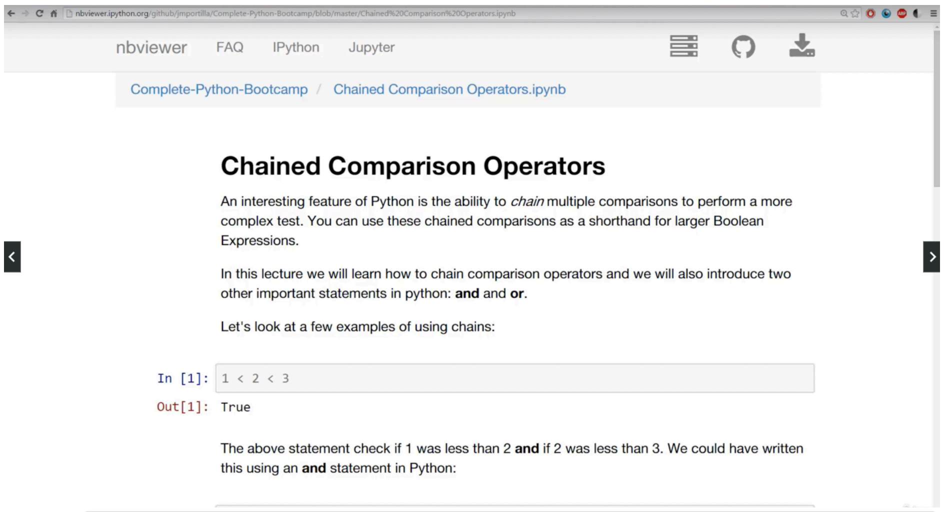
SECTION 22: ERRORS AND EXCEPTIONS HANDLING - 46 minutes, 6 parts

4/4 Chained Comparison Operators

- -> chained comparison operators
- -> chaining multiple comparisons to perform a more complex test
- -> these can be used as a shorthand for longer boolean operations
- -> we also have the `and` and `or` statements
- **-> in the project ipynb file**
  - -> he gives an example of a boolean operation
  - -> he then combines multiples of these operations
  - **-> we also have and statements in Python**
    - -> he does an example with this, which combines two of them together
  - **-> he then combines multiples of them together**
    - -> a < and > statement in the same line
    - -> Python checks both instances of the comparisons in this case
  - **-> we can also use an `or` boolean operator**
    - -> Python checks both instances of the comparisons
  - **-> either of these statements can be True**
    - -> checking if the first condition or if the second condition is True
    - -> he does another example
    - -> chained comparison operators


nbviewer
FAQ
IPython
Jupyter
Complete-Python-Bootcamp / Chained Comparison Operators.ipynb

Chained Comparison Operators

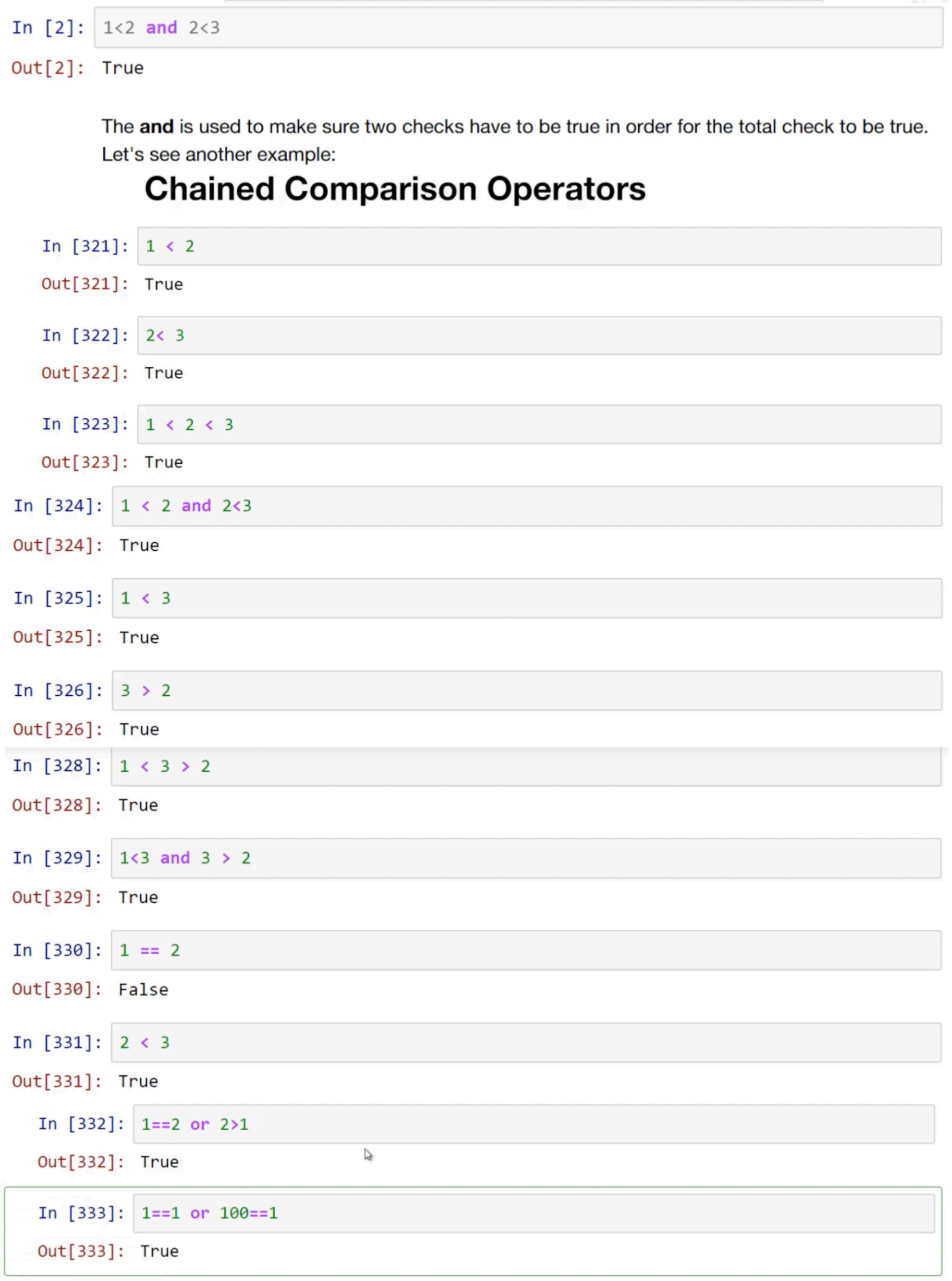
An interesting feature of Python is the ability to *chain* multiple comparisons to perform a more complex test. You can use these chained comparisons as a shorthand for larger Boolean Expressions.

In this lecture we will learn how to chain comparison operators and we will also introduce two other important statements in python: **and** and **or**.

Let's look at a few examples of using chains:

```
In [1]: 1 < 2 < 3
Out[1]: True
```

The above statement check if 1 was less than 2 **and** if 2 was less than 3. We could have written this using an **and** statement in Python:



```
In [2]: 1<2 and 2<3
Out[2]: True
```

The **and** is used to make sure two checks have to be true in order for the total check to be true. Let's see another example:

Chained Comparison Operators

```
In [321]: 1 < 2
Out[321]: True

In [322]: 2< 3
Out[322]: True

In [323]: 1 < 2 < 3
Out[323]: True

In [324]: 1 < 2 and 2<3
Out[324]: True

In [325]: 1 < 3
Out[325]: True

In [326]: 3 > 2
Out[326]: True

In [328]: 1 < 3 > 2
Out[328]: True

In [329]: 1<3 and 3 > 2
Out[329]: True

In [330]: 1 == 2
Out[330]: False

In [331]: 2 < 3
Out[331]: True

In [332]: 1==2 or 2>1
Out[332]: True

In [333]: 1==1 or 100==1
Out[333]: True
```

- **-> `and` and `or` statements in the code**
  - -> these can be treated as chained comparison operators
  - -> we also have a quiz for this

Chained Comparison Operators ¶

An interesting feature of Python is the ability to *chain* multiple comparisons to perform a more complex test. You can use these chained comparisons as a shorthand for larger Boolean Expressions.

In this lecture we will learn how to chain comparison operators and we will also introduce two other important statements in python: **and** and **or**.

Let's look at a few examples of using chains:

```
In [1]: 1 < 2 < 3
```

```
Out[1]: True
```

The above statement check if 1 was less than 2 **and** if 2 was less than 3. We could have written this using an **and** statement in Python: