PEP Scala (2)

Email:	christian.urban at kcl.ac.uk
Office:	N7.07 (North Wing, Bush House)

Slides & Code:	KEATS

Office Hours:	Thursdays 12:00 – 14:00
Additionally:	(for Scala) Tuesdays 10:45 – 11:45

My Scala Version

```
$ scala

Welcome to Scala 2.13.1 (Java HotSpot(TM)
64-Bit Server VM, Java 9). Type in expressions
for evaluation. Or try :help.

scala>
```

With older versions you will get strange results with my reference implementation.

Reference Implementation

Keep your implementation and my reference implementation separate.

```
$ scala -cp collatz.jar

scala> CW6a.collatz(6)
res0: Long = 8


scala> import CW6a._
scala> collatz(9)
res1: Long = 19
```

Preliminary Part 7

$$\texttt{overlap}(d_1, d_2) = \frac{d_1 \cdot d_2}{max(d_1^2, d_2^2)}$$

where d_1^2 means $d_1 \cdot d_1$ and so on

Assignments

Don't change anything with the templates!

Avoid at all costs:

- `var`
- `return`
- `ListBuffer`
- `mutable`
- `.par`

Assignments

Don't change anything with the templates!

Avoid at all costs:

- `var`
- `return`
- `ListBuffer`
- `mutable`
- `.par`

"Scala — Slowly compiled academic language"
— a joke(?) found on Twitter

Email: Hate 'val'

Subject: **Hate '`val`'** 01:00 AM

Hello Mr Urban,

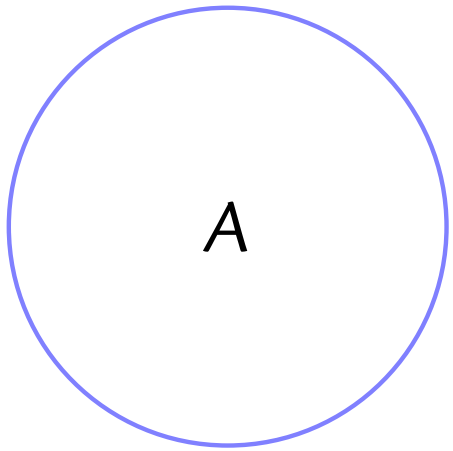
I just wanted to ask, how are we suppose to work with the completely useless **`val`**, that can't be changed ever? Why is this rule active at all? I've spent 4 hours not thinking on the coursework, but how to bypass this annoying rule. What's the whole point of all these coursework, when we can't use everything Scala gives us?!?

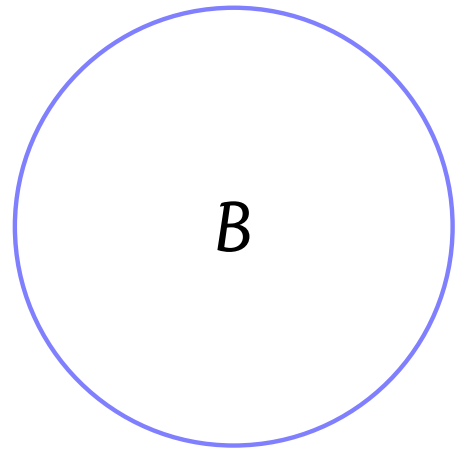
Regards.

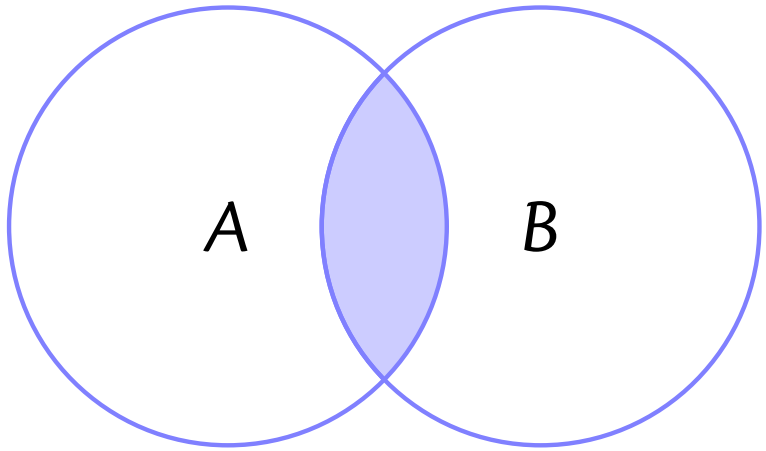
« deleted »

Par: Intersections

$A = \{1, 2, 3, \ldots, 1000\}$ $B = \{1, 5, 9, 13, \ldots, 997\}$





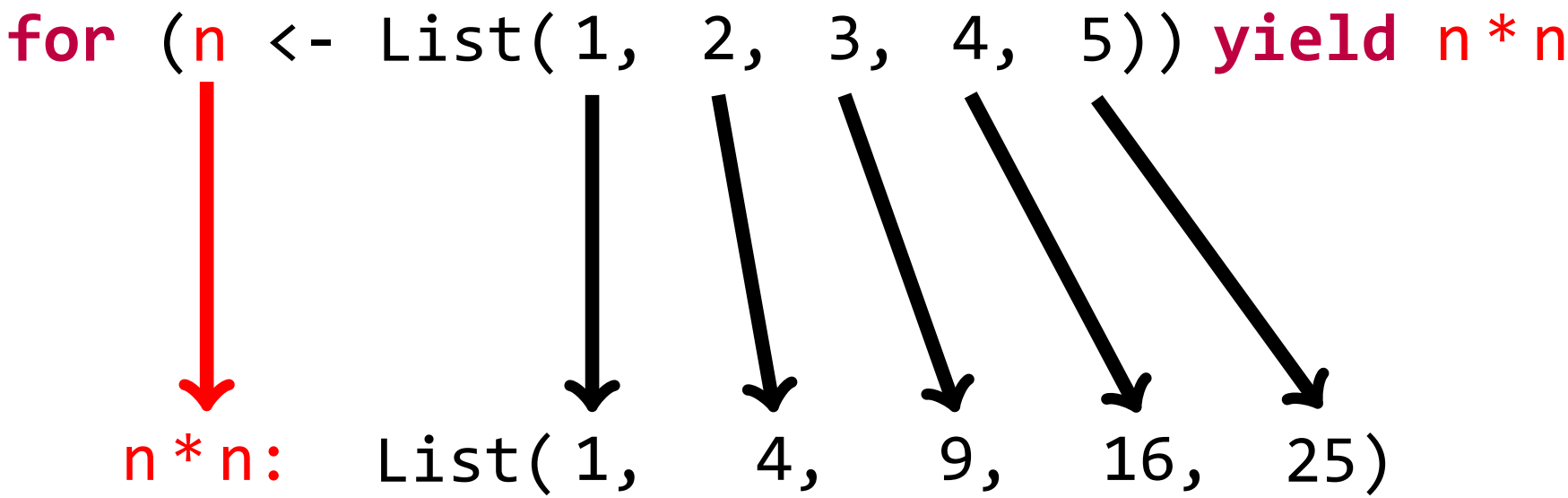


How many elements are in $A \cap B$?

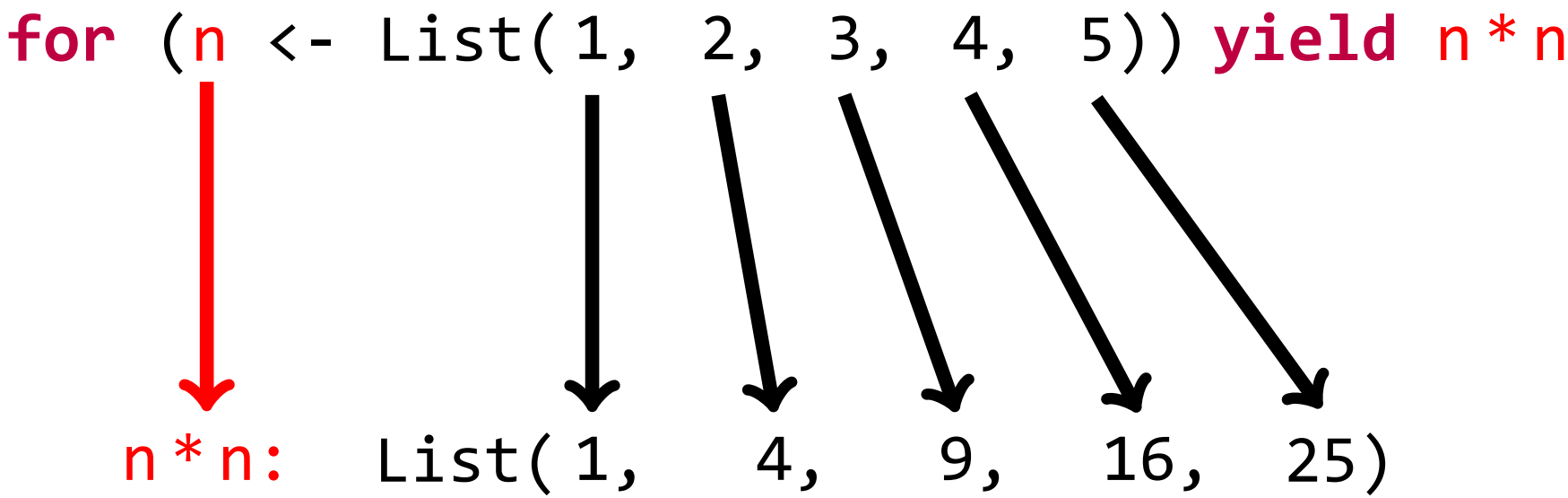
For-Comprehensions Again

```
for (n <- List(1, 2, 3, 4, 5)) yield n*n
```

For-Comprehensions Again



For-Comprehensions Again



This is for when the for-comprehension **yields / produces** a result.

For-Comprehensions Again

```
for (n <- List(1, 2, 3, 4, 5)) yield n*n
```

vs

```
for (n <- List(1, 2, 3, 4, 5)) println(n)
```

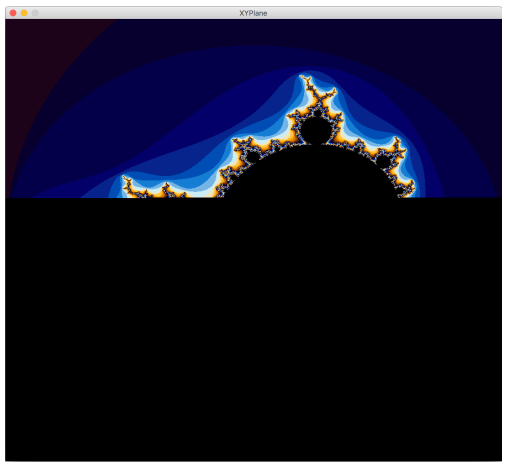
The second version is in case the for **does not** produce any result.

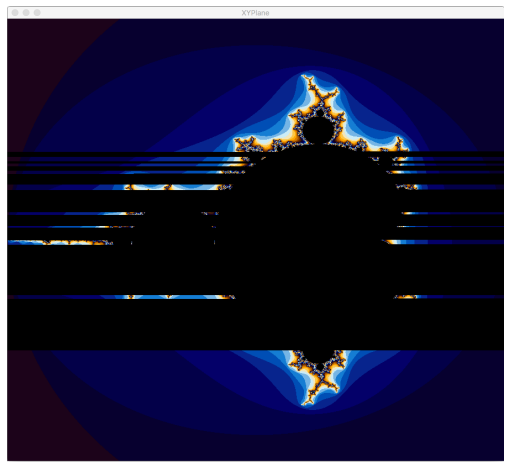
Why Scala? No null!

- **You can avoid `null`:**

"I call it my billion-dollar mistake. It was the invention of the null reference in 1965. At that time, I was designing the first comprehensive type system for references in an object oriented language (ALGOL W). My goal was to ensure that all use of references should be absolutely safe, with checking performed automatically by the compiler. But I couldn't resist the temptation to put in a null reference, simply because it was so easy to implement. This has led to innumerable errors, vulnerabilities, and system crashes, which have probably caused a billion dollars of pain and damage in the last forty years." Sir Tony (Hoare)

Questions?





My Office Hours: Thursdays 12 – 14

And specifically for Scala: Tuesdays 10:45 – 11:45