

計算機科学第一（講義）

単体テスト， テスト駆動開発

脇田建

講義資料の入手方法

- 先週、作成した lecture ディレクトリ (build.sbtというファイルがあるところ) に移動してから以下のコマンドを実行

git pull

小テスト

テスト駆動開発

Test Driven Development (TDD)

テスト駆動開発の実際

- ✿ ひとまず、やる気のないコードを作成
- ✿ テストを実施するコードを作成（完璧でなくてよい）
- ✿ 以下を繰り返し
 - ✿ テストを実行
 - ✿ テストに合格するようにプログラムを修正
 - ✿ 想定外のバグを発見 → バグを再現するテストを追加

例：閏年の計算

- * 目標：西暦(Y)が与えられたときに、その年が閏年か否かを答えるメソッドleapYearを作成しなさい。

日本における閏年の根拠法

明治三十一年勅令第九十号

- 明治三十一年勅令第九十号（閏年ニ関スル件・明治三十一年五月十一日勅令第九十号）
- 神武天皇即位紀元年数ノ四ヲ以テ整除シ得ヘキ年ヲ閏年トス
- 但シ紀元年数ヨリ六百六十ヲ減シテ百ヲ以テ整除シ得ヘキモノノ中更ニ四ヲ以テ商ヲ整除シ得サル年ハ平年トス

日本における閏年の根拠法

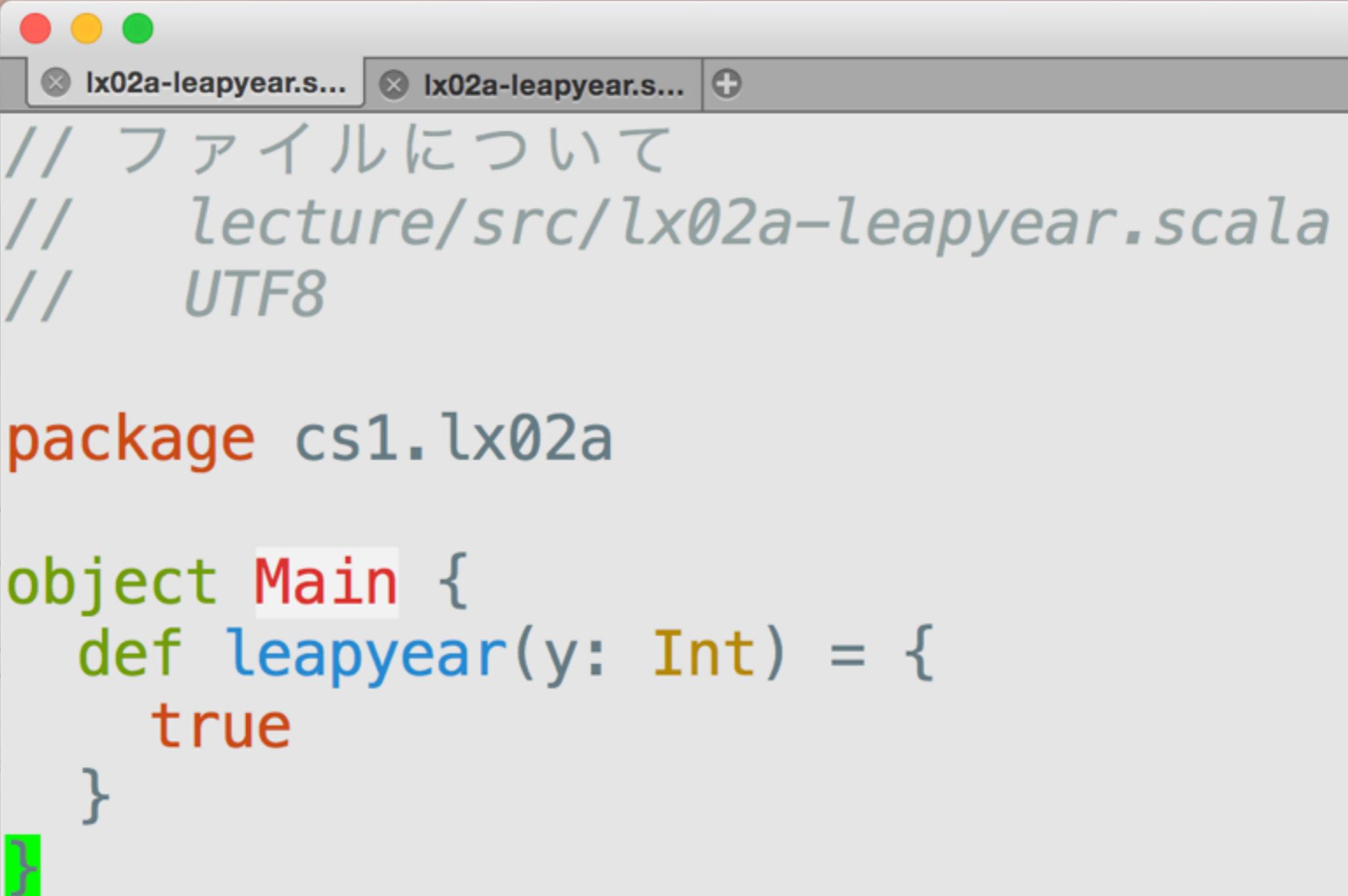
明治三十一年勅令第九十号

- ❖ 明治三十一年勅令第九十号（閏年ニ関スル件・明治三十一年五月十一日勅令第九十号）
 - ❖ 神武天皇が即位なさった年を紀元とする年数¹（これが紀元年数）が四で割り切れるものを閏年とする
 - ❖ ただし、紀元年数から 660 を減じたもの²が 100 で割り切れるもののうち、さらにその商が 4 で割り切れないもの³は平年とする。
 - ❖ *¹ - これが紀元年数
 - ❖ *² - 神武天皇の即位の年は西暦（-660）年とされている
 - ❖ *³ - つまり、西暦換算が 400 で割り切れないものについて語っている

早い話が、

- ✿ グレゴリオ暦では、次の規則に従って400年間に（100回ではなく）97回の閏年を設ける。
- ✿ 西暦年が4で割り切れる年は閏年
- ✿ ただし、西暦年が100で割り切れる年は平年
- ✿ ただし、西暦年が400で割り切れる年は閏年

ステップ1：やる気のないコードとして、これ以上はないほど愚かなコードを作る。型だけは仕様に合わせる。



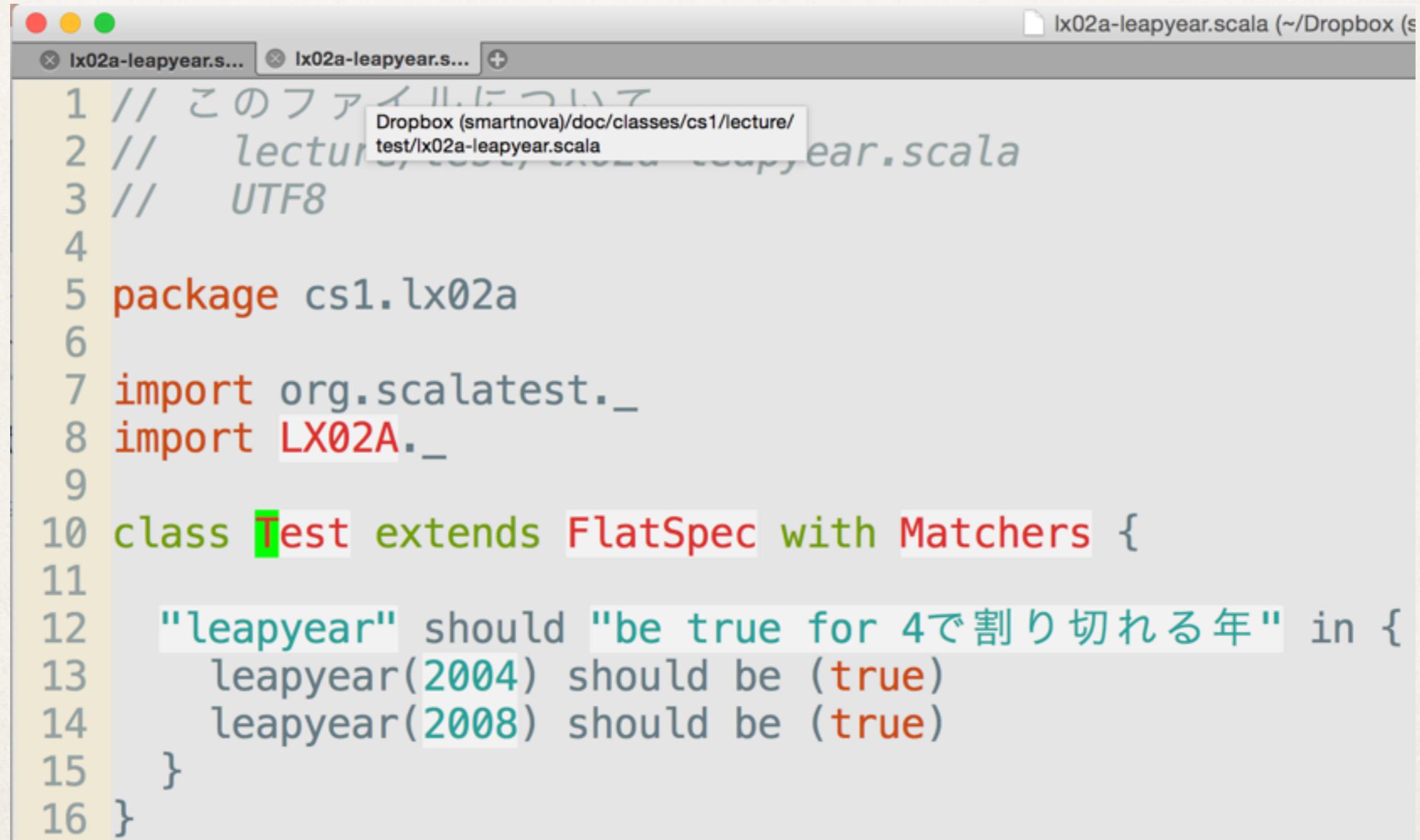
```
// ファイルについて
// lecture/src/lx02a-leapyear.scala
// UTF8

package cs1.lx02a

object Main {
    def leapyear(y: Int) = {
        true
    }
}
```

ステップ2: テストのためのコードを作成

完璧でなくてよい



A screenshot of a Mac OS X desktop showing a code editor window. The window title is "lx02a-leapyear.scala (~/Dropbox (s...)" and it contains the following Scala code:

```
1 // このファイルについて
2 // lecture, exercise, etc., ..., leapyear.scala
3 // UTF8
4
5 package cs1.lx02a
6
7 import org.scalatest._
8 import LX02A._
9
10 class Test extends FlatSpec with Matchers {
11
12   "leapyear" should "be true for 4で割り切れる年" in {
13     leapyear(2004) should be (true)
14     leapyear(2008) should be (true)
15   }
16 }
```

The code is a Scala test specification using Scalatest. It defines a class `Test` that extends `FlatSpec` and `Matchers`. It contains one test method, `leapyear`, which checks if leapyears (2004 and 2008) return `true` when passed to the `leapyear` function from the `LX02A` module.

ステップ2: テストのためのコードを作成

完璧でなくてよい

```
lx02a-leapyear.s... lx02a-leapyear.s...
1 // このファイルについて
2 // lecturer
3 // UTF8
4
5 package cs1.lx02a
6
7 import org.scalatest._
8 import Main._

9
10 class Test extends FlatSpec with Matchers {
11
12   "leapyear" should "be true for 4で割り切れる年" in {
13     leapyear(2004) should be (true)
14     leapyear(2008) should be (true)
15   }
16 }
```

テスト対象の object の名前
“import Main._” 宣言により,
Main.leapyear でなく単に
leapyear と参照できる

sbtコマンドを起動してテストを実行

Scala Build Tool

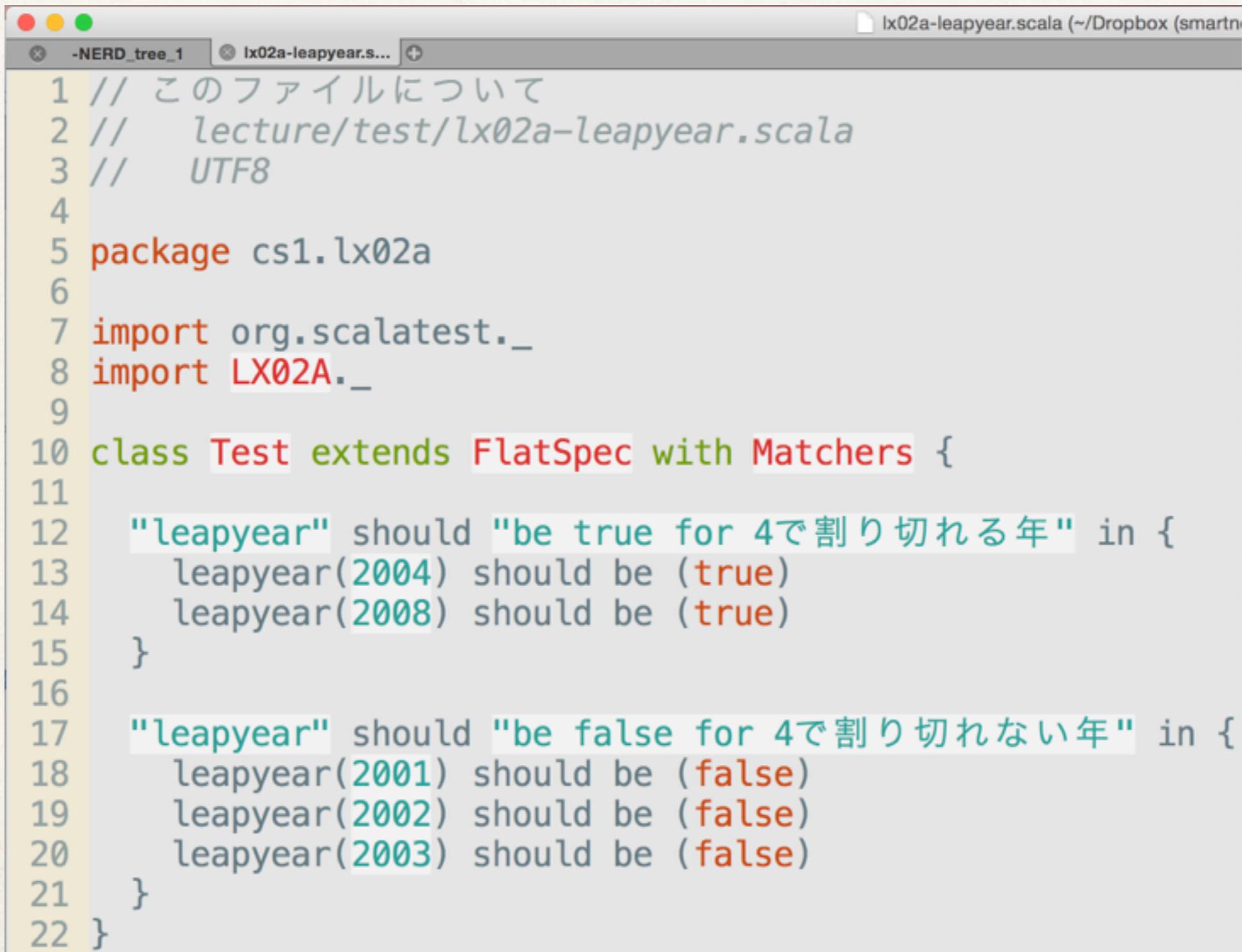
```
1. Default
> test
[info] Updating {file:/Users/wakita/Dropbox%20(smartnova)/doc/classes/cs1/lectur
e/}lecture...
[info] Resolving jline#jline;2.12.1 ...
[info] Done updating.
[info] Compiling 3 Scala sources to /Users/wakita/tmp/cs1f/scala-2.11/classes...
[info] Compiling 2 Scala sources to /Users/wakita/tmp/cs1f/scala-2.11/test-class
es...
[info] Test:
[info] leapyear
[info] - should be true for 4で割り切れる年
[info] LX02ATest:
[info] leapyear
[info] - should be true for 4で割り切れる年
[info] leapyear
[info] - should be false for 4で割り切れない年
[info] leapyear
[info] - should be false to 100で割り切れず
[info] Run completed in 525 milliseconds.
[info] Total number of tests run: 4
[info] Suites: completed 2, aborted 0
[info] Tests: succeeded 4, failed 0, canceled 0, ignored 0, pending 0
[info]
[success] Total time: 6 s, completed 2015/10/13 10:01:27
> |
```

全テストをパス。完璧！

と、喜んでいると、天の声

- ✿ 曰く「4で割り切れない年は平年」
- ✿ 「やべ、テストが甘い！追加しなくちゃ」

4で割り切れない年のテストを追加



A screenshot of a terminal window titled "lx02a-leapyear.scala (~/Dropbox (smartn...)" showing Scala test code. The code defines a class "Test" that extends "FlatSpec with Matchers". It contains two test cases: one for leap years (years divisible by 4) and one for non-leap years (years not divisible by 4). The non-leap year test includes comments in Japanese explaining the requirement.

```
1 // このファイルについて
2 //   lecture/test/lx02a-leapyear.scala
3 //   UTF8
4
5 package cs1.lx02a
6
7 import org.scalatest._
8 import LX02A._
9
10 class Test extends FlatSpec with Matchers {
11
12   "leapyear" should "be true for 4で割り切れる年" in {
13     leapyear(2004) should be (true)
14     leapyear(2008) should be (true)
15   }
16
17   "leapyear" should "be false for 4で割り切れない年" in {
18     leapyear(2001) should be (false)
19     leapyear(2002) should be (false)
20     leapyear(2003) should be (false)
21   }
22 }
```

再度テストを実行

```
1. Default

> test
[info] Test:
[info] leapyear
[info] - should be true for 4で割り切れる年
[info] leapyear
[info] - should be false for 4で割り切れない年 *** FAILED ***
[info]     true was not false (lx02a-leapyear.scala:18)
[info] Run completed in 477 milliseconds.

[info] Total number of tests run: 2
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 1, failed 1
[info] *** 1 TEST FAILED ***
[error] Failed tests:
[error]       cs1.lx02a.Test
[error] (test:test) sbt.TestsFailedException
[error] Total time: 1 s, completed
>
```

leapyearのテストで問題発見
テスト (lx02a-leapyear.scala) の18行目を見て
false が欲しい (should be false for ...) のに,
実際は true じゃん (true was not false) よん。

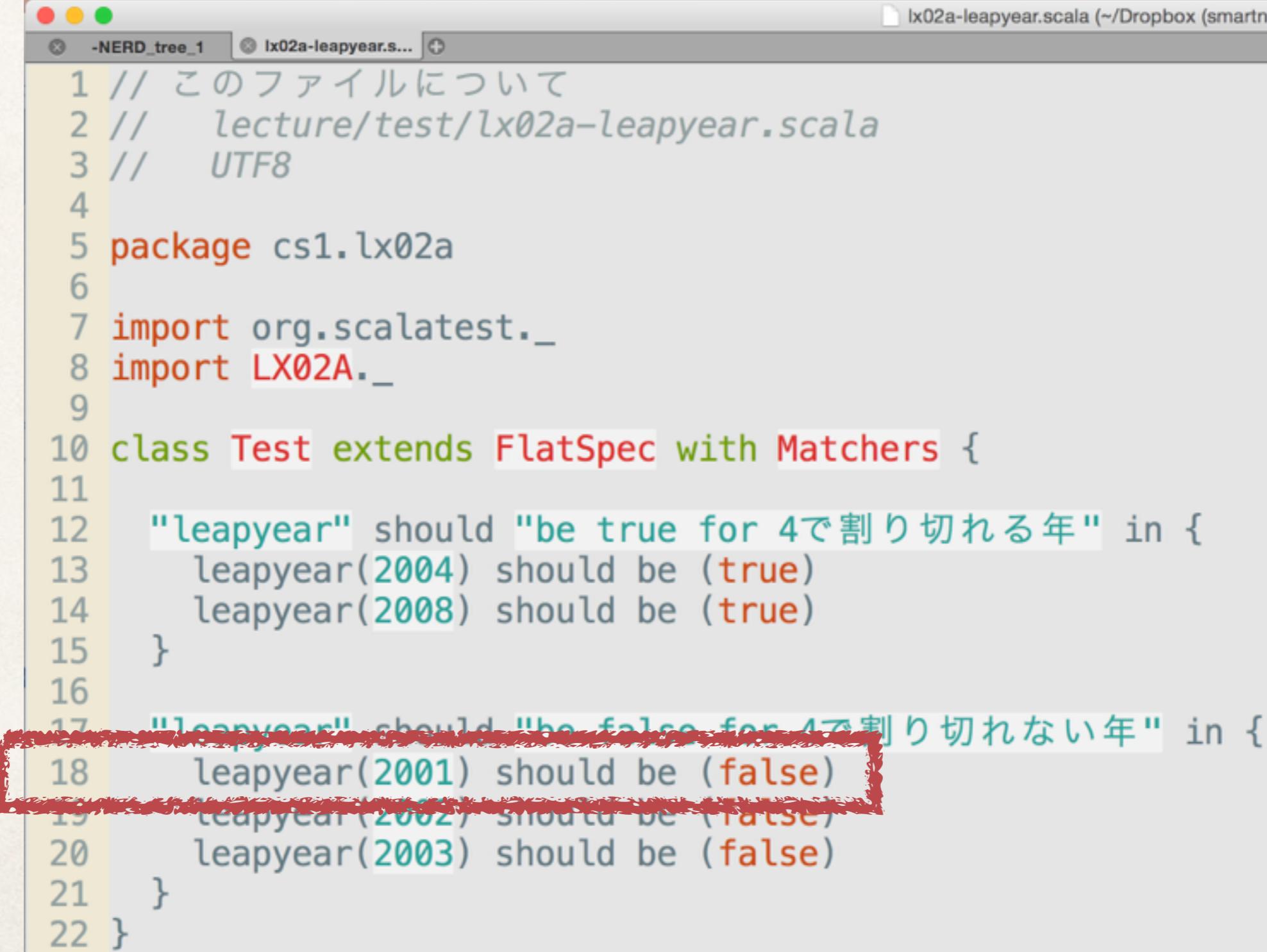
再度テストを実行

```
1. Default

> test
[info] Test:
[info] leapyear
[info] - should be true for 4で割り切れる年
[info] leapyear
[info] - should be false for 4で割り切れない
[info] true was not false (lx02a-leapyear)
[info] Run completed in 477 milliseconds.
[info] Total number of tests run: 2
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 1, failed 1, canceled 0, ignored 0, pending 0
[info] *** 1 TEST FAILED ***
[error] Failed tests:
[error] cs1.lx02a.Test
[error] (test:test) sbt.TestsFailedException: Tests unsuccessful
[error] Total time: 1 s, completed 2015/10/15 10:24:57
>
```

一箇所コケたよ
コケたテストは cs1.lx02a.Test
残念

そこでテストコードの18行目を見る
と、もちろんテストの内容は正しい



The screenshot shows a terminal window with the title bar "lx02a-leapyear.scala (~/Dropbox (smartn...)" and the tab "-NERD_tree_1". The window contains the following Scala code:

```
1 // このファイルについて
2 //   lecture/test/lx02a-leapyear.scala
3 //   UTF8
4
5 package cs1.lx02a
6
7 import org.scalatest._
8 import LX02A._
9
10 class Test extends FlatSpec with Matchers {
11
12   "leapyear" should "be true for 4で割り切れる年" in {
13     leapyear(2004) should be (true)
14     leapyear(2008) should be (true)
15   }
16
17   "leapyear" should "be false for 4で割り切れない年" in {
18     leapyear(2001) should be (false)
19     leapyear(2002) should be (false)
20     leapyear(2003) should be (false)
21   }
22 }
```

The line "leapyear" should "be false for 4で割り切れない年" is highlighted with a red box.

で，プログラムの問題を探す (までもなく，明らかに適当なのが)

- 以下を修正して， $\text{leapyear}(2001) \rightarrow \text{false}$ となるようにすればよい。



```
// ファイルについて
// lecture/src/lx02a-leapyear.scala
// UTF8

package cs1.lx02a

object Main {
    def leapyear(y: Int) = {
        false
    }
}
```

- (半端に) ずる賢い人は $\text{leapyear}(y) = \{ \text{false} \}$ に変更

19行目は ok だが、

今度はさっちは成功していた13行目が . . .

```
1. Default

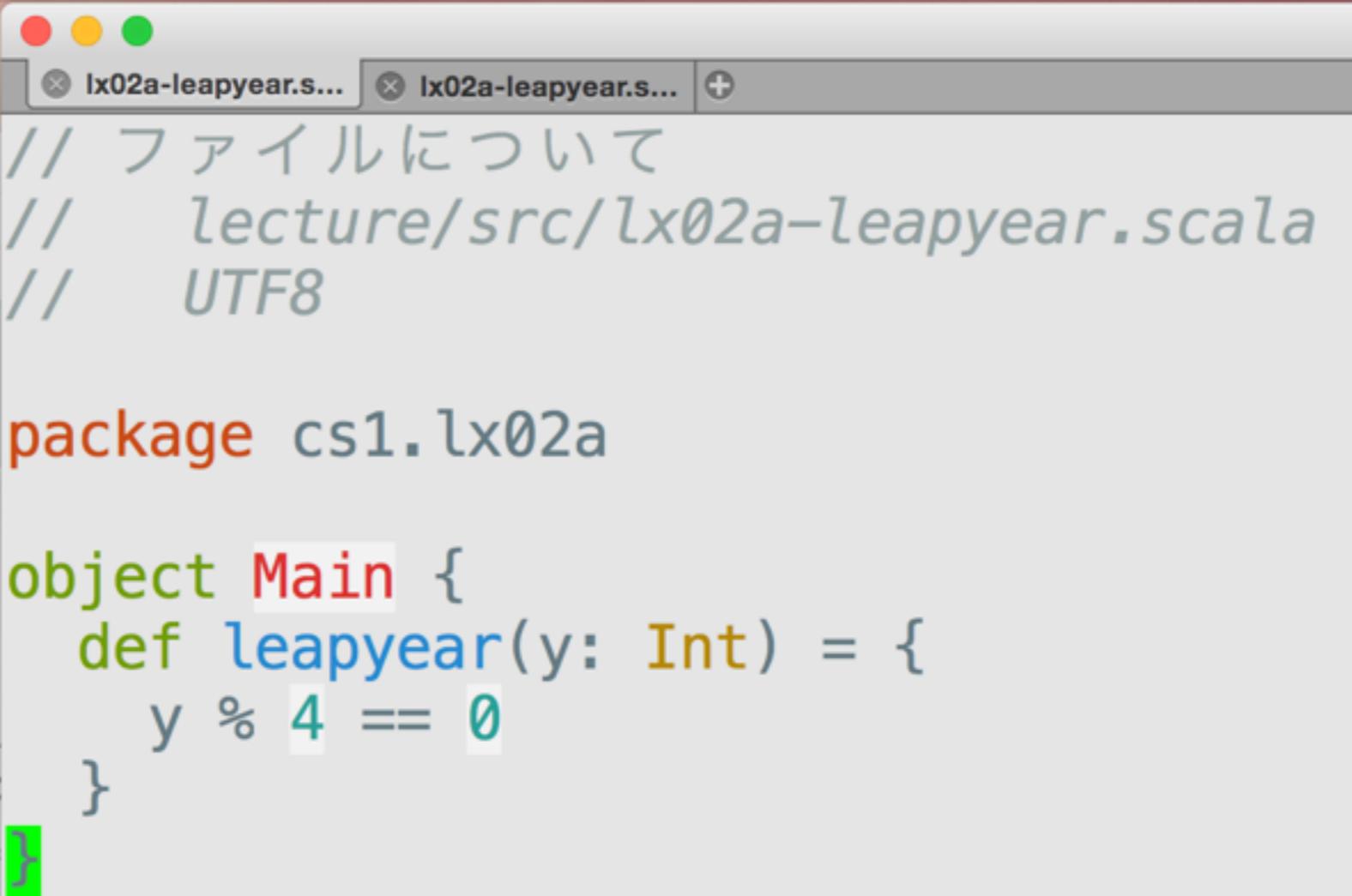
> test
[info] Compiling 1 Scala source to /Users/wakita/tmp/cs1f/scala-2.11/classes...
[info] Test:
[info] leapyear
[info] - should be true for 4で割り切れる年 *** FAILED ***
[info]   false was not true (lx02a-leapyear.scala:13)
[info] leapyear
[info] - should be false for 4で割り切れない年
[info] Run completed in 502 milliseconds.
[info] Total number of tests run: 2
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 1, failed 1, canceled 0
[info] *** 1 TEST FAILED ***
[error] Failed tests:
[error]   cs1.lx02a.Test
[error] (test:test) sbt.TestsFailedException: Tests unsuccessful
[error] Total time: 1 s, completed 2015/10/13 10:33:37
> █
```

leapyear(2004) should be (true)

もう少し真面目に対応するか

4で割り切れば閏年なんでしょ？

第一の条件：西暦年が4で割り切れる年は閏年



```
// ファイルについて
// lecture/src/lx02a-leapyear.scala
// UTF8

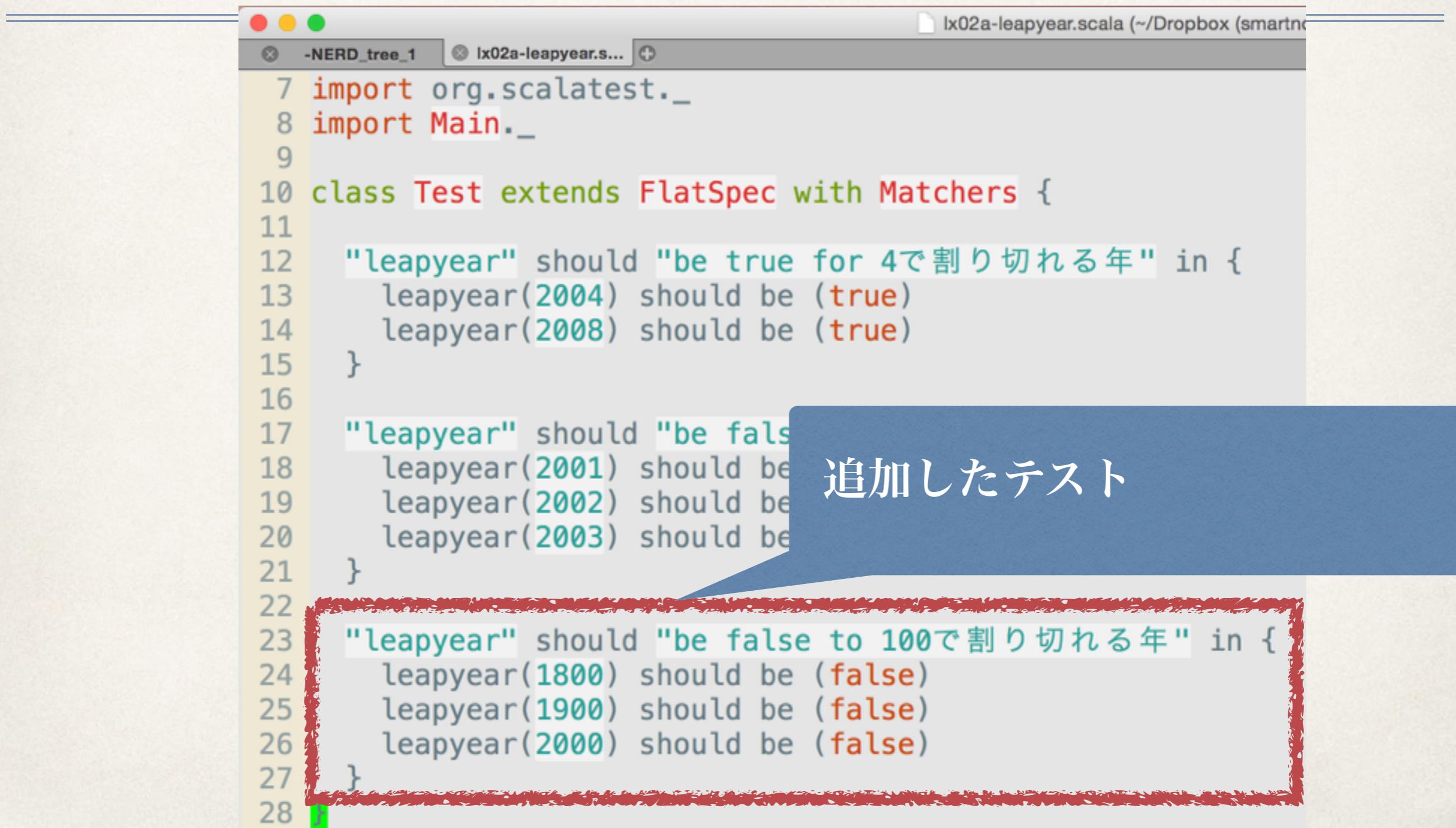
package cs1.lx02a

object Main {
    def leapyear(y: Int) = {
        y % 4 == 0
    }
}
```

やった～！すべてパス

```
1. Default  
> test  
[info] Test:  
[info] leapyear  
[info] - should be true for 4で割り切れる年  
[info] leapyear  
[info] - should be false for 4で割り切れない年  
[info] Run completed in 460 milliseconds.  
[info] Total number of tests run: 2  
[info] Suites: completed 1, aborted 0  
[info] Tests succeeded 2, failed 0, canceled 0, ignored 0, pending 0  
[info] All tests passed.  
[info] Total time: 1s, completed 2015/10/13 10:37:03  
>
```

調子にのって、 二番目のテストを追加



A screenshot of a terminal window titled "lx02a-leapyear.scala (~/Dropbox (smartnc...))". The window shows Scala test code for determining leap years. A blue callout bubble points from the text "追加したテスト" (Added test) to the line "leapyear(1800) should be (false)".

```
lx02a-leapyear.scala (~/Dropbox (smartnc...))
-NERD_tree_1 lx02a-leapyear.s...
1 import org.scalatest._
2 import Main._
3
4 class Test extends FlatSpec with Matchers {
5
6   "leapyear" should "be true for 4で割り切れる年" in {
7     leapyear(2004) should be (true)
8     leapyear(2008) should be (true)
9   }
10
11   "leapyear" should "be false to 100で割り切れる年" in {
12     leapyear(2001) should be (false)
13     leapyear(2002) should be (false)
14     leapyear(2003) should be (false)
15   }
16
17   "leapyear" should "be false to 400で割り切れる年" in {
18     leapyear(2000) should be (true)
19   }
20
21 }
```

追加したテスト

三度テストを実行

```
1. Default  
> test  
[info] Test:  
[info] leapyear  
[info] - should be true for 4で割り切れる年  
[info] leapyear  
[info] - should be false for 4で割り切れない年  
[info] leapyear  
[info] - should be false to 100で割り切れる年 *** FAILED ***  
[info] true was not false (lx02a-leapyear.scala:24)  
[info] Run completed in 480 milliseconds.  
[info] Total number of tests run: 3  
[info] Suites: completed 1, aborted 0  
[info] Tests: succeeded 2, failed 1, canceled 0, ignored 0, pending 0  
[info] *** 1 TEST FAILED ***  
[error] Failed tests:  
[error]      cs1.lx02a.Test  
[error] (test:test) sbt.TestsFailedException  
[error] Total time: 1 s, completed 2015-07-10T14:45:21+00:00  
>
```

もちろんこける（2つ成功、1つ失敗）

テストにあわせて修正

```
leapyear.scala ex01a-leapyear.... +  
object ex1a {  
    def leapyear(y: Int) = {  
        !(y % 100 == 0) &&  
        y % 4 == 0  
    }  
}
```

4度目のテスト

```
1. Default  
> test  
[info] Test:  
[info] leapyear  
[info] - should be true for 4で割り切れる年  
[info] leapyear  
[info] - should be false for 4で割り切れない年  
[info] leapyear  
[info] - should be false to 100で割り切れる年  
[info] Run completed in 472 milliseconds.  
[info] Total number of tests run: 3  
[info] Suites: completed 1, aborted 0  
[info] Tests: succeeded 3, failed 0, canceled 0, ignored 0, pending 0  
[info] All tests passed.  
[success] Total time: 1 s  
>
```

All tests passed.
緑の字が目に優しいぜ！

天の声 いやいや、まだ駄目でしょ

- ✿ 曰く「ただし、西暦年が400で割り切れる年は閏年」
- ✿ ということは、2000年とか1600年は閏年？

あとは任せた

- ✿ 課題の内容は別途詳述します。自分で閏年のプログラムを完成して下さい。
- ✿ 注意：今日の課題はほかにもあります。

プログラム開発環境

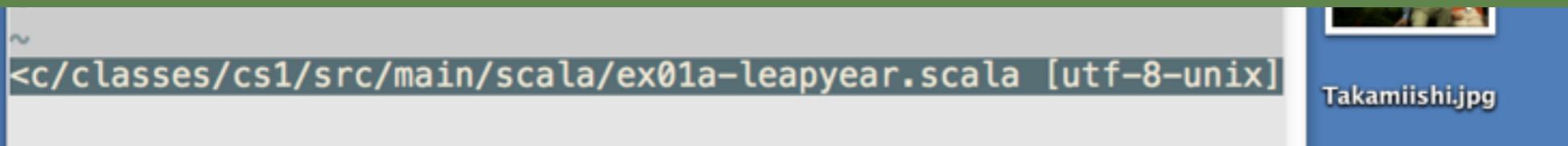
開発環境

テキストエディタ

プログラムとテストコードを編集

作業内容

- コードの修正
- ファイルの保存
- 作業中はエディタのウィンドウは開きっぱなし



プログラム

A screenshot of a terminal window titled '1. Default' showing the output of an sbt test run:

```
[info] leapyear
[info] - should be false for 4で割り切れない年
[info] leapyear
[info] - should be false to 100で割り切れる年
[info] Run completed in 835 milliseconds.
[info] Total number of tests run: 3
[info] Suites: completed 1, aborted 0
[info] Tests: succeeded 3, failed 0, canceled 0, ignored 0, pending 0
[info] All tests passed.
[success] Total time: 1 s, completed 2014/10/07 12:21:45
```

ターミナル

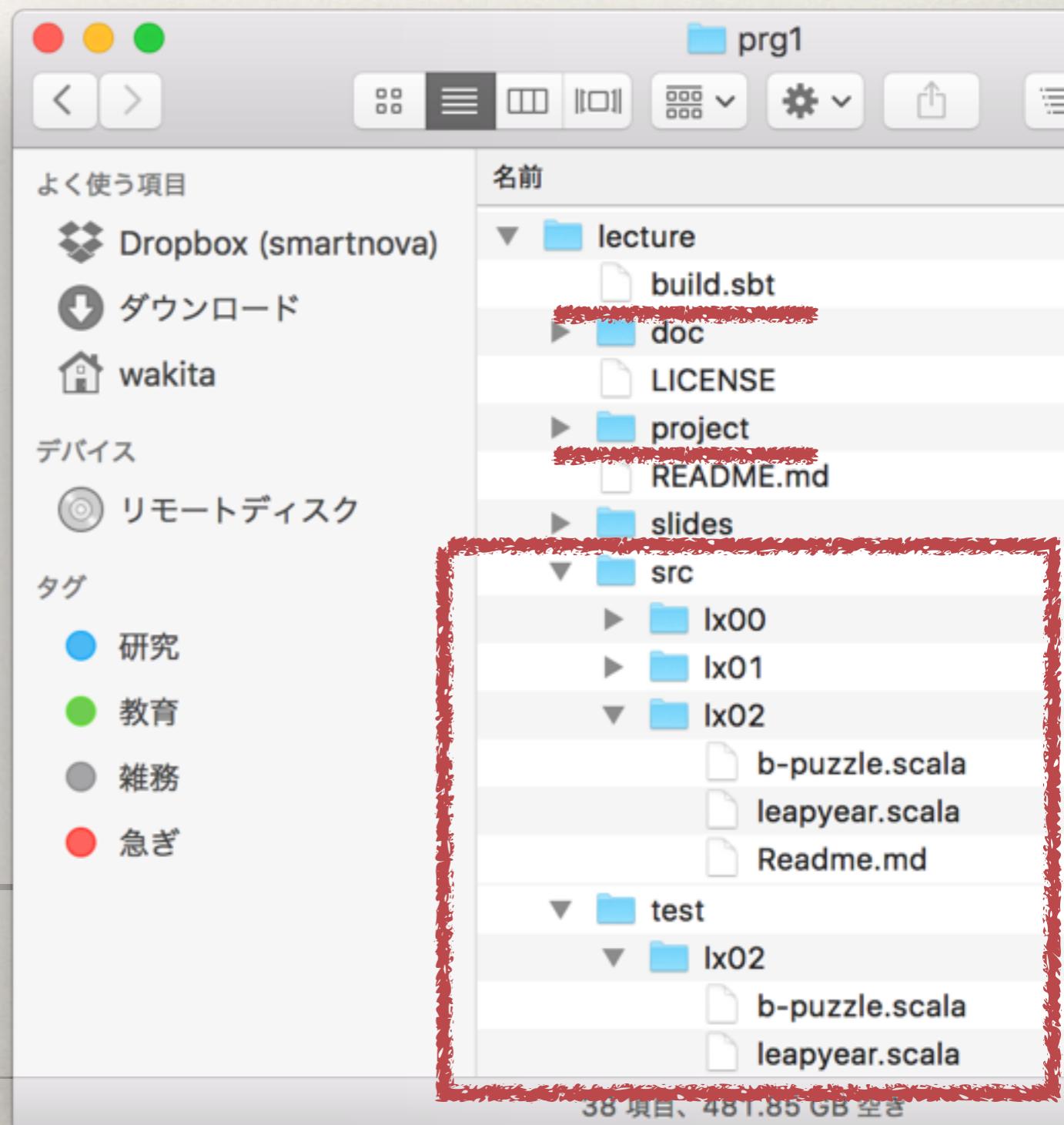
sbt を動かしっぱなし

ときどき “test” を実行

“~test”で継続テストをするのもよい

sbtプロジェクトの構成

- ❖ build.sbt
sbt の設定ファイル
- ❖ project/
sbtが勝手に作る。気にしない。
- ❖ src/
プログラムの置き場所
- ❖ test/
テストコードの置き場所



cs1/build.sbt の主な内容

記述には一級の正確さが求められます

```
scalaVersion := "2.11.8"
```

```
scalacOptions ++= Seq("-optimize", "-feature", "-unchecked", "-deprecation")
```

```
javaOptions in run ++= Seq( "-Xmx2G", "-verbose:gc")
```

```
libraryDependencies += "org.scalatest" % "scalatest_2.11" % "3.0.0" % "test"
```

```
libraryDependencies += "org.scalacheck" %% "scalacheck" % "1.13.2" % "test"
```

宿題1：テスト駆動開発を演習室の環境で実践すること

- ✿ 授業で説明を受けた閏年のプログラムをテスト駆動方式にしたがって完成させなさい。
ファイルは配布されたものをコピーし、以下のように命名したものを作成する
 - ✿ {src,test}/lx02/**my**leapyear.scala
- ✿ くれぐれもテスト駆動開発の順序を守ること
 - ✿ プログラムの仕様をテストとして記述し、テストを潰しながらプログラムを完成に導くこと。
 - ✿ プログラムを書いてから、**後付けでテストを作らない**こと。
- ✿ 提出は不要だが、次回の授業は宿題を実施したことを前提として進めます。

宿題：パズルを解くプログラムを完成させなさい

- 右のパズルを自動的に解くプログラムをテスト駆動開発方式で作成しなさい。

(科学雑誌Newton@facebook)

- ファイル

{src,test}/lx02b-puzzle.scala

この紙の上には、
1という数字が□個、
2という数字が□個、
3という数字が□個、
1から3まで以外の数字が
□個書いてある。