What's an exception?

An exception is an error that happens in code.

Some types of errors can be predicted and named.

Catching an exception

An exception is caught first by creating a code block around the code that gets the error.

This is done with the try statement code block.

The try statement

The try statement actually has two code blocks.

The first is declared directly after the try keyword, and this code block ends, and is followed by the declaration of the catch keyword.

The catch keyword includes the declaration of variables, in parentheses, and then has its own code block.

```
try {
    // statements that might get errors
} catch (Exception e) {
    // code to 'handle' the exception
}
```

The Scanner class

The Scanner class is described as a simple text scanner, which can parse primitive types and strings.

To use the Scanner class, we have to create an instance of Scanner.

This means we're creating an object of type Scanner.

We'll use the keyword, new, to do it.

new keyword

The new keyword is used in what Java calls a Class Instance Creation Expression.

In its simplest form, it's the word new, followed by the class name, and empty parentheses.

```
ClassName variableName = new ClassName();
```

We can optionally pass arguments in those parentheses, as we saw with methods.

```
ClassName variableName = new ClassName(argument1, argument2);
```

We saw that we could do this with the String class, passing the text in the parentheses.

Instantiating Scanner

For reading input from the console or terminal, we instantiate a scanner object using new, followed by the Scanner class name, and passing System.in, as an argument, in the parentheses.

```
Scanner sc = new Scanner(System.in);
```

For reading input from a file, we instantiate a scanner object using new, again with the Scanner class name, but pass a File object, as an argument, in the parentheses.

File is another class provided by Java, for reading and writing files.

```
Scanner sc = new Scanner(new File("nameOfFileOnFileSystem"));
```

Using the import statement

I haven't talked about the import statement yet, but this statement lets us use classes from other people's code.

In this case, Java provides a library of code, which includes the Scanner class in a library called java.util.

```
import java.util.Scanner;
```

IntelliJ Auto-import setting

Earlier in the course, when I configured IntelliJ with a number of different settings, I recommended that you enable these two options in the Auto Import menu.

Add unambiguous imports on the fly and Optimise imports on the fly.

If you enable these options, then IntelliJ will automatically add and/or remove import statements as required. You almost always want to enable these options.

Let's go back to the code so I can show you what I mean.