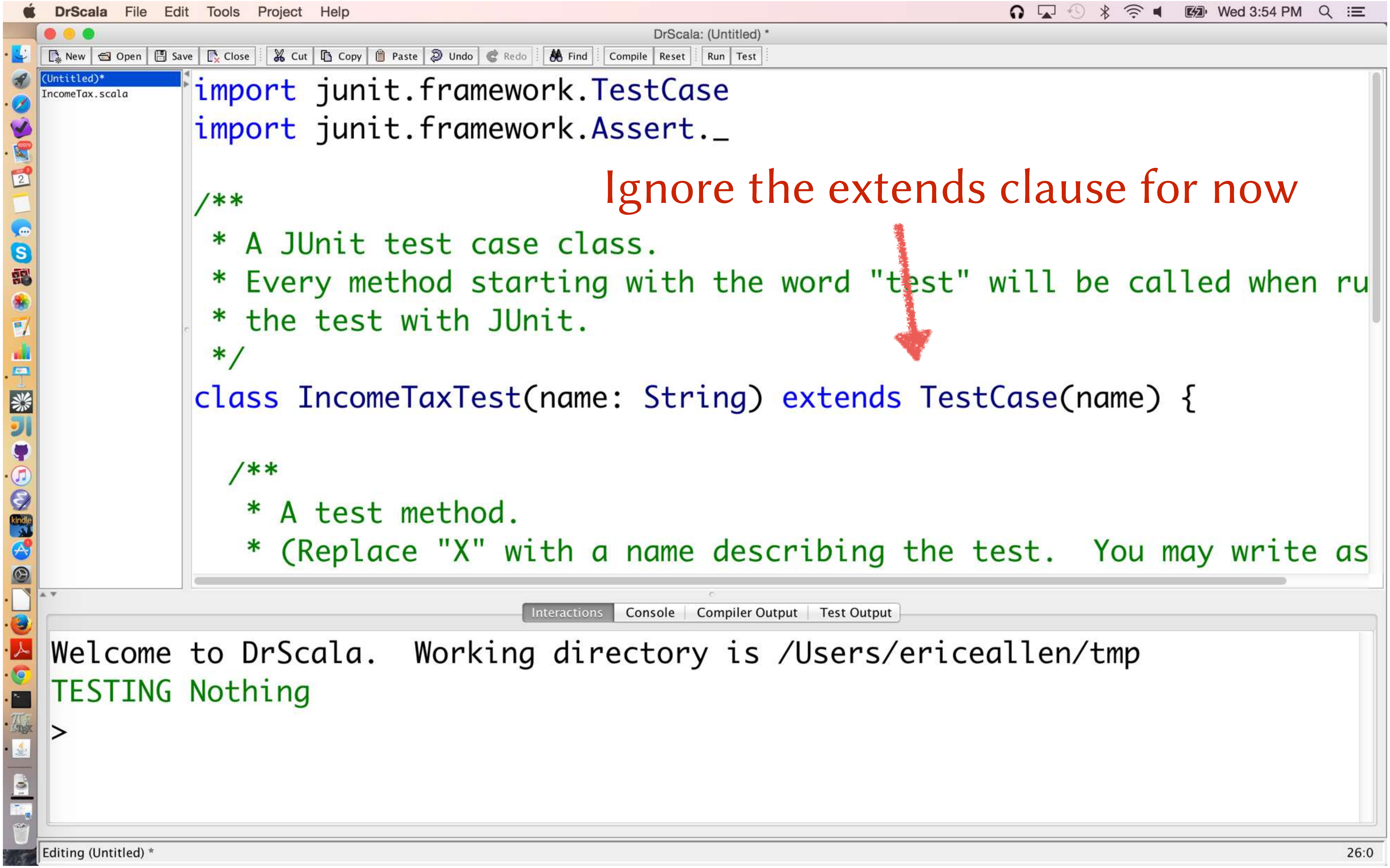
A new test class is created



```
import junit.framework.TestCase
import junit.framework.Assert._

/**
 * A JUnit test case class.
 * Every method starting with the word "test" will be called when ru
 * the test with JUnit.
 */
class IncomeTaxTest(name: String) extends TestCase(name) {

  /**
   * A test method.
   * (Replace "X" with a name describing the test.  You may write as
```

Note that this is *not* a case class

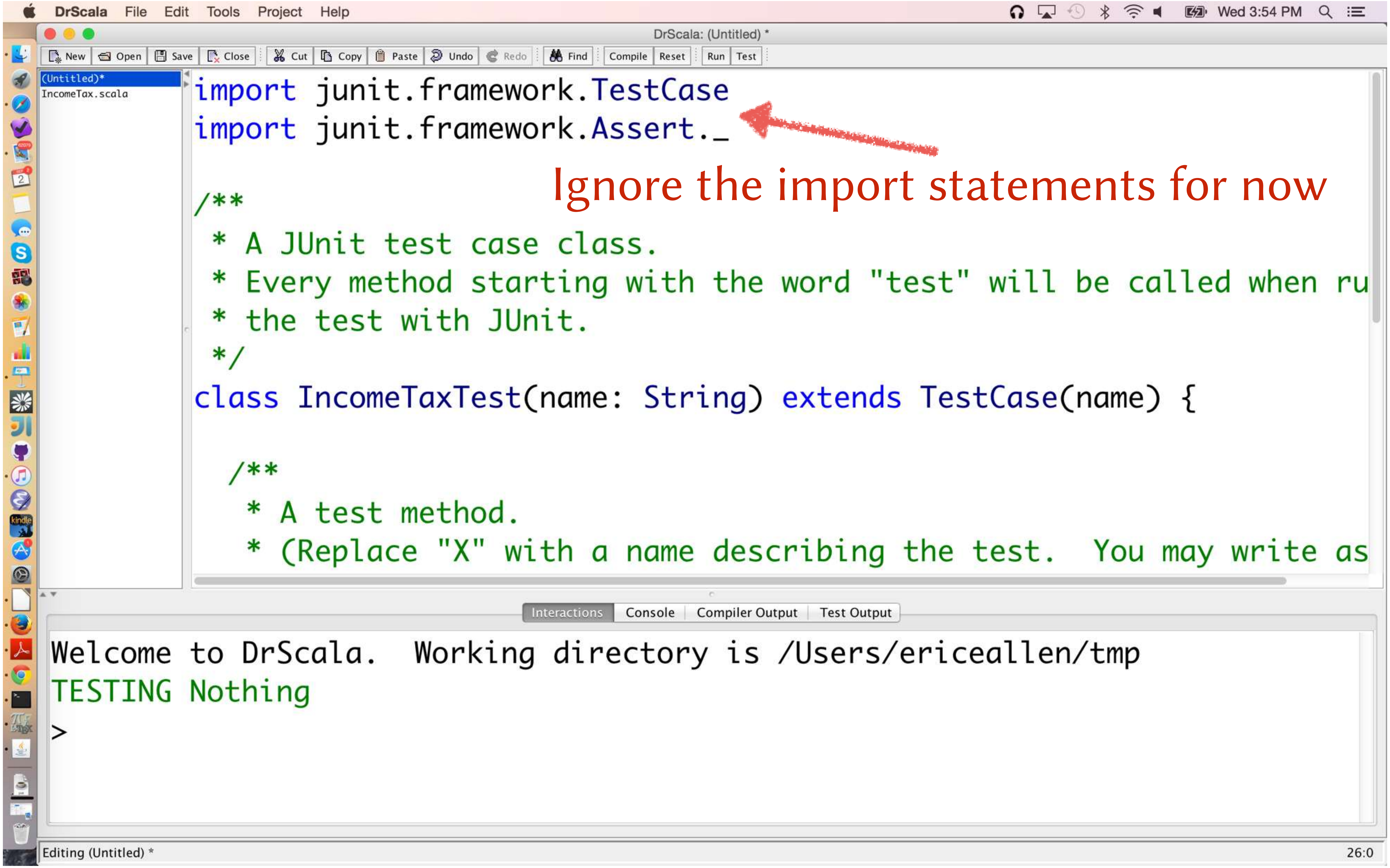
```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
TESTING Nothing
>
```


DrScala File Edit Tools Project Help
Wed 3:54 PM
DrScala: (Untitled) *
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test
(Untitled)*
IncomeTax.scala
Interactions Console Compiler Output Test Output
Editing (Untitled) *
26:0

```
import junit.framework.TestCase
import junit.framework.Assert._

/**
 * A JUnit test case class.
 * Every method starting with the word "test" will be called when ru
 * the test with JUnit.
 */
class IncomeTaxTest(name: String) extends TestCase(name) {

  /**
   * A test method.
   * (Replace "X" with a name describing the test.  You may write as
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
TESTING Nothing
>
```

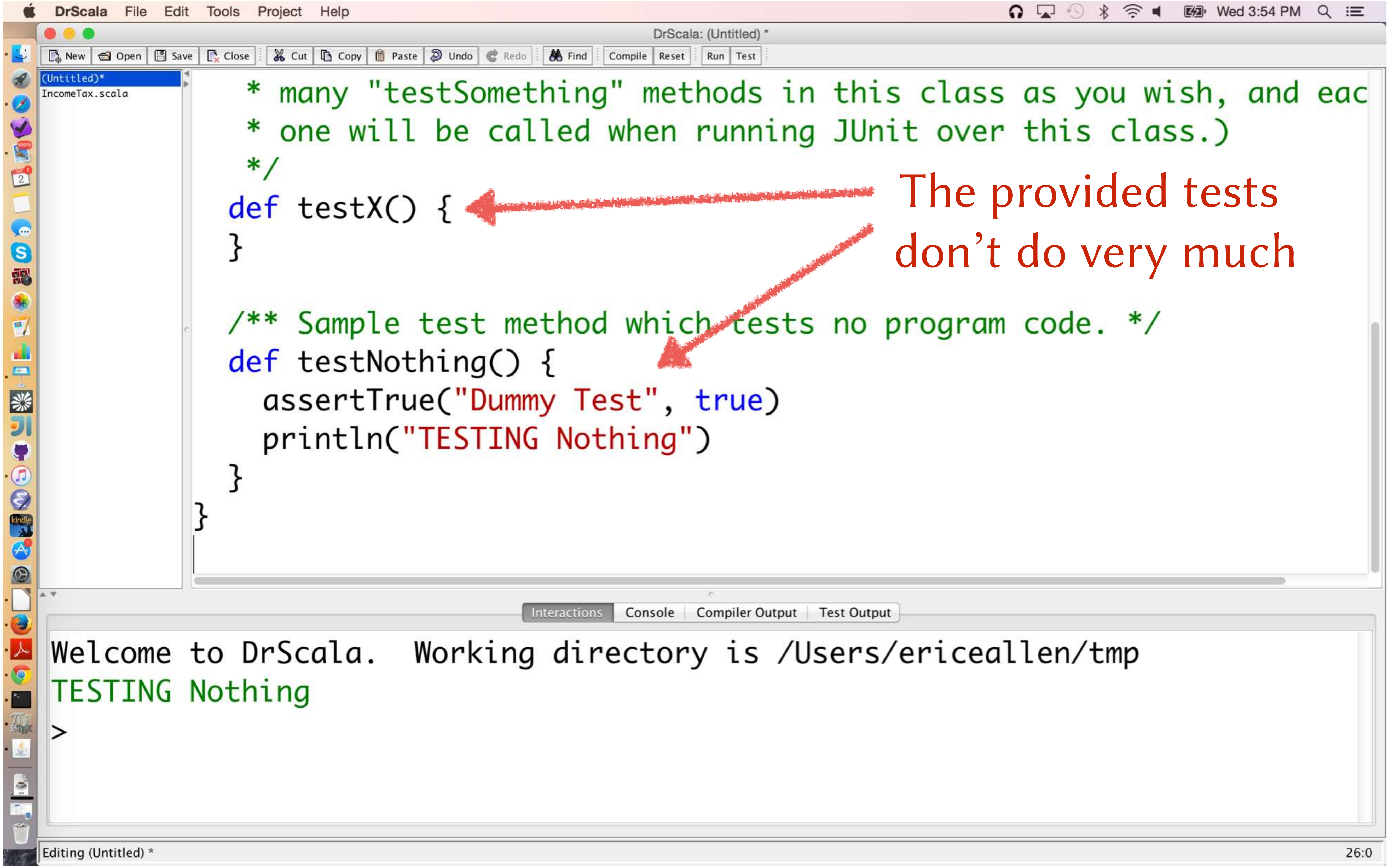
Ignore the extends clause for now



```
import junit.framework.TestCase
import junit.framework.Assert._

/**
 * A JUnit test case class.
 * Every method starting with the word "test" will be called when ru
 * the test with JUnit.
 */
class IncomeTaxTest(name: String) extends TestCase(name) {

  /**
   * A test method.
   * (Replace "X" with a name describing the test.  You may write as
```

Ignore the import statements for now

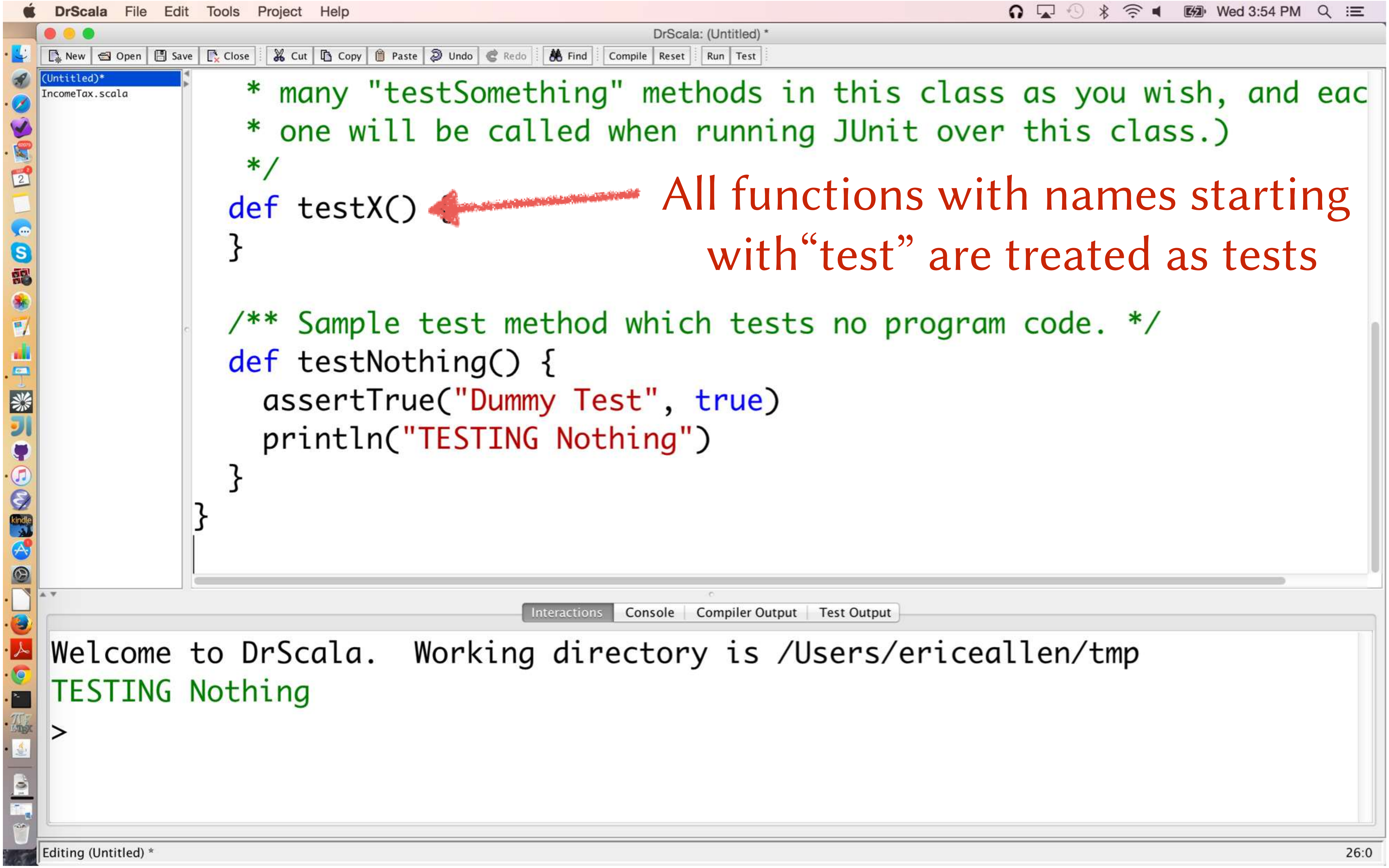
```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
TESTING Nothing
>
```



```
   * many "testSomething" methods in this class as you wish, and eac
   * one will be called when running JUnit over this class.)
   */
  def testX() {
  }

  /** Sample test method which tests no program code. */
  def testNothing() {
    assertTrue("Dummy Test", true)
    println("TESTING Nothing")
  }
}
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
TESTING Nothing
>
```

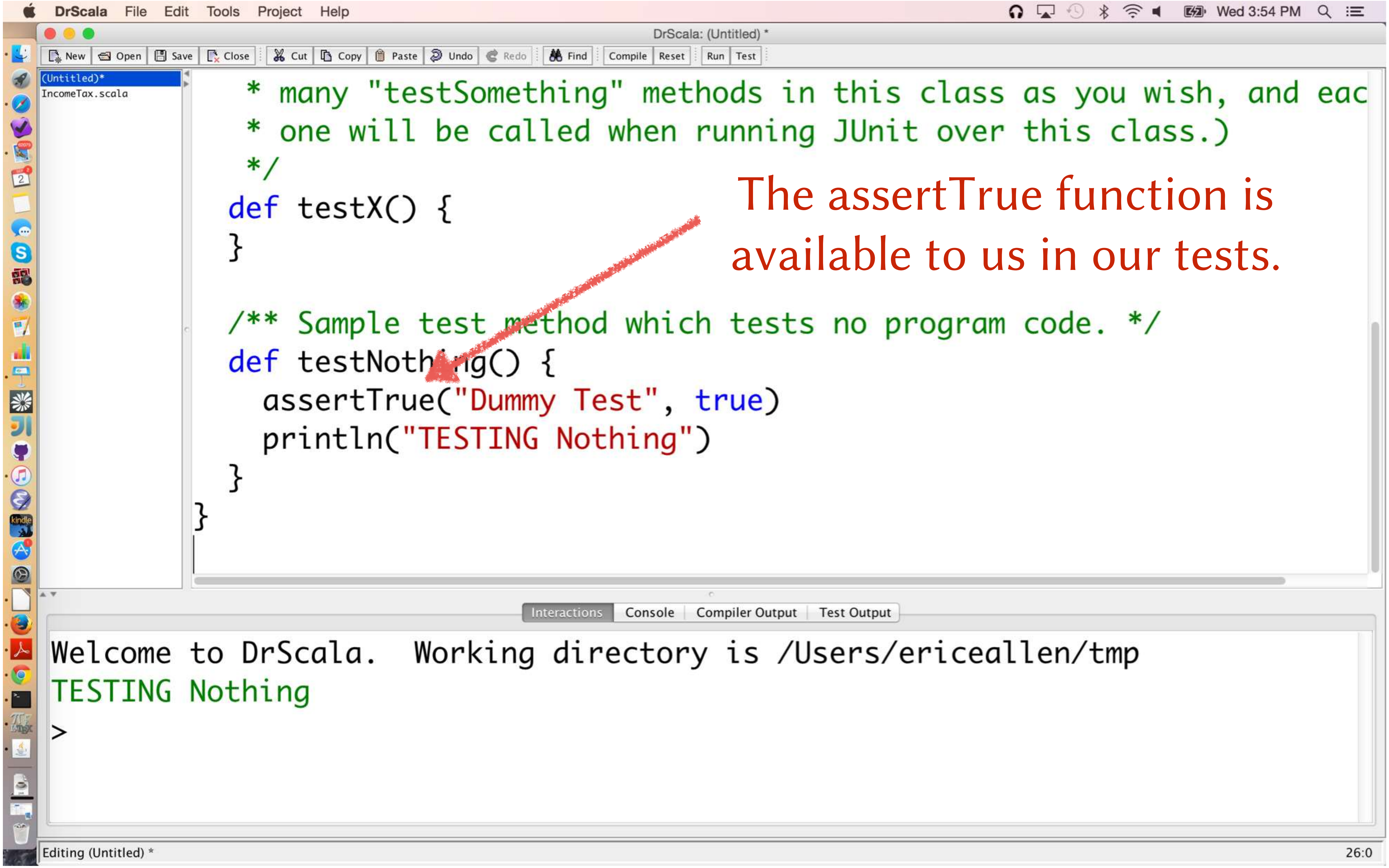
The provided tests don't do very much


DrScala File Edit Tools Project Help
Wed 3:54 PM
DrScala: (Untitled) *
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test
(Untitled)*
IncomeTax.scala
Interactions Console Compiler Output Test Output
Editing (Untitled) *
26:0

```
   * many "testSomething" methods in this class as you wish, and eac
   * one will be called when running JUnit over this class.)
   */
  def testX() {
  }

  /** Sample test method which tests no program code. */
  def testNothing() {
    assertTrue("Dummy Test", true)
    println("TESTING Nothing")
  }
}
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
TESTING Nothing
>
```

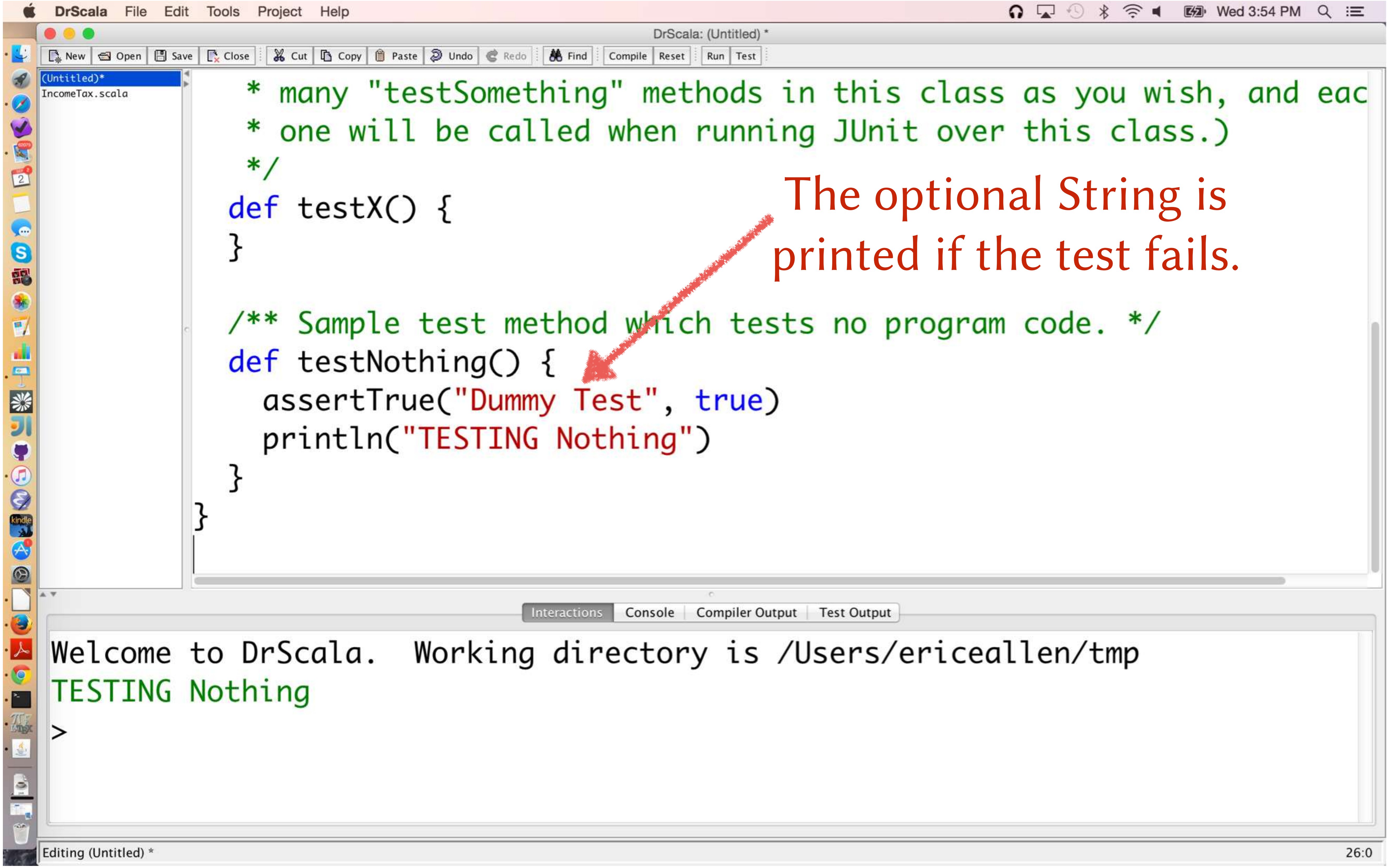
All functions with names starting with“test” are treated as tests



```
   * many "testSomething" methods in this class as you wish, and eac
   * one will be called when running JUnit over this class.)
   */
  def testX() {
  }

  /** Sample test method which tests no program code. */
  def testNothing() {
    assertTrue("Dummy Test", true)
    println("TESTING Nothing")
  }
}
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
TESTING Nothing
>
```

The assertTrue function is available to us in our tests.

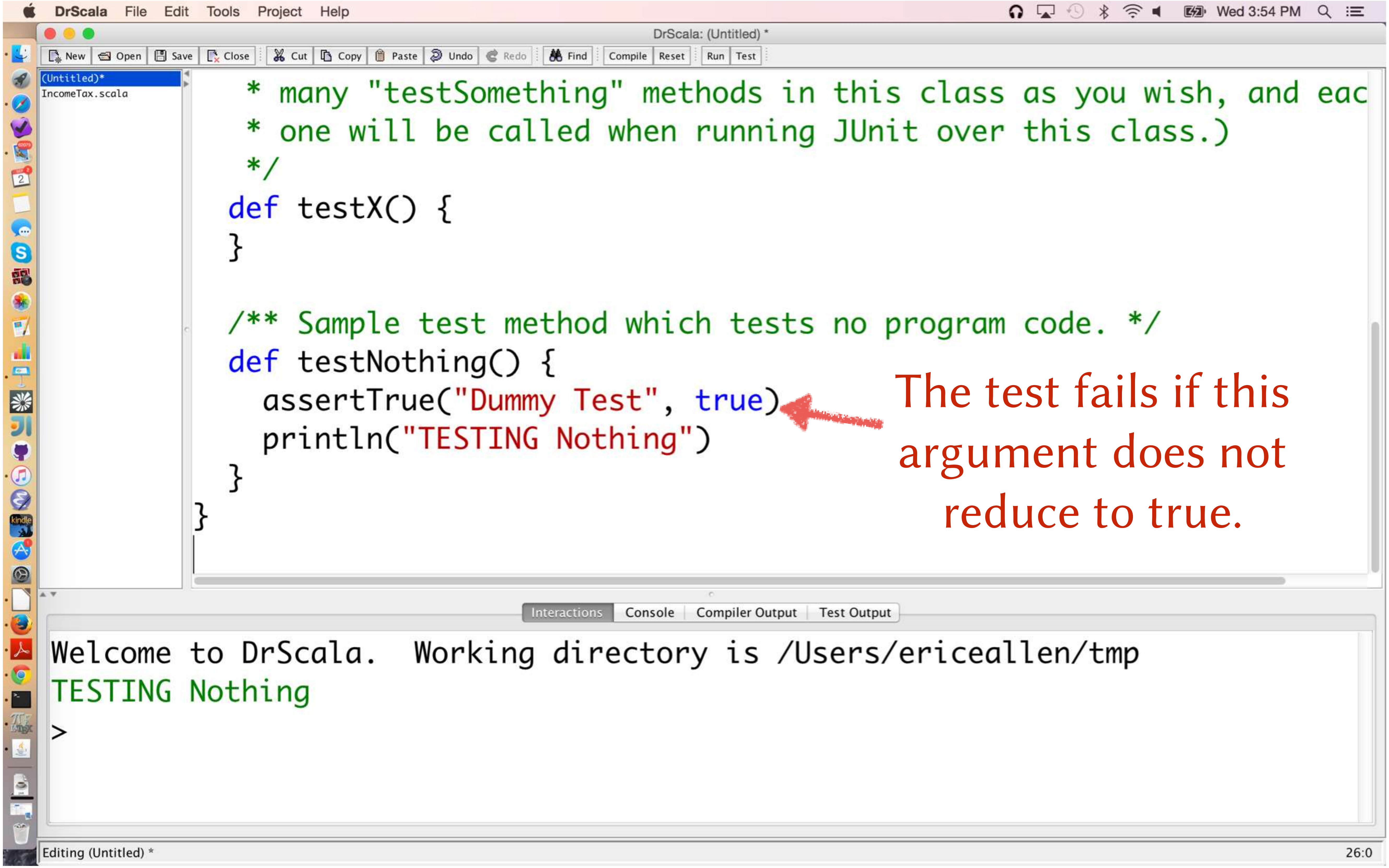


DrScala File Edit Tools Project Help
Wed 3:54 PM
DrScala: (Untitled) *
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test
(Untitled)*
IncomeTax.scala
Interactions Console Compiler Output Test Output
Editing (Untitled) *
26:0

```
   * many "testSomething" methods in this class as you wish, and eac
   * one will be called when running JUnit over this class.)
   */
  def testX() {
  }

  /** Sample test method which tests no program code. */
  def testNothing() {
    assertTrue("Dummy Test", true)
    println("TESTING Nothing")
  }
}
```

The optional String is printed if the test fails.

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
TESTING Nothing
>
```

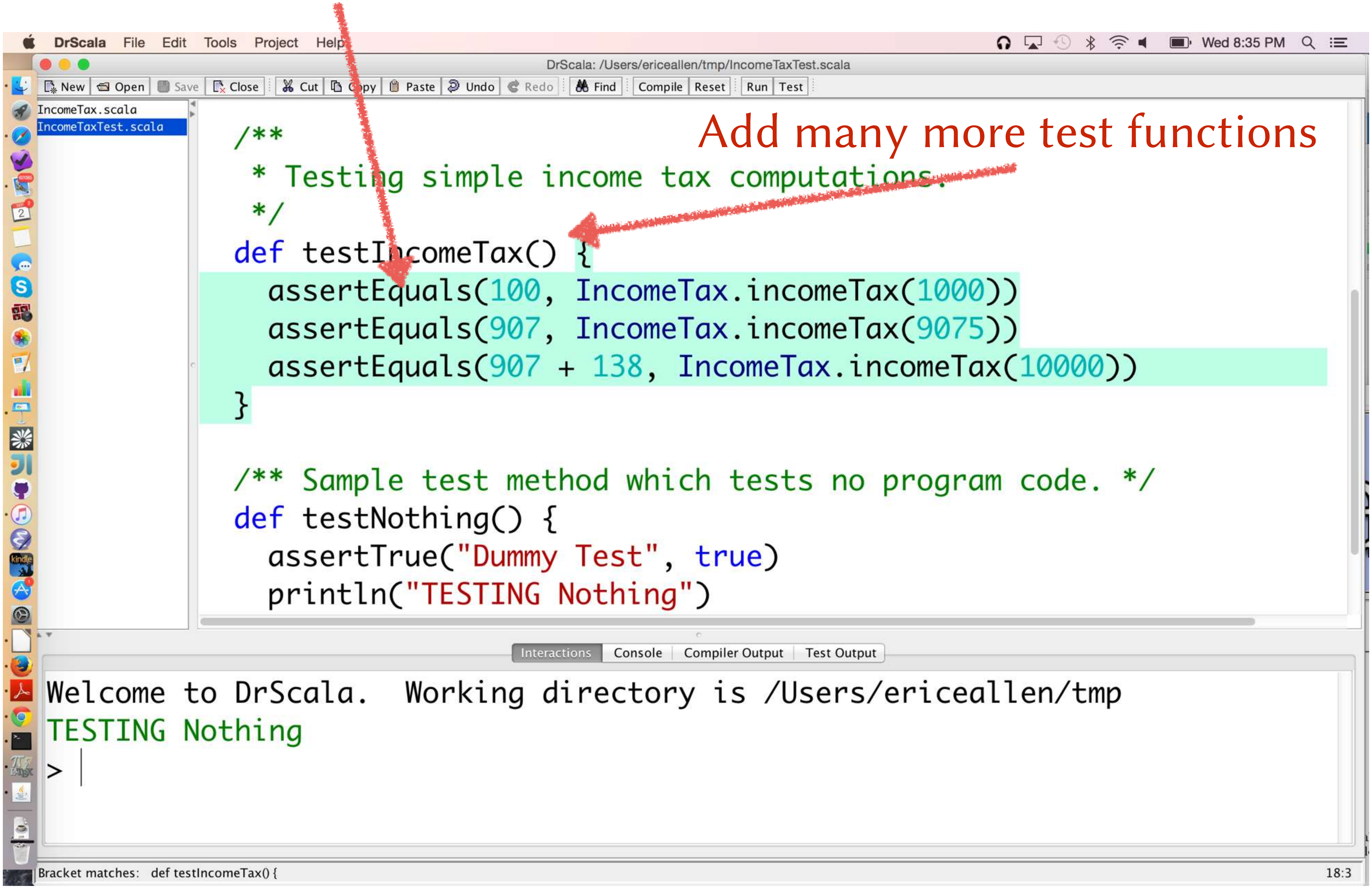


```
   * many "testSomething" methods in this class as you wish, and eac
   * one will be called when running JUnit over this class.)
   */
  def testX() {
  }

  /** Sample test method which tests no program code. */
  def testNothing() {
    assertTrue("Dummy Test", true)
    println("TESTING Nothing")
  }
}
```

The test fails if this argument does not reduce to true.

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
TESTING Nothing
>
```

assertEquals fails if its two arguments are not equal


DrScala File Edit Tools Project Help
Wed 8:35 PM
DrScala: /Users/ericeallen/tmp/IncomeTaxTest.scala
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test
IncomeTax.scala
IncomeTaxTest.scala
Interactions Console Compiler Output Test Output

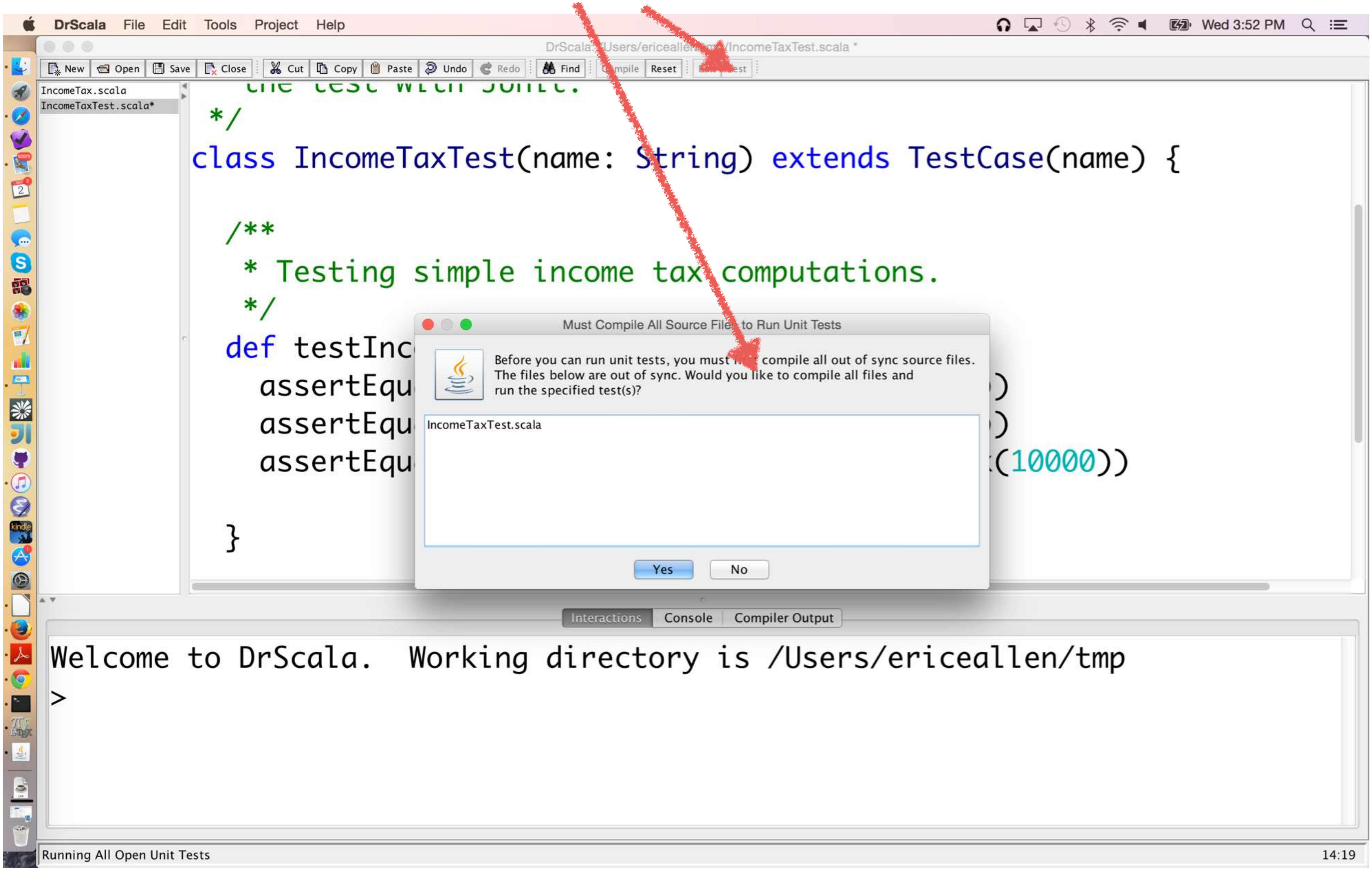
Add many more test functions

```
/**
 * Testing simple income tax computations.
 */
def testIncomeTax() {
  assertEquals(100, IncomeTax.incomeTax(1000))
  assertEquals(907, IncomeTax.incomeTax(9075))
  assertEquals(907 + 138, IncomeTax.incomeTax(10000))
}

/** Sample test method which tests no program code. */
def testNothing() {
  assertTrue("Dummy Test", true)
  println("TESTING Nothing")
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
TESTING Nothing
>
```

Bracket matches: def testIncomeTax() { 18:3

Hitting the Test button prompts us to compile


DrScala File Edit Tools Project Help
Wed 3:52 PM
DrScala: /Users/ericeallen/tmp/IncomeTaxTest.scala *
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test
IncomeTax.scala
IncomeTaxTest.scala*
Interactions Console Compiler Output

```
  the test with JUnit.
 */
class IncomeTaxTest(name: String) extends TestCase(name) {

  /**
   * Testing simple income tax computations.
   */
  def testInc
    assertEqu                                              )
    assertEqu                                              )
    assertEqu                                             (10000))

  }
```

Must Compile All Source Files to Run Unit Tests

Before you can run unit tests, you must first compile all out of sync source files. The files below are out of sync. Would you like to compile all files and run the specified test(s)?

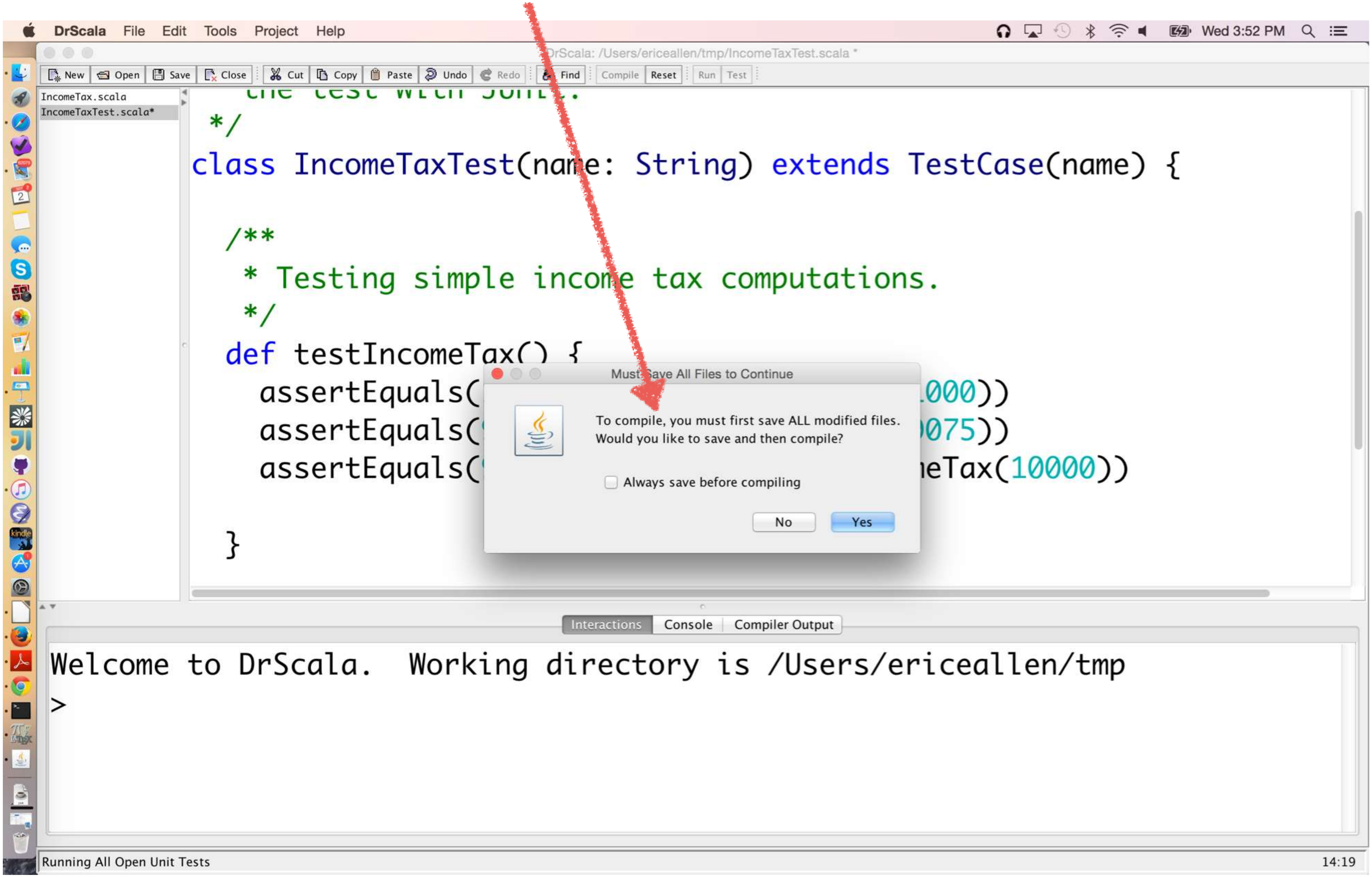
IncomeTaxTest.scala

Yes No

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
>
```

Running All Open Unit Tests 14:19

Agreeing to compile prompts us to save



```
  the test with JUnit.
*/
class IncomeTaxTest(name: String) extends TestCase(name) {

  /**
   * Testing simple income tax computations.
   */
  def testIncomeTax() {
    assertEquals(                               000))
    assertEquals(                               075))
    assertEquals(                            eTax(10000))

  }
```

Must Save All Files to Continue

To compile, you must first save ALL modified files.
Would you like to save and then compile?

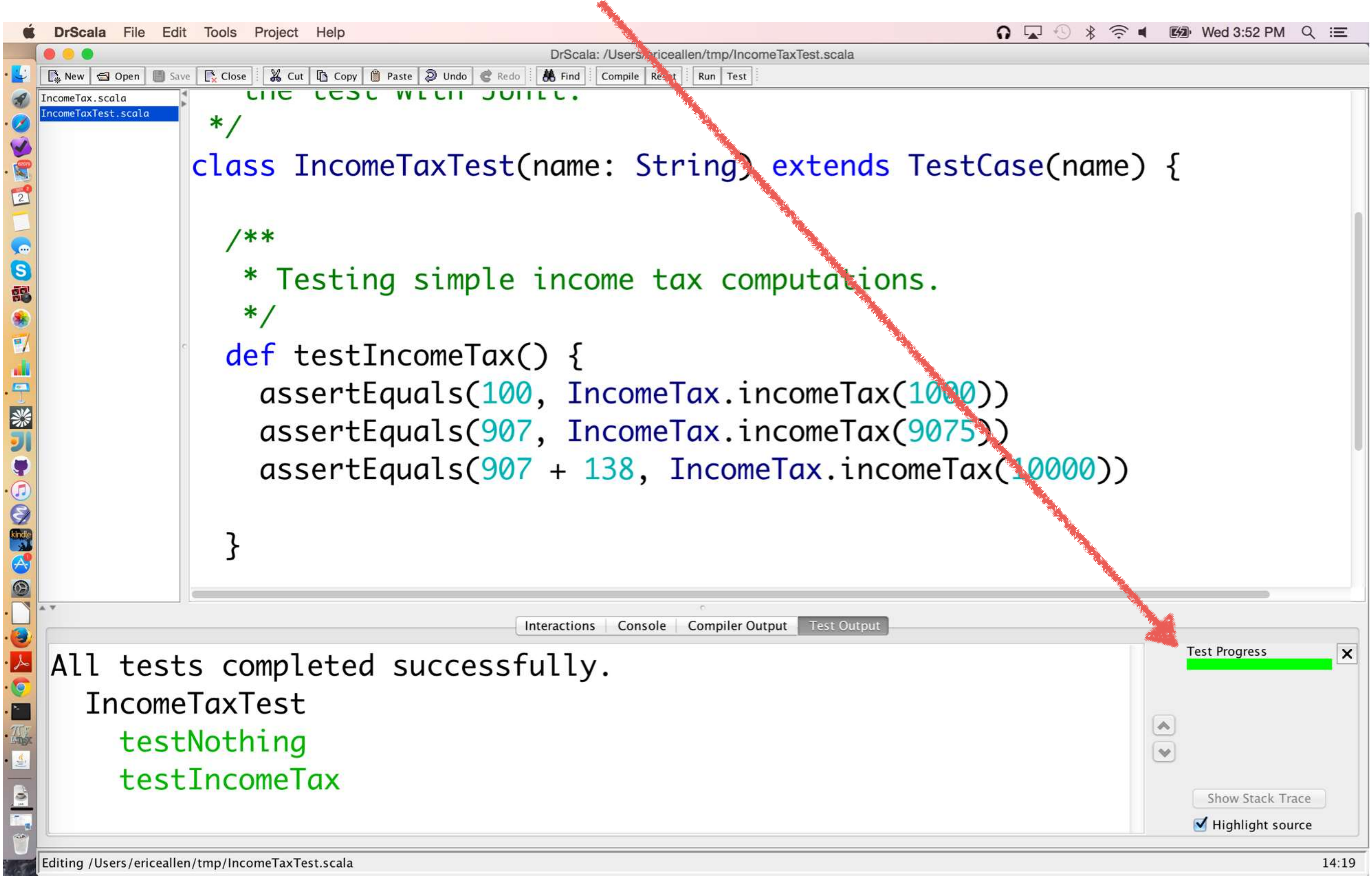
□ Always save before compiling

No Yes

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
>
```

Running All Open Unit Tests 14:19

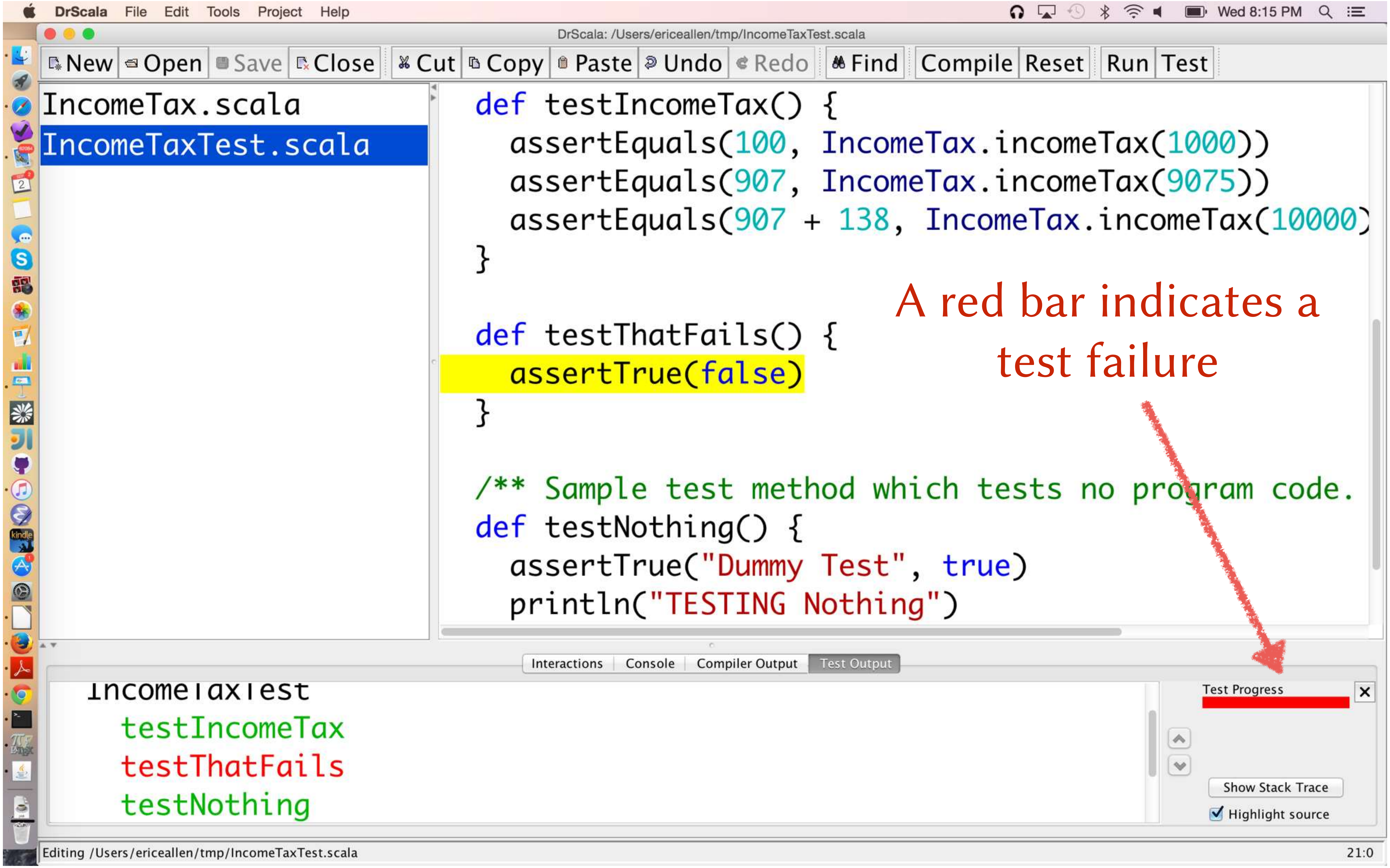
A green bar indicates that all tests passed



```
   the test with JUnit.
 */
class IncomeTaxTest(name: String) extends TestCase(name) {

  /**
   * Testing simple income tax computations.
   */
  def testIncomeTax() {
    assertEquals(100, IncomeTax.incomeTax(1000))
    assertEquals(907, IncomeTax.incomeTax(9075))
    assertEquals(907 + 138, IncomeTax.incomeTax(10000))

  }
```

```
All tests completed successfully.
  IncomeTaxTest
    testNothing
    testIncomeTax
```

DrScala File Edit Tools Project Help
Wed 3:52 PM
DrScala: /Users/ericeallen/tmp/IncomeTaxTest.scala
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test
IncomeTax.scala
IncomeTaxTest.scala
Interactions Console Compiler Output Test Output
Test Progress
Show Stack Trace
☑ Highlight source
Editing /Users/ericeallen/tmp/IncomeTaxTest.scala
14:19

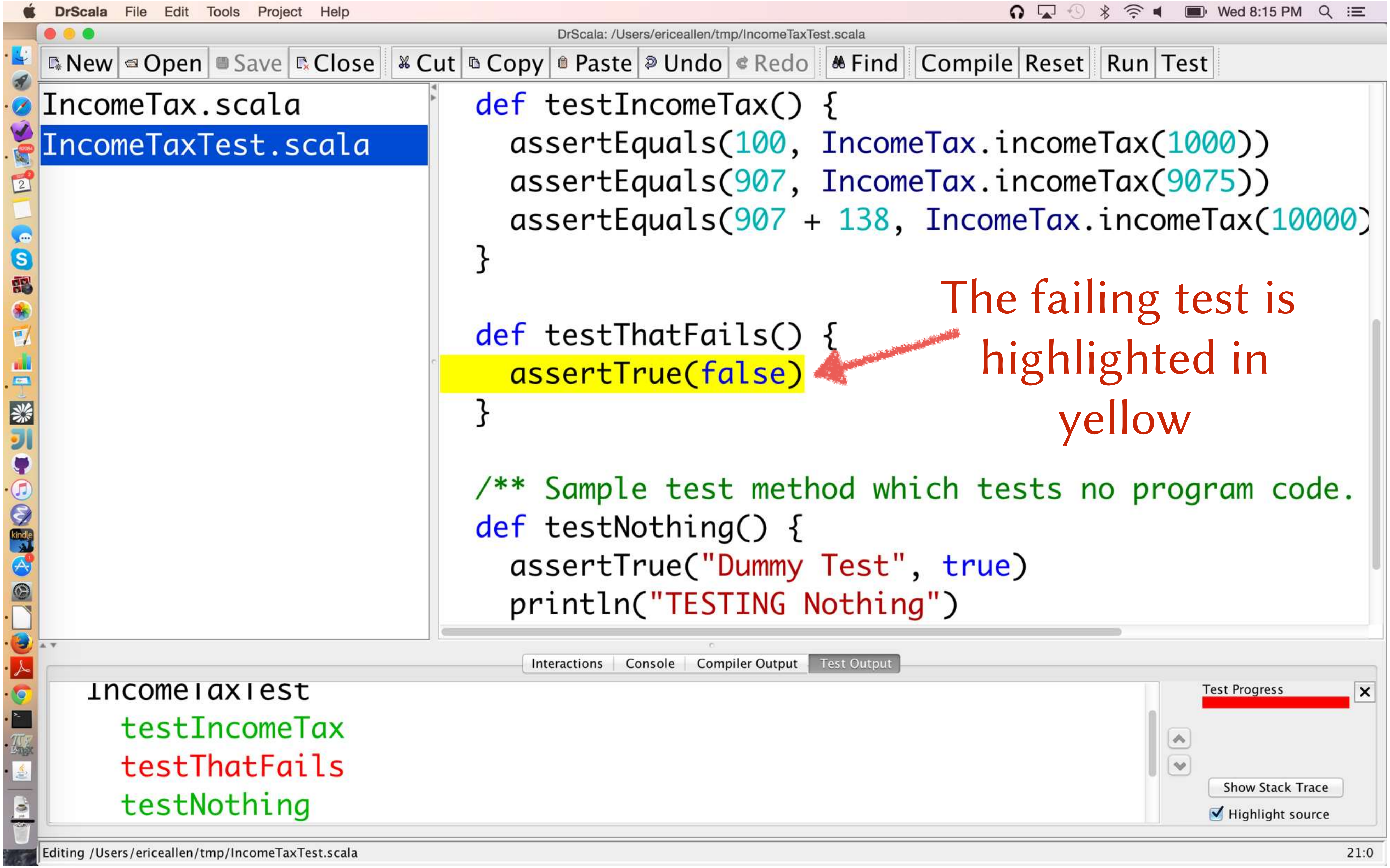


```
  def testIncomeTax() {
    assertEquals(100, IncomeTax.incomeTax(1000))
    assertEquals(907, IncomeTax.incomeTax(9075))
    assertEquals(907 + 138, IncomeTax.incomeTax(10000)
  }

  def testThatFails() {
    assertTrue(false)
  }

  /** Sample test method which tests no program code.
  def testNothing() {
    assertTrue("Dummy Test", true)
    println("TESTING Nothing")
```

```
IncomeTaxTest
  testIncomeTax
  testThatFails
  testNothing
```

Editing /Users/ericeallen/tmp/IncomeTaxTest.scala 21:0

A red bar indicates a test failure


DrScala File Edit Tools Project Help
Wed 8:15 PM
DrScala: /Users/ericeallen/tmp/IncomeTaxTest.scala
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test
IncomeTax.scala
IncomeTaxTest.scala
Interactions Console Compiler Output Test Output
Test Progress
Show Stack Trace
☑ Highlight source

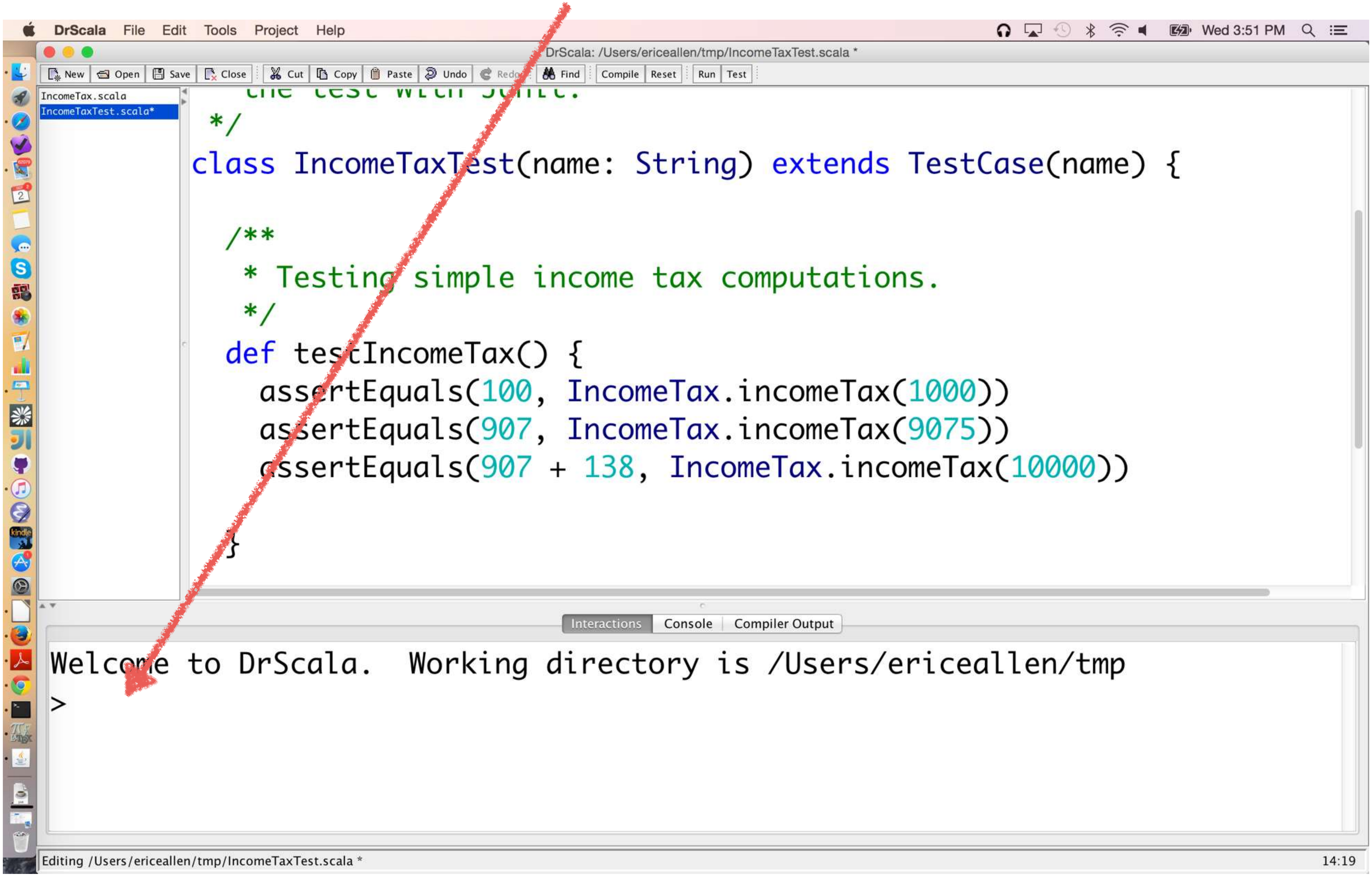
```
def testIncomeTax() {
  assertEquals(100, IncomeTax.incomeTax(1000))
  assertEquals(907, IncomeTax.incomeTax(9075))
  assertEquals(907 + 138, IncomeTax.incomeTax(10000)
}

def testThatFails() {
  assertTrue(false)
}

/** Sample test method which tests no program code.
def testNothing() {
  assertTrue("Dummy Test", true)
  println("TESTING Nothing")
```

The failing test is highlighted in yellow

```
IncomeTaxTest
  testIncomeTax
  testThatFails
  testNothing
```

Editing /Users/ericeallen/tmp/IncomeTaxTest.scala 21:0

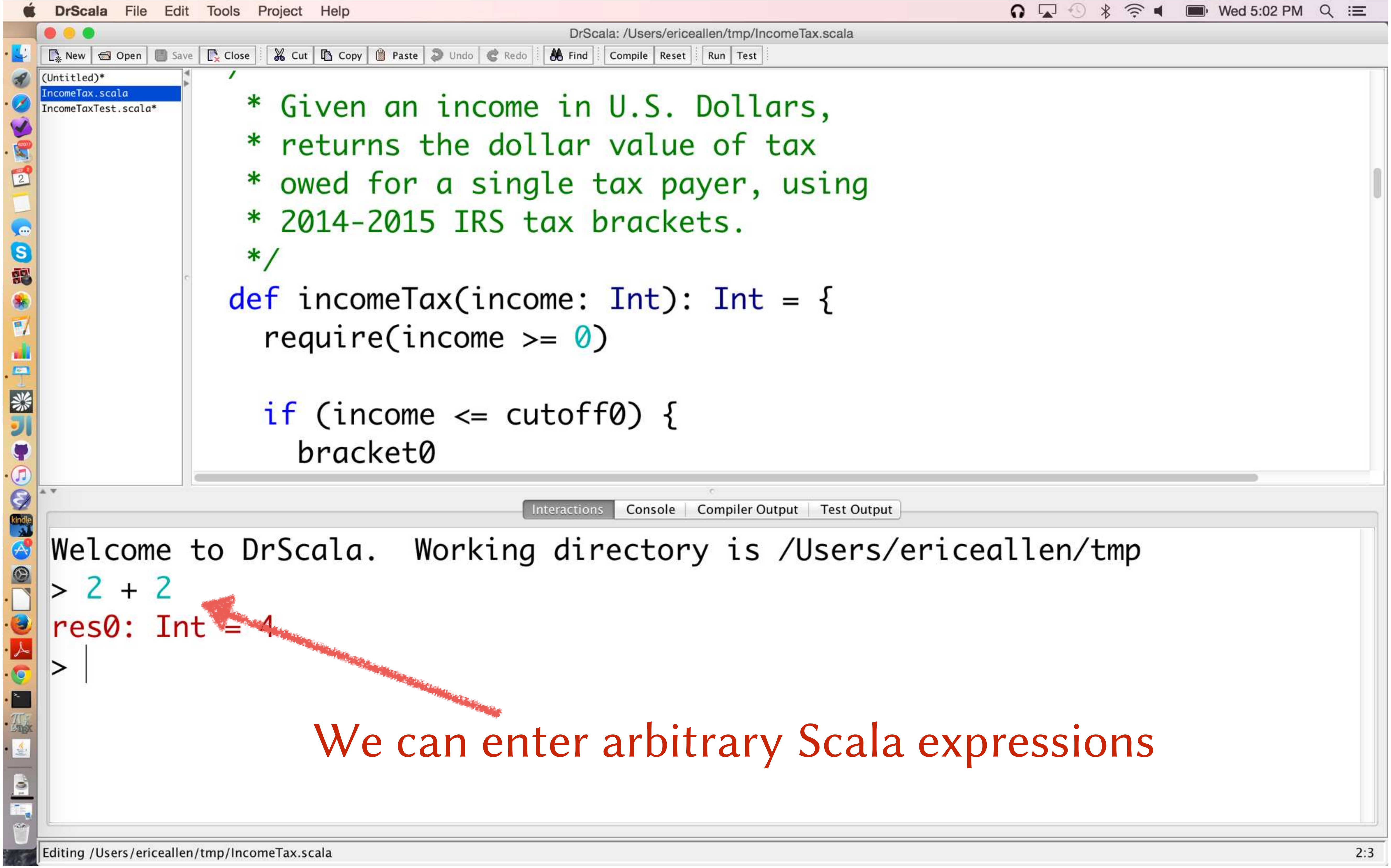
To interact with our program, we use the Interactions Pane



```
   the test with JUnit.
 */
class IncomeTaxTest(name: String) extends TestCase(name) {

  /**
   * Testing simple income tax computations.
   */
  def testIncomeTax() {
    assertEquals(100, IncomeTax.incomeTax(1000))
    assertEquals(907, IncomeTax.incomeTax(9075))
    assertEquals(907 + 138, IncomeTax.incomeTax(10000))

  }
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
>
```

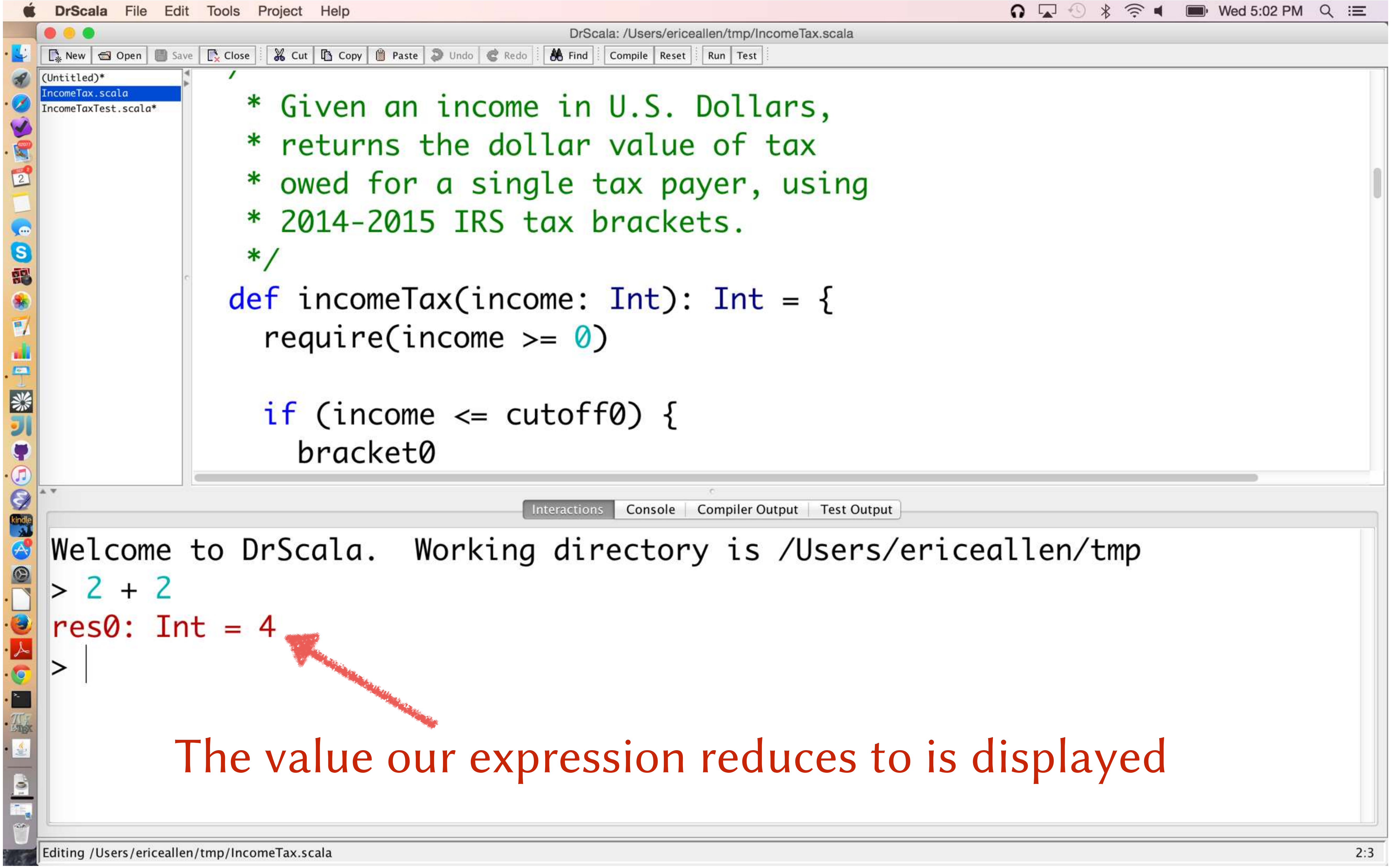
Editing /Users/ericeallen/tmp/IncomeTaxTest.scala * 14:19



```
 * Given an income in U.S. Dollars,
 * returns the dollar value of tax
 * owed for a single tax payer, using
 * 2014-2015 IRS tax brackets.
 */
def incomeTax(income: Int): Int = {
  require(income >= 0)

  if (income <= cutoff0) {
    bracket0
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
> 2 + 2
res0: Int = 4
>
```

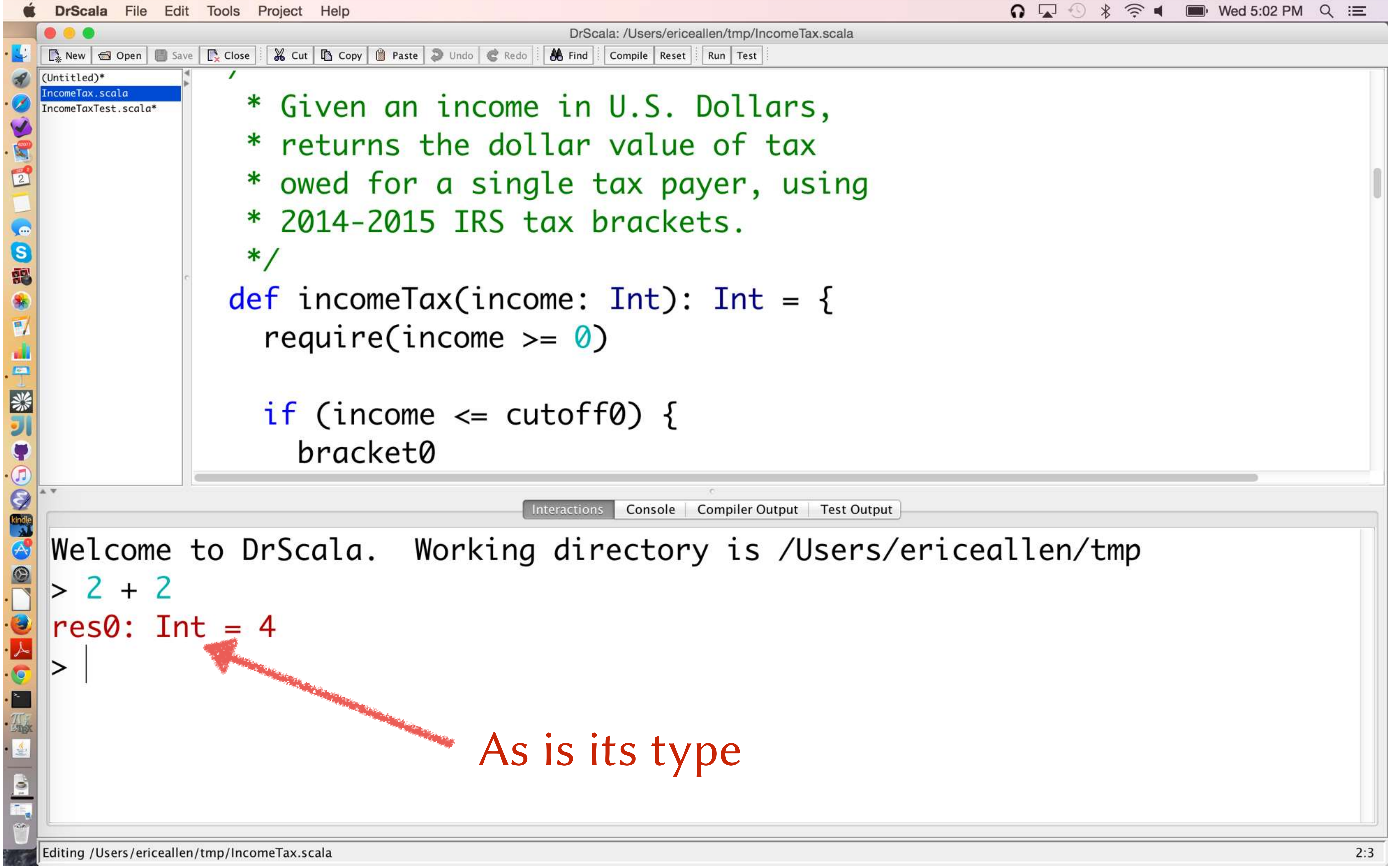
We can enter arbitrary Scala expressions



```
  /
   * Given an income in U.S. Dollars,
   * returns the dollar value of tax
   * owed for a single tax payer, using
   * 2014-2015 IRS tax brackets.
   */
def incomeTax(income: Int): Int = {
  require(income >= 0)

  if (income <= cutoff0) {
    bracket0
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
> 2 + 2
res0: Int = 4
>
```

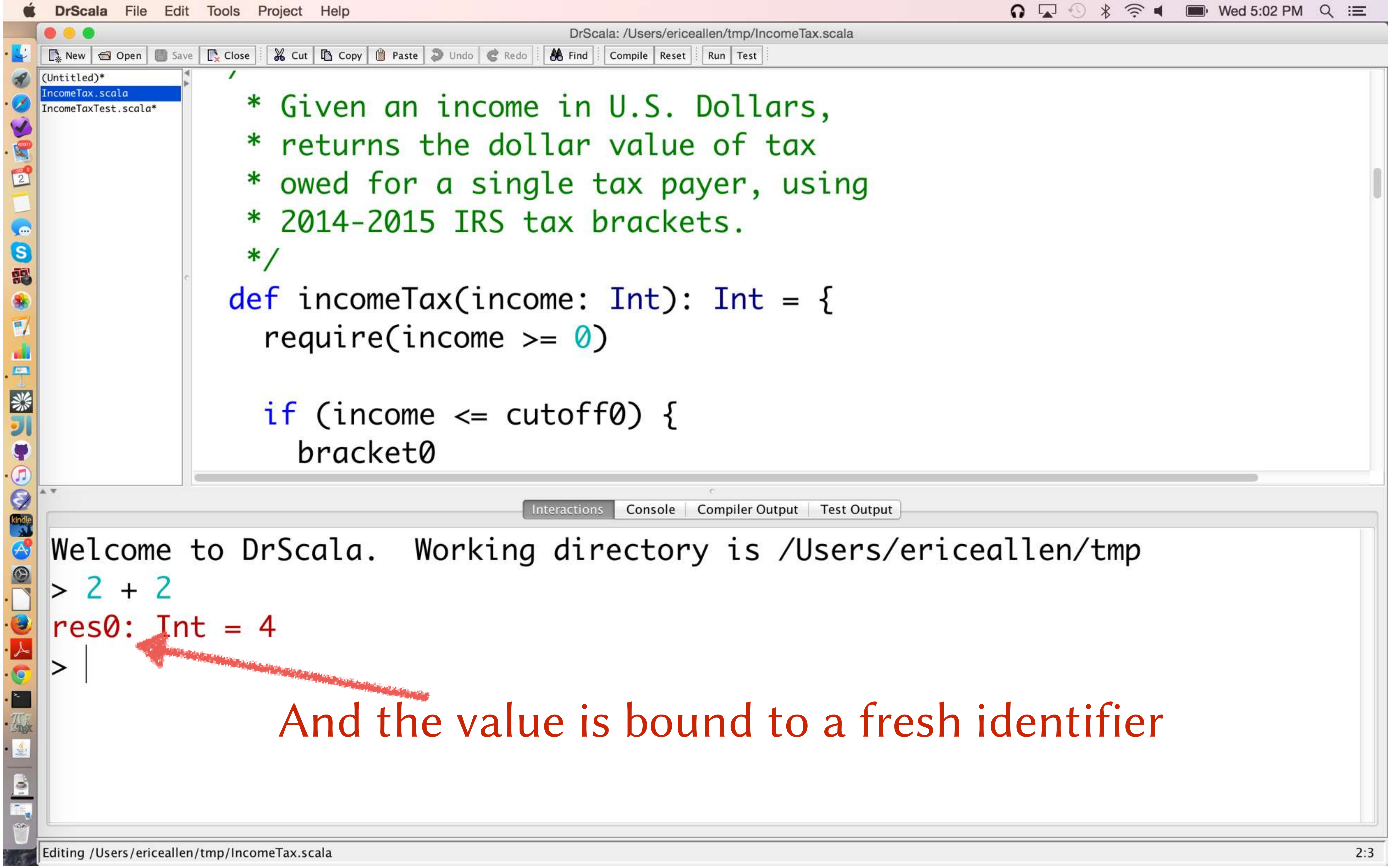
The value our expression reduces to is displayed



```
 /
  * Given an income in U.S. Dollars,
  * returns the dollar value of tax
  * owed for a single tax payer, using
  * 2014-2015 IRS tax brackets.
  */
def incomeTax(income: Int): Int = {
  require(income >= 0)

  if (income <= cutoff0) {
    bracket0
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
> 2 + 2
res0: Int = 4
> 
```

As is its type

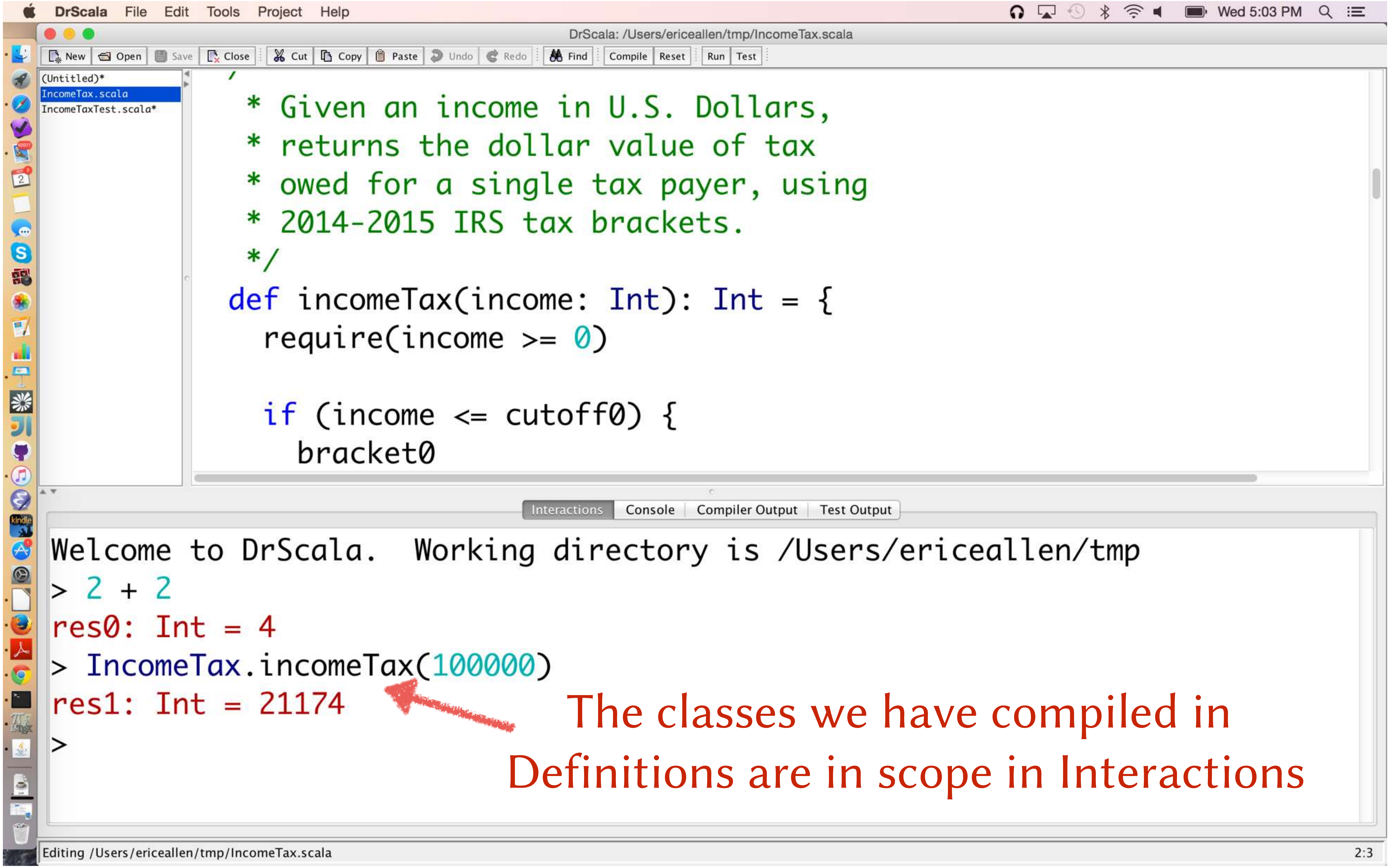


```
  /
   * Given an income in U.S. Dollars,
   * returns the dollar value of tax
   * owed for a single tax payer, using
   * 2014-2015 IRS tax brackets.
   */
  def incomeTax(income: Int): Int = {
    require(income >= 0)

    if (income <= cutoff0) {
      bracket0
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
> 2 + 2
res0: Int = 4
> 
```

Editing /Users/ericeallen/tmp/IncomeTax.scala

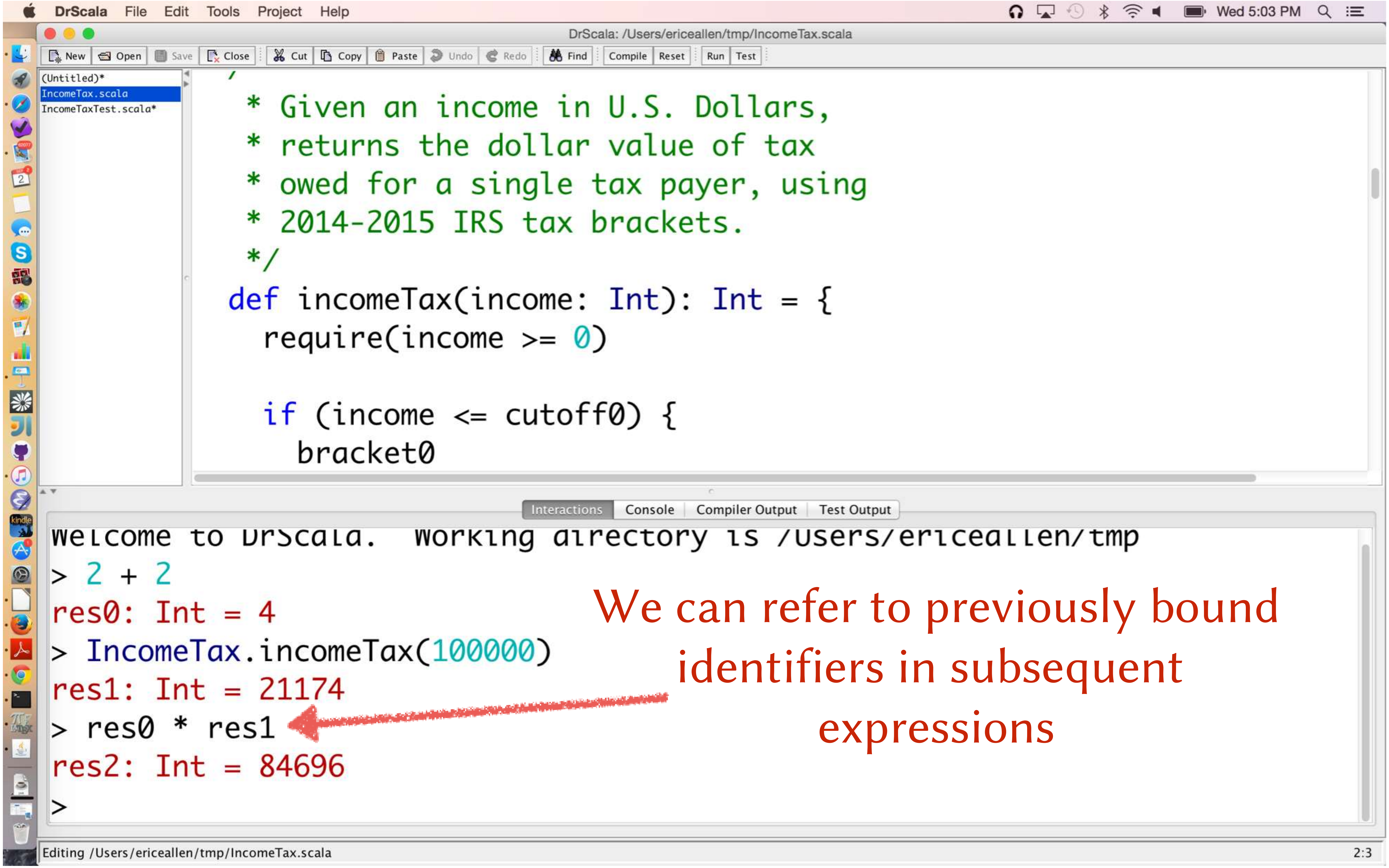
And the value is bound to a fresh identifier



```
  * Given an income in U.S. Dollars,
  * returns the dollar value of tax
  * owed for a single tax payer, using
  * 2014-2015 IRS tax brackets.
  */
def incomeTax(income: Int): Int = {
  require(income >= 0)

  if (income <= cutoff0) {
    bracket0
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
> 2 + 2
res0: Int = 4
> IncomeTax.incomeTax(100000)
res1: Int = 21174
>
```

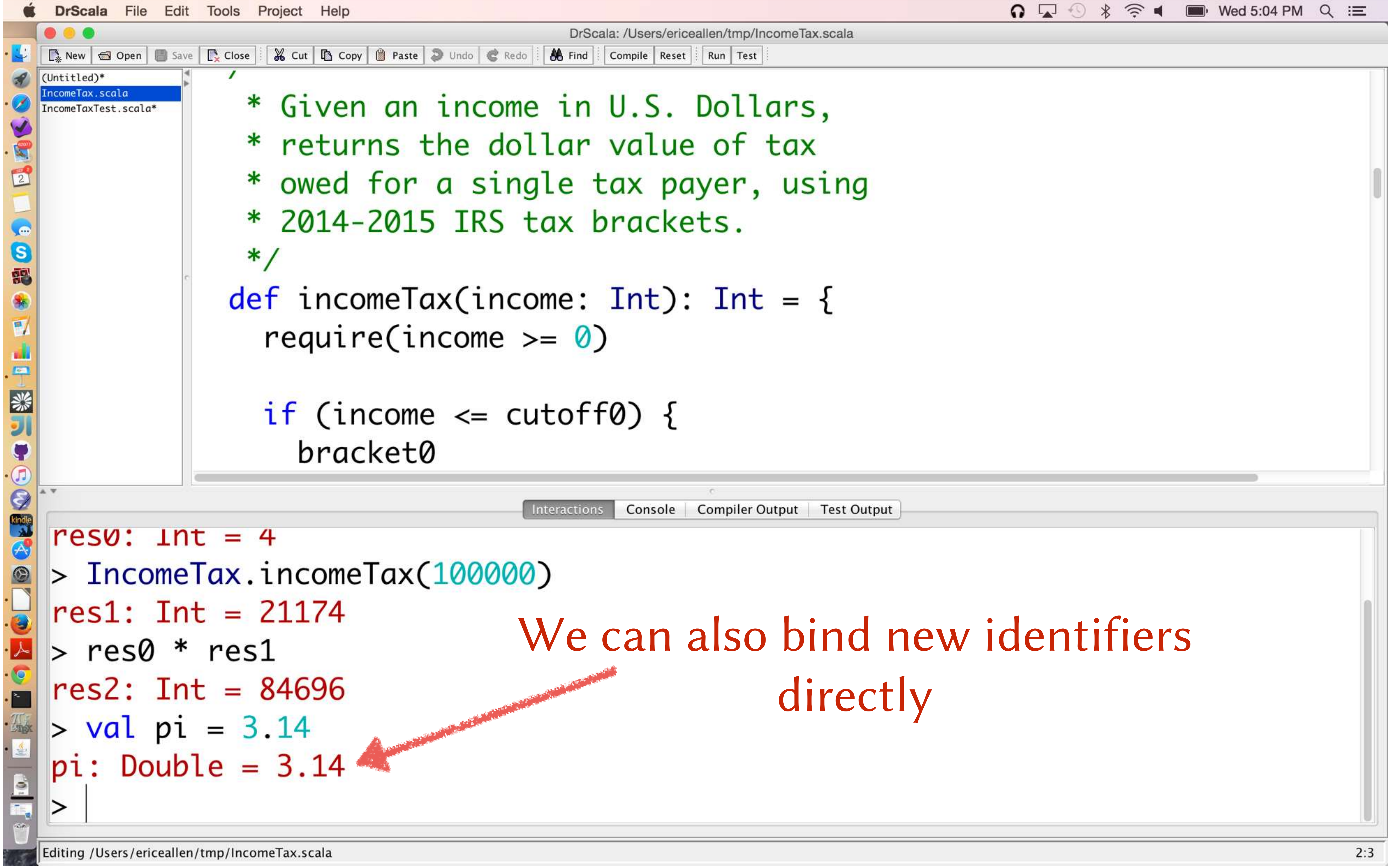
The classes we have compiled in Definitions are in scope in Interactions



```
/
  * Given an income in U.S. Dollars,
  * returns the dollar value of tax
  * owed for a single tax payer, using
  * 2014-2015 IRS tax brackets.
  */
def incomeTax(income: Int): Int = {
  require(income >= 0)

  if (income <= cutoff0) {
    bracket0
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
> 2 + 2
res0: Int = 4
> IncomeTax.incomeTax(100000)
res1: Int = 21174
> res0 * res1
res2: Int = 84696
>
```

We can refer to previously bound identifiers in subsequent expressions

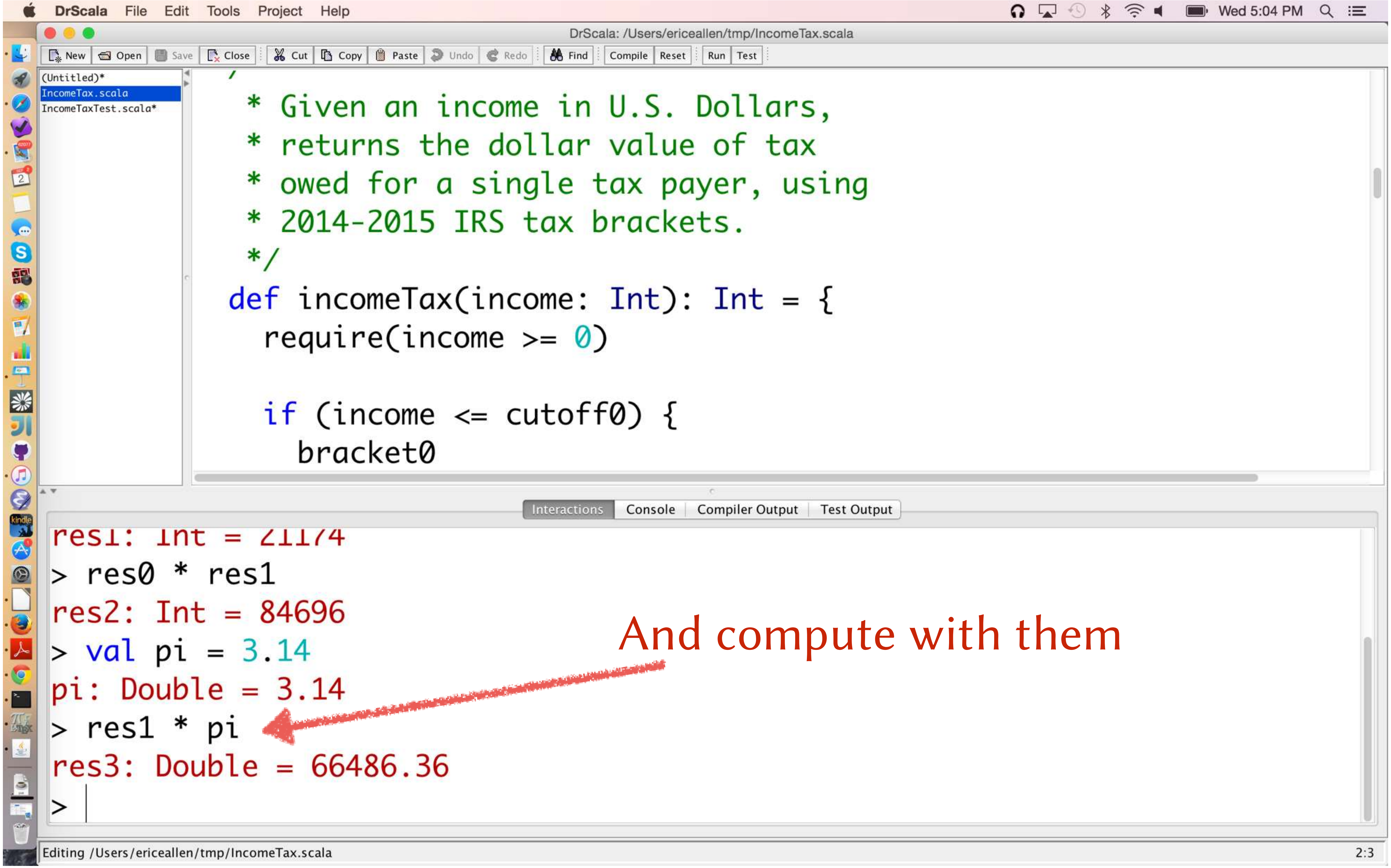


DrScala File Edit Tools Project Help
Wed 5:04 PM
DrScala: /Users/ericeallen/tmp/IncomeTax.scala
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test
(Untitled)*
IncomeTax.scala
IncomeTaxTest.scala*
Interactions Console Compiler Output Test Output
Editing /Users/ericeallen/tmp/IncomeTax.scala
2:3

```scala
  * Given an income in U.S. Dollars,
  * returns the dollar value of tax
  * owed for a single tax payer, using
  * 2014-2015 IRS tax brackets.
  */
def incomeTax(income: Int): Int = {
  require(income >= 0)

  if (income <= cutoff0) {
    bracket0
```

```
res0: Int = 4
> IncomeTax.incomeTax(100000)
res1: Int = 21174
> res0 * res1
res2: Int = 84696
> val pi = 3.14
pi: Double = 3.14
>
```

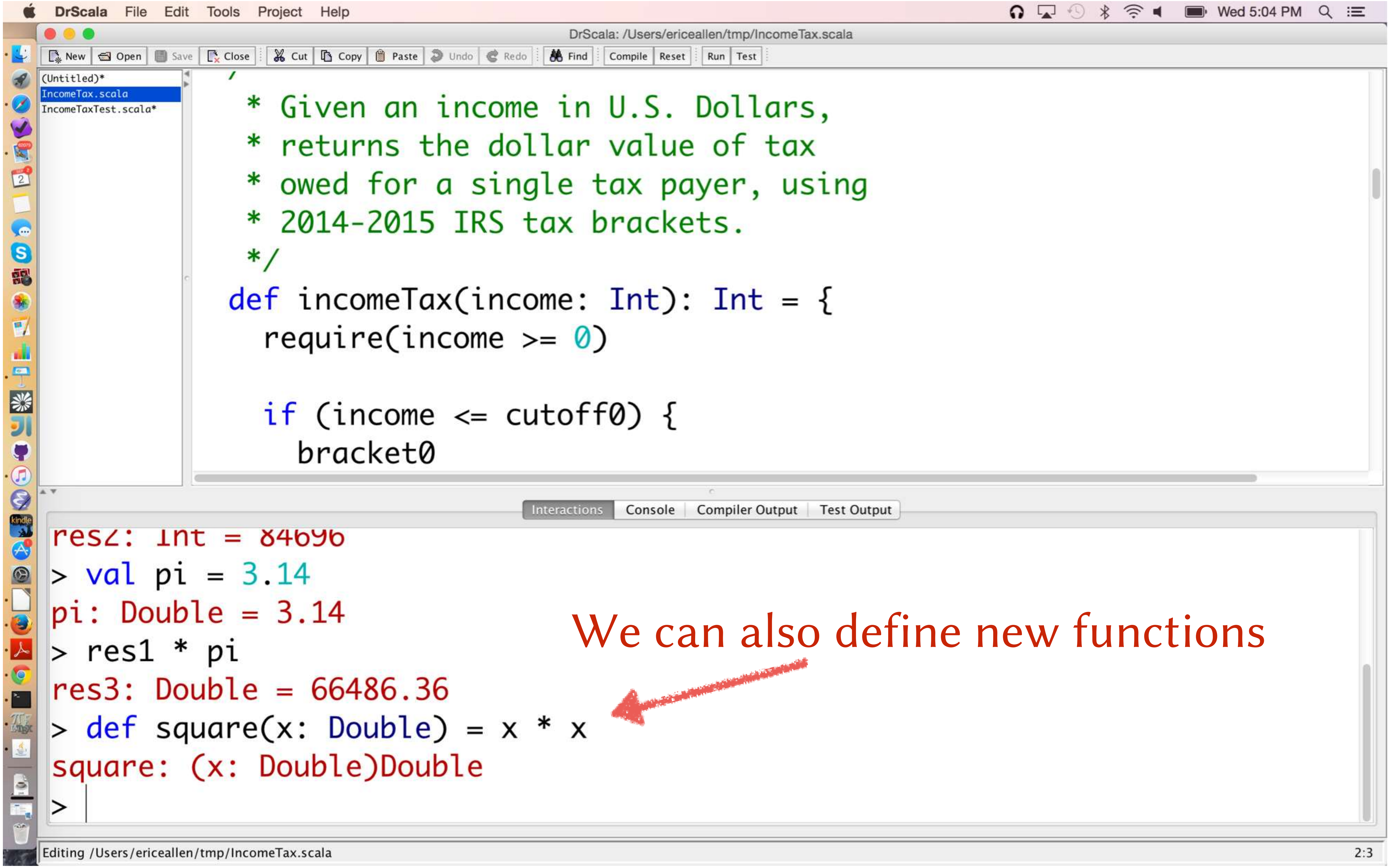
We can also bind new identifiers directly



```
/
 * Given an income in U.S. Dollars,
 * returns the dollar value of tax
 * owed for a single tax payer, using
 * 2014-2015 IRS tax brackets.
 */
def incomeTax(income: Int): Int = {
  require(income >= 0)

  if (income <= cutoff0) {
    bracket0
```

```
res1: Int = 21174
> res0 * res1
res2: Int = 84696
> val pi = 3.14
pi: Double = 3.14
> res1 * pi
res3: Double = 66486.36
>
```

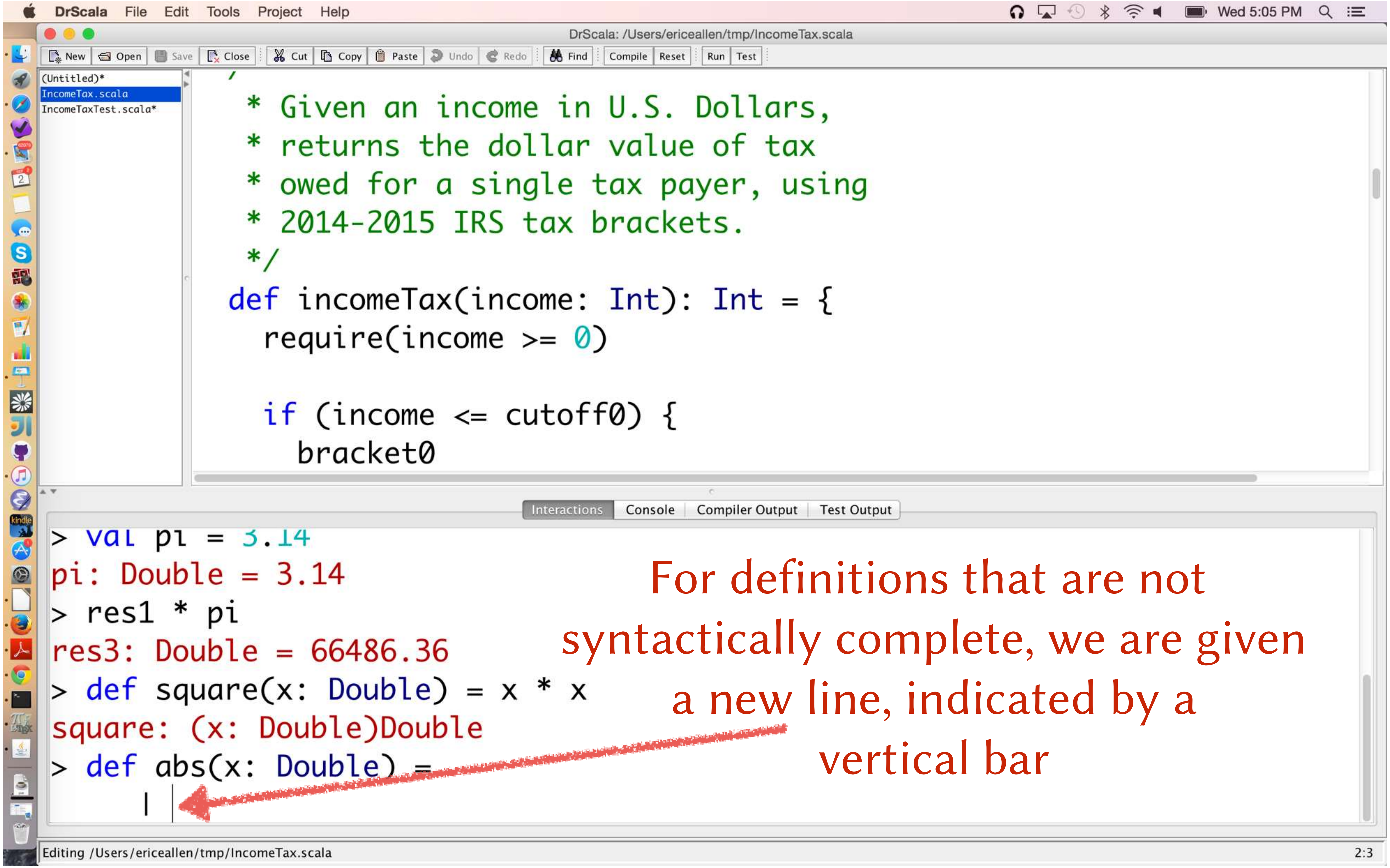
And compute with them



```
 /
  * Given an income in U.S. Dollars,
  * returns the dollar value of tax
  * owed for a single tax payer, using
  * 2014-2015 IRS tax brackets.
  */
def incomeTax(income: Int): Int = {
  require(income >= 0)

  if (income <= cutoff0) {
    bracket0
```

```
res2: Int = 84696
> val pi = 3.14
pi: Double = 3.14
> res1 * pi
res3: Double = 66486.36
> def square(x: Double) = x * x
square: (x: Double)Double
>
```

We can also define new functions

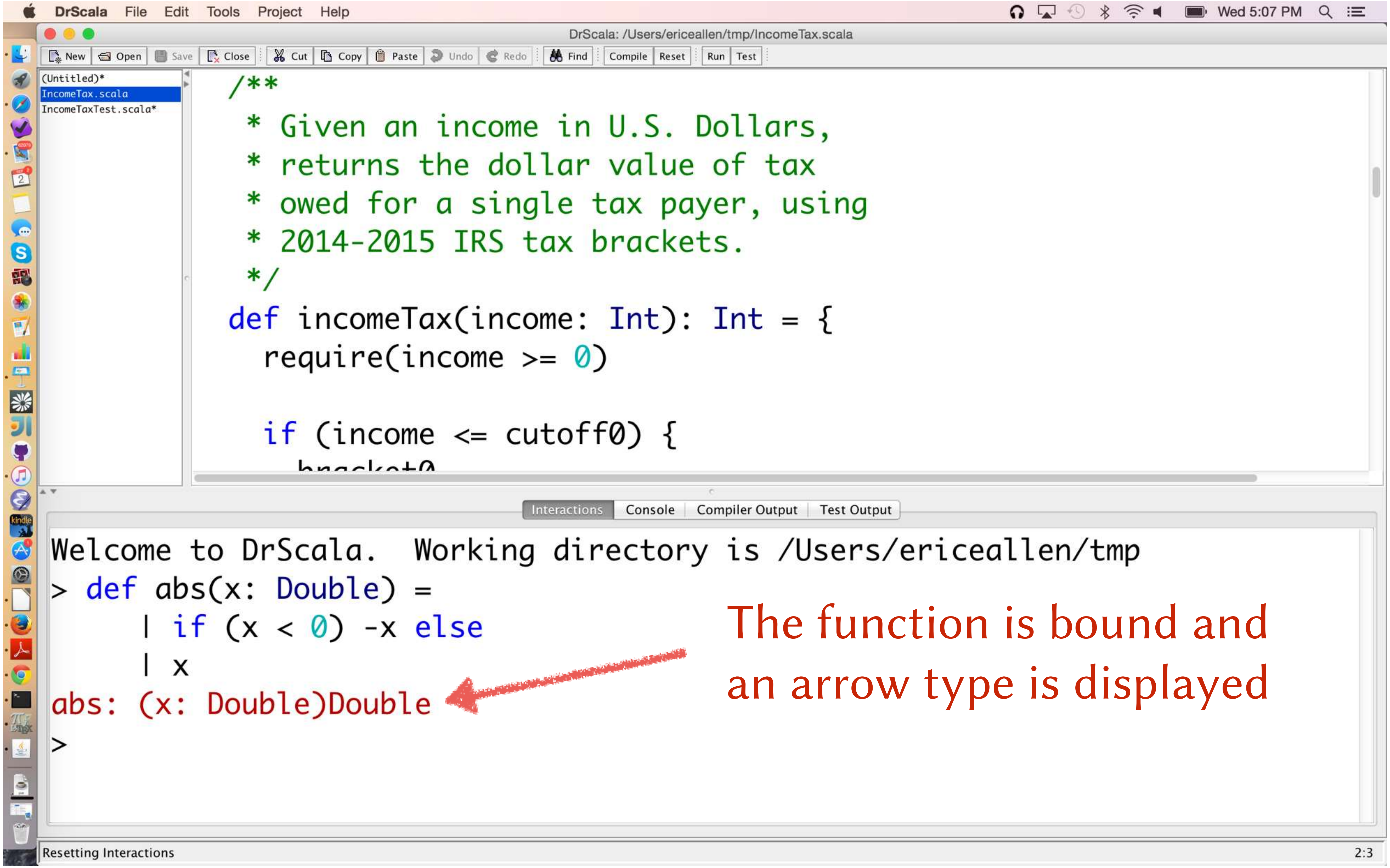


```
/
 * Given an income in U.S. Dollars,
 * returns the dollar value of tax
 * owed for a single tax payer, using
 * 2014-2015 IRS tax brackets.
 */
def incomeTax(income: Int): Int = {
  require(income >= 0)

  if (income <= cutoff0) {
    bracket0
```

```
> val pi = 3.14
pi: Double = 3.14
> res1 * pi
res3: Double = 66486.36
> def square(x: Double) = x * x
square: (x: Double)Double
> def abs(x: Double) =
      | |
```

Editing /Users/ericeallen/tmp/IncomeTax.scala 2:3

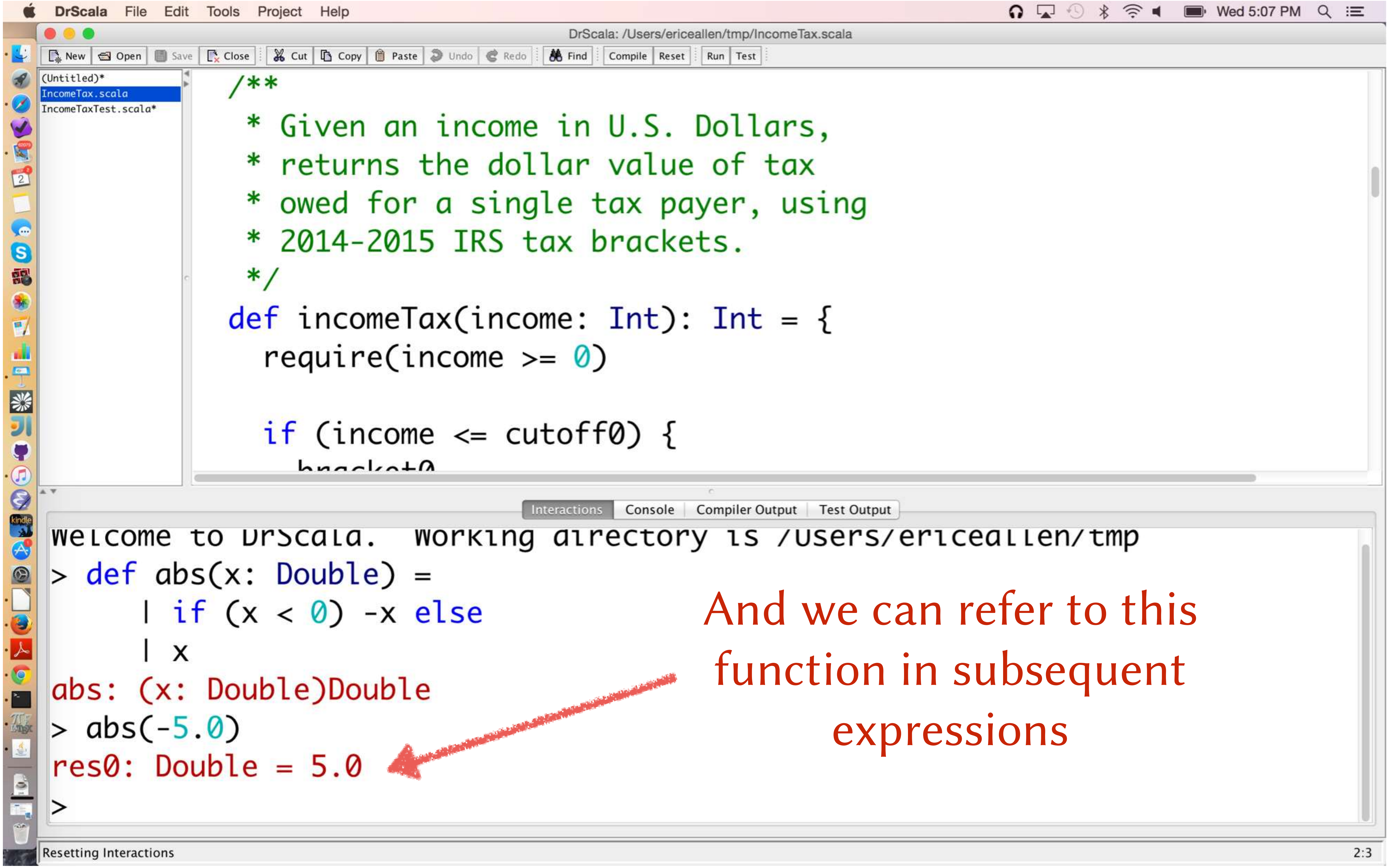
For definitions that are not syntactically complete, we are given a new line, indicated by a vertical bar


DrScala File Edit Tools Project Help
Wed 5:07 PM
DrScala: /Users/ericeallen/tmp/IncomeTax.scala
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test
(Untitled)*
IncomeTax.scala
IncomeTaxTest.scala*
Interactions Console Compiler Output Test Output
Resetting Interactions
2:3

```
/**
 * Given an income in U.S. Dollars,
 * returns the dollar value of tax
 * owed for a single tax payer, using
 * 2014-2015 IRS tax brackets.
 */
def incomeTax(income: Int): Int = {
  require(income >= 0)

  if (income <= cutoff0) {
    bracket0
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
> def abs(x: Double) =
     | if (x < 0) -x else
     | x
abs: (x: Double)Double
>
```

The function is bound and an arrow type is displayed

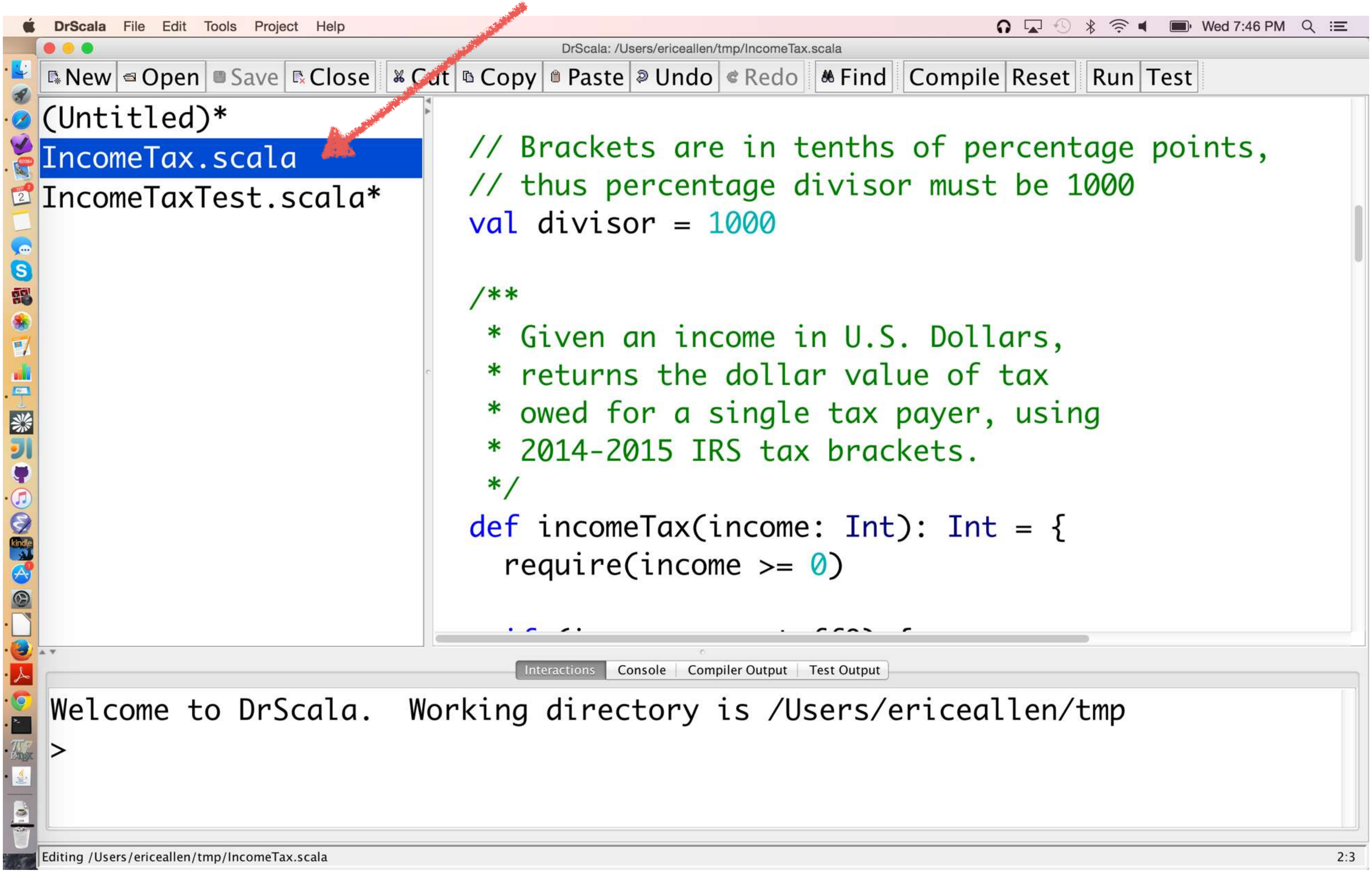

DrScala File Edit Tools Project Help
Wed 5:07 PM
DrScala: /Users/ericeallen/tmp/IncomeTax.scala
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test
(Untitled)*
IncomeTax.scala
IncomeTaxTest.scala*
Interactions Console Compiler Output Test Output
Resetting Interactions
2:3

```
/**
 * Given an income in U.S. Dollars,
 * returns the dollar value of tax
 * owed for a single tax payer, using
 * 2014-2015 IRS tax brackets.
 */
def incomeTax(income: Int): Int = {
  require(income >= 0)

  if (income <= cutoff0) {
    bracket0
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
> def abs(x: Double) =
     | if (x < 0) -x else
     | x
abs: (x: Double)Double
> abs(-5.0)
res0: Double = 5.0
>
```

And we can refer to this function in subsequent expressions

We can click on the file to appear in Definitions

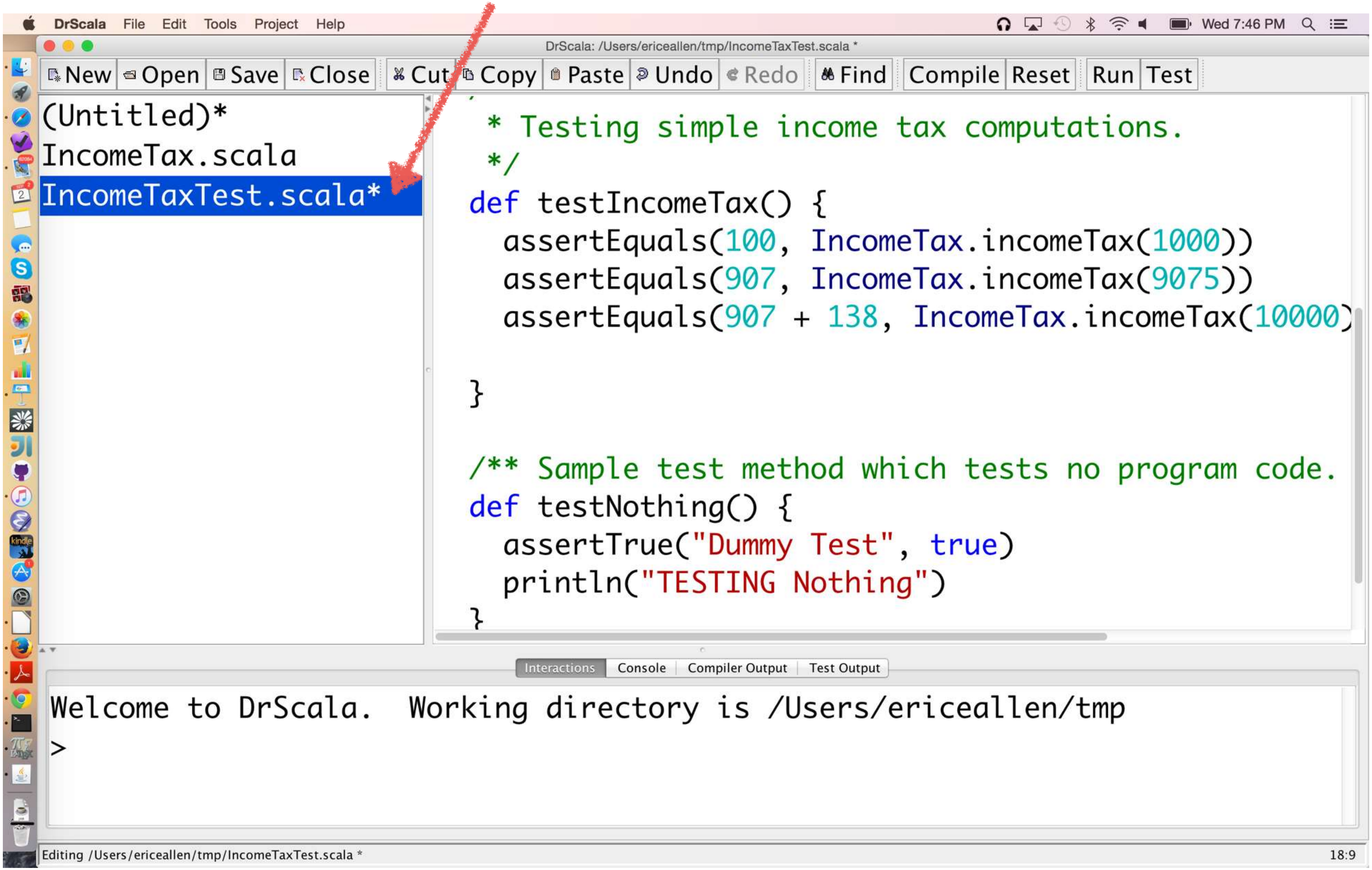


```
// Brackets are in tenths of percentage points,
// thus percentage divisor must be 1000
val divisor = 1000

/**
 * Given an income in U.S. Dollars,
 * returns the dollar value of tax
 * owed for a single tax payer, using
 * 2014-2015 IRS tax brackets.
 */
def incomeTax(income: Int): Int = {
  require(income >= 0)
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
>
```

Editing /Users/ericeallen/tmp/IncomeTax.scala 2:3

Files that have not been saved include an asterisk

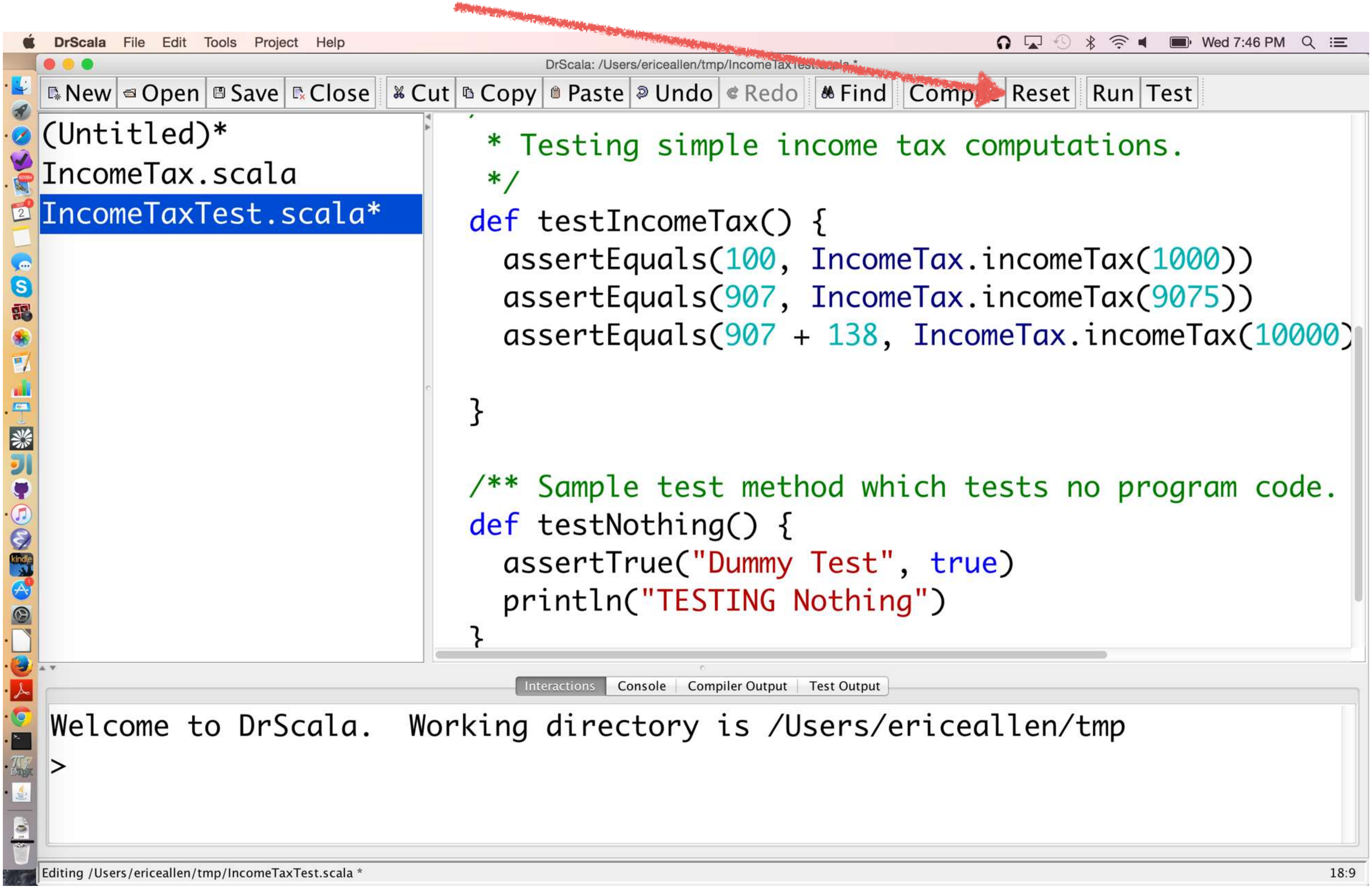


```
  * Testing simple income tax computations.
  */
def testIncomeTax() {
  assertEquals(100, IncomeTax.incomeTax(1000))
  assertEquals(907, IncomeTax.incomeTax(9075))
  assertEquals(907 + 138, IncomeTax.incomeTax(10000)

}

/** Sample test method which tests no program code.
def testNothing() {
  assertTrue("Dummy Test", true)
  println("TESTING Nothing")
}
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
>
```

Editing /Users/ericeallen/tmp/IncomeTaxTest.scala * 18:9

Reset resets the Interactions session

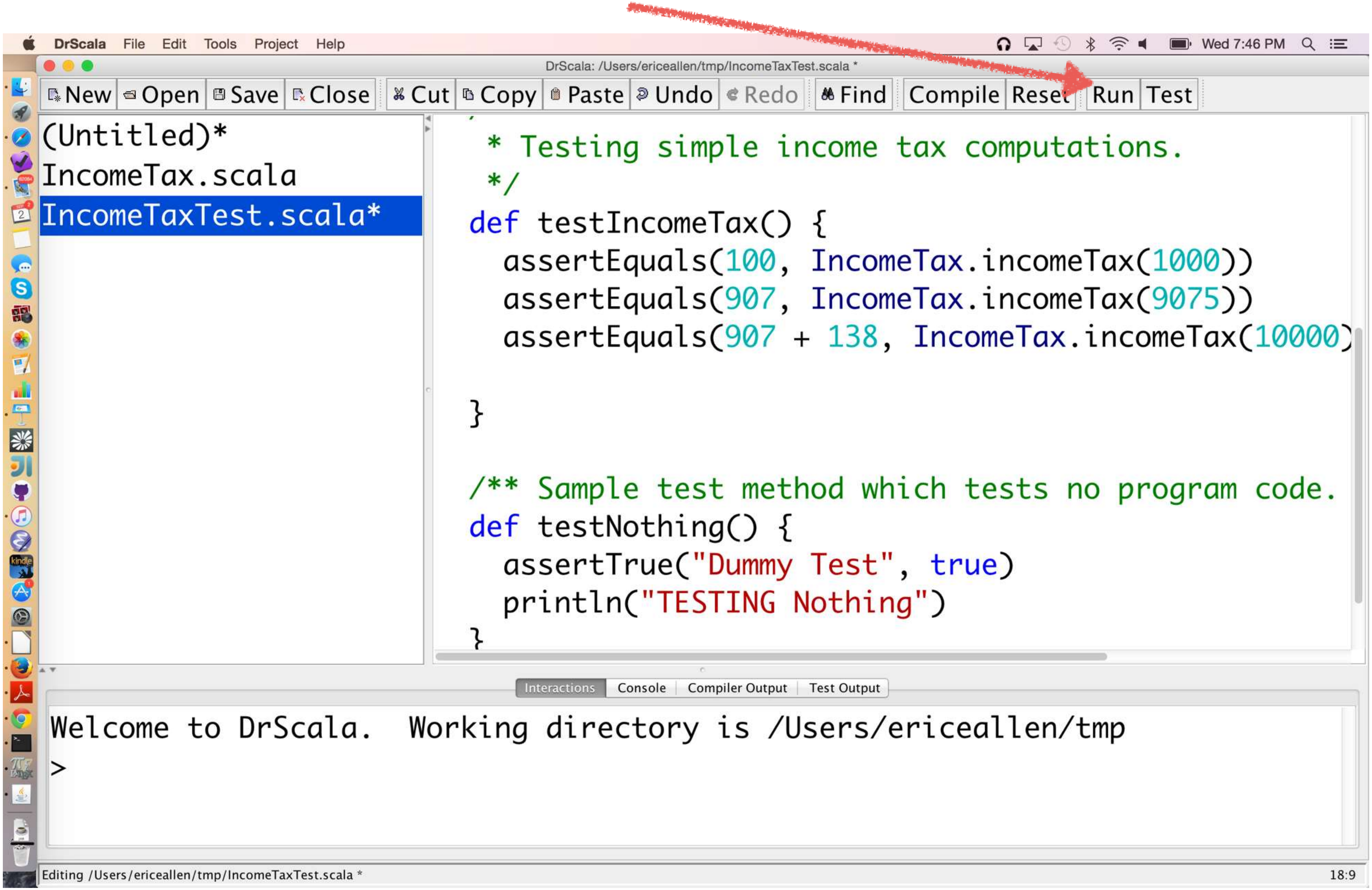


```
  * Testing simple income tax computations.
  */
def testIncomeTax() {
  assertEquals(100, IncomeTax.incomeTax(1000))
  assertEquals(907, IncomeTax.incomeTax(9075))
  assertEquals(907 + 138, IncomeTax.incomeTax(10000)

}

/** Sample test method which tests no program code.
def testNothing() {
  assertTrue("Dummy Test", true)
  println("TESTING Nothing")
}
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
>
```

Editing /Users/ericeallen/tmp/IncomeTaxTest.scala *   18:9

Run executes Definitions



```
  * Testing simple income tax computations.
  */
def testIncomeTax() {
  assertEquals(100, IncomeTax.incomeTax(1000))
  assertEquals(907, IncomeTax.incomeTax(9075))
  assertEquals(907 + 138, IncomeTax.incomeTax(10000)

}

/** Sample test method which tests no program code.
def testNothing() {
  assertTrue("Dummy Test", true)
  println("TESTING Nothing")
}
```

```
Welcome to DrScala.  Working directory is /Users/ericeallen/tmp
>
```

Editing /Users/ericeallen/tmp/IncomeTaxTest.scala *   18:9