Announcements and Feedback

- AI Quiz due Monday, 2/6 @ 11:59
- PA0 will hopefully be posted over the weekend
  - PA1 will follow shortly after
- Stay seated until the class ends completely
Things WILL go wrong...often

Being a good computer scientist does not mean getting things 100% right all of the time. Things WILL go wrong.

A good computer scientist knows how to solve problems, and how to recover when things go wrong.
Being a good computer scientist does not mean getting things 100% right all of the time. Things WILL go wrong.

A good computer scientist knows how to solve problems, and how to recover when things go wrong.

Let's talk about some useful tools for recovering...
The REPL (read - eval - print loop)

- **From IntelliJ:** Ctrl+Shift+D
  - Highlight a line and press Ctrl+Shift+X to execute
  - Copy+paste a line and press Ctrl+Enter to execute

- **From the command line:** scala
  - Paste or type commands to run them
  - Type :help to get a list of additional commands

- **From SBT:** console
Basic Debugging
Unit Testing

- Break the big problem into smaller problems
  - Test each small solution before combining them
- Useful for debugging
  - Sanity check each step in a large process to make sure it works
  - Separate the UI from the tests
- Useful way to encode your assumptions, constraints, etc
  - Automatic reminder if your assumptions change
  - Also acts as self-documentation
Unit Testing

- Break the big problem into smaller problems
  - Test each solution initially before combining them

  If you’re building a boat, you aren’t going to build the entire thing then just throw it in the water and hope it floats...you would test throughout the whole process.

- Useful way to encode your assumptions, constraints, etc
  - Automatic reminder if your assumptions change
  - Also acts as self-documentation

The same logic applies to your coding projects!
class HelloWorldTest extends AnyFlatSpec {
  "HelloWorld.doThings()" should "return 5" in {
    assert (HelloWorld.doThings() == 5)
  }
  it should "not return 10" in {
    assert (HelloWorld.doThings() != 10)
  }
  "HelloWorld.x" should "have type Float" in {
    assert (HelloWorld.x.isInstanceOf[Float])
  }
  "Register(0).addToValue" should "return the input value"
  in {
    val reg = Register(0)
    for (i <- 1 to 10000) { assert (reg.addToValue(i) == i) }
  }
}
ScalaTest

class HelloWorldTest extends AnyFlatSpec {
  "HelloWorld.doThings()" should "return 5" in {
    assert (HelloWorld.doThings() == 5)
  }
  it should "not return 10" in {
    assert (HelloWorld.doThings() != 10)
  }
  "HelloWorld.x" should "have type Float" in {
    assert (HelloWorld.x.isInstanceOf[Float])
  }
  "Register(0).addToValue" should "return the input value" in {
    val reg = Register(0)
    for (i <- 1 to 10000) { assert (reg.addToValue(i) == i) }
  }
}
Live Demo

ScalaTest
Profiling

- IntelliJ -> Profilers

- SBT -> HProf
  - [https://docs.oracle.com/javase/8/docs/technotes/samples/hprof.html](https://docs.oracle.com/javase/8/docs/technotes/samples/hprof.html)
Profiling

- IntelliJ -> Profilers

- SBT -> HProf
  - [https://docs.oracle.com/javase/8/docs/technotes/samples/hprof.html](https://docs.oracle.com/javase/8/docs/technotes/samples/hprof.html)

```scala
fork in run := true
javaOptions in run += "-agentlib:hprof=cpu=samples,depth=10"
```
Profiling

- IntelliJ -> Profilers

- SBT -> HProf
  - https://docs.oracle.com/javase/8/docs/technotes/samples/hprof.html

```scala
fork in run := true
javaOptions in run += "-agentlib:hprof=cpu=samples,depth=10"
```
HProf Traces

JAVA PROFILE 1.0.1, created Fri Sep  3 02:24:46 2021

Copyright (c) 2003, 2005, Oracle and/or its affiliates. All rights reserved.

Redistribution and use in source and binary forms, with or without

...

TRACE 300207:
   scala.collection.StrictOptimizedLinearSeqOps.drop(LinearSeq.scala:261)
   scala.collection.StrictOptimizedLinearSeqOps.drop$(LinearSeq.scala:257)
   scala.collection.immutable.List.drop(List.scala:79)
   scala.collection.immutable.List.drop(List.scala:79)
   ...

CPU SAMPLES BEGIN (total = 185) Fri Sep  3 02:24:48 2021
rank  self  accum  count  trace  method
  1 44.86% 44.86%   83 300207 scala.collection.StrictOptimizedLinearSeqOps.drop
  2 35.14% 80.00%   65 300210 scala.collection.immutable.$colon$colon.tail
  3  5.95% 85.95%   11 300071 java.lang.ClassLoader.defineClass1
  4  2.16% 88.11%    4 300209 scala.collection.immutable.Range.foreach$mVc$sp
HProf Traces

JAVA PROFILE 1.0.1, created Fri Sep 3 02:24:46 2021

Copyright (c) 2003, 2005, Oracle and/or its affiliates. All rights reserved.

Redistribution and use in source and binary forms, with or without

...  

TRACE 300207

scala.collection.StrictOptimizedLinearSeqOps.drop(LinearSeq.scala:261)
scala.collection.StrictOptimizedLinearSeqOps.drop$(LinearSeq.scala:257)
scala.collection.immutable.List.drop(List.scala:79)
scala.collection.immutable.List.drop(List.scala:79)

...

CPU SAMPLES BEGIN (total = 185) Fri Sep 3 02:24:48 2021

rank  self  accum  count  trace  method
1 44.86% 44.86%  83 300207 scala.collection.StrictOptimizedLinearSeqOps.drop
2 35.14% 80.00%  65 300210 scala.collection.immutable.$colon$colon.tail
3  5.95% 85.95%  11 300071 java.lang.ClassLoader.defineClass1
4  2.16% 88.11%  4 300209 scala.collection.immutable.Range.foreach$mVc$sp