Where can you leave the parens off a tuple?

When passing a tuple to a function that takes a single argument.

Give the two forms of the type of function that takes two Ints and returns and String.

```
(Int,Int) => String
```

or

```
Function2[Int,Int, String]
```

What are the advantages of case classes over extractors for pattern matching?

- They're easier to set up and define.
- They're faster because the compiler can't optimize extractors, since their unapply and unapplySeq can do anything.
- If inheriting a sealed class, the compiler can check for pattern match exhaustiveness.

What does representation independence allow?

Patterns to match objects of different types.

How do case classes break representation independence?

If a match of a selector s succeeds against case C(...), you know the selector expression is an instance of C.

To be an extractor you need __, but __ is optional.

Extraction method (unapply)

Injenction method (apply)

What does an extractor with two or more variables return?

An n-element tuple wrapped in a Some.

Why are injenctions and extractions often paired together?

You can then use the object's name for both a constructor and a pattern, which simulates the convention for pattern matching with case classes.

What does an extractor with one variable return?

An element wrapped in a Some, because there is no 1-tuple.

What does an extractor with no variables return?

A Boolean indicating match success or failure.

Describe an extractor with a variable number of variables.

It is named `unapplySeq` and returns `Option[Seq[T]]` for any type T.

What are the two main purposes of extractors?

- To allow constructor pattern matching without a case class. For example, for pre-existing types like String:

```
s match { case EMail(...
```

- To allow the definition of new patterns that don't follow the internal representation of the scrutinee.

Describe regex extraction.

If X is a regex with three capturing groups,

```
valX(a, b, c) = someString
```

a, b, c, will be bound to String matches or null for no match.

What is the consequence of breaking representational independence?

Changing case class (e.g. changing its name, moving it to another package) will break client code that matches on it.