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

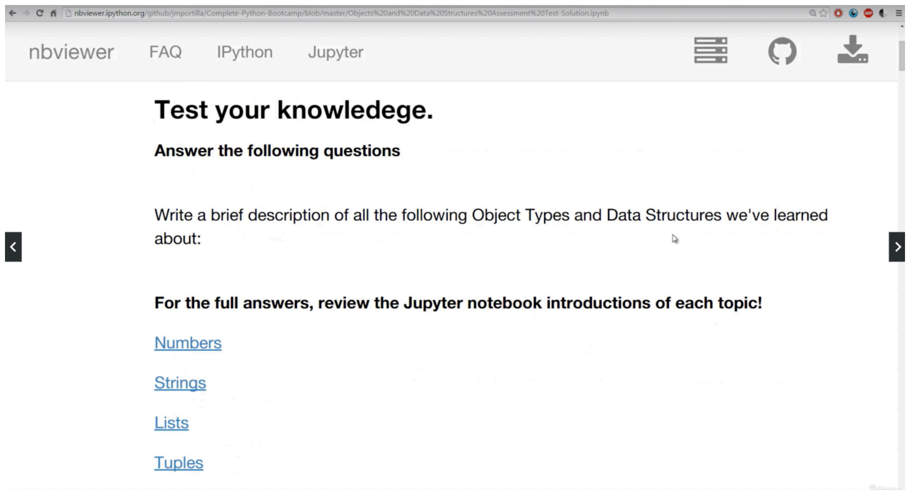
2/4 Objects and Data Structures Assessment - Solutions

- -> the objects and data structures assessment
- -> a description of all the objects and data types which we've learned about
- -> he's gone to the solutions ipynb file
- -> this is for tuples and dictionaries

- **-> numbers**
  - -> write an equation that uses multiplication, division, subtraction and exponentials
  - -> it's all of these things in one cell

- **-> then explaining what the cells will produce**
  - -> 2/3 makes 0
  - -> Python 2 does class division for integers
  - -> we want float division in this example
  - -> this is done by changing 2 to 2.0

- **-> then three questions about Python code**
  - -> typing the code to check that order of operations is understood
  - -> then the type of the result of this operation



nbviewer.ipython.org/github/jmportilla/Complete-Python-Bootcamp/blob/master/Objects%20and%20Data%20Structures%20Assessment%20Test-Solution.ipynb

Test your knowledege.

Answer the following questions

Write a brief description of all the following Object Types and Data Structures we've learned about:

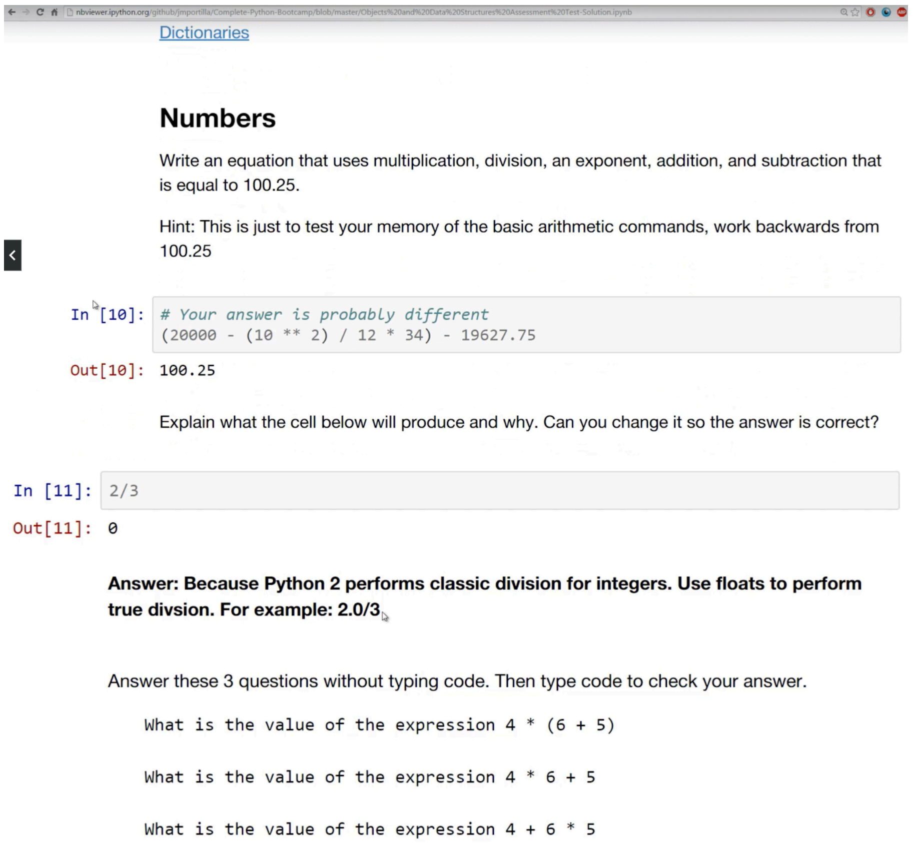
For the full answers, review the Jupyter notebook introductions of each topic!

Numbers

Strings

Lists

Tuples



Dictionaries

Numbers

Write an equation that uses multiplication, division, an exponent, addition, and subtraction that is equal to 100.25.

Hint: This is just to test your memory of the basic arithmetic commands, work backwards from 100.25

```
In [10]: # Your answer is probably different
         (20000 - (10 ** 2) / 12 * 34) - 19627.75
Out[10]: 100.25
```

Explain what the cell below will produce and why. Can you change it so the answer is correct?

```
In [11]: 2/3
Out[11]: 0
```

Answer: Because Python 2 performs classic division for integers. Use floats to perform true divsion. For example: 2.0/3

Answer these 3 questions without typing code. Then type code to check your answer.

```
What is the value of the expression 4 * (6 + 5)

What is the value of the expression 4 * 6 + 5

What is the value of the expression 4 + 6 * 5
```

- **-> the string section for this**
  - -> then what you would use to find a number's square root, as well as it's square
  - -> there is a map module for this

  - **-> indexing starts at 0**
    - -> for indexing and slicing
    - -> he prints out the same number backwards, then the 0th element of it

  - **-> building out a list**
    - -> this method between brackets
    - -> or having a list of 0 and multiplying it by 3

  - **-> then sorting a list**
    - -> this could have been done with the sorted function, or with the method sort()

  - **-> then re-assigning hello to a list**
    - -> indexing for nested lists
    - -> this could also be done using the sorted function, or sort() method

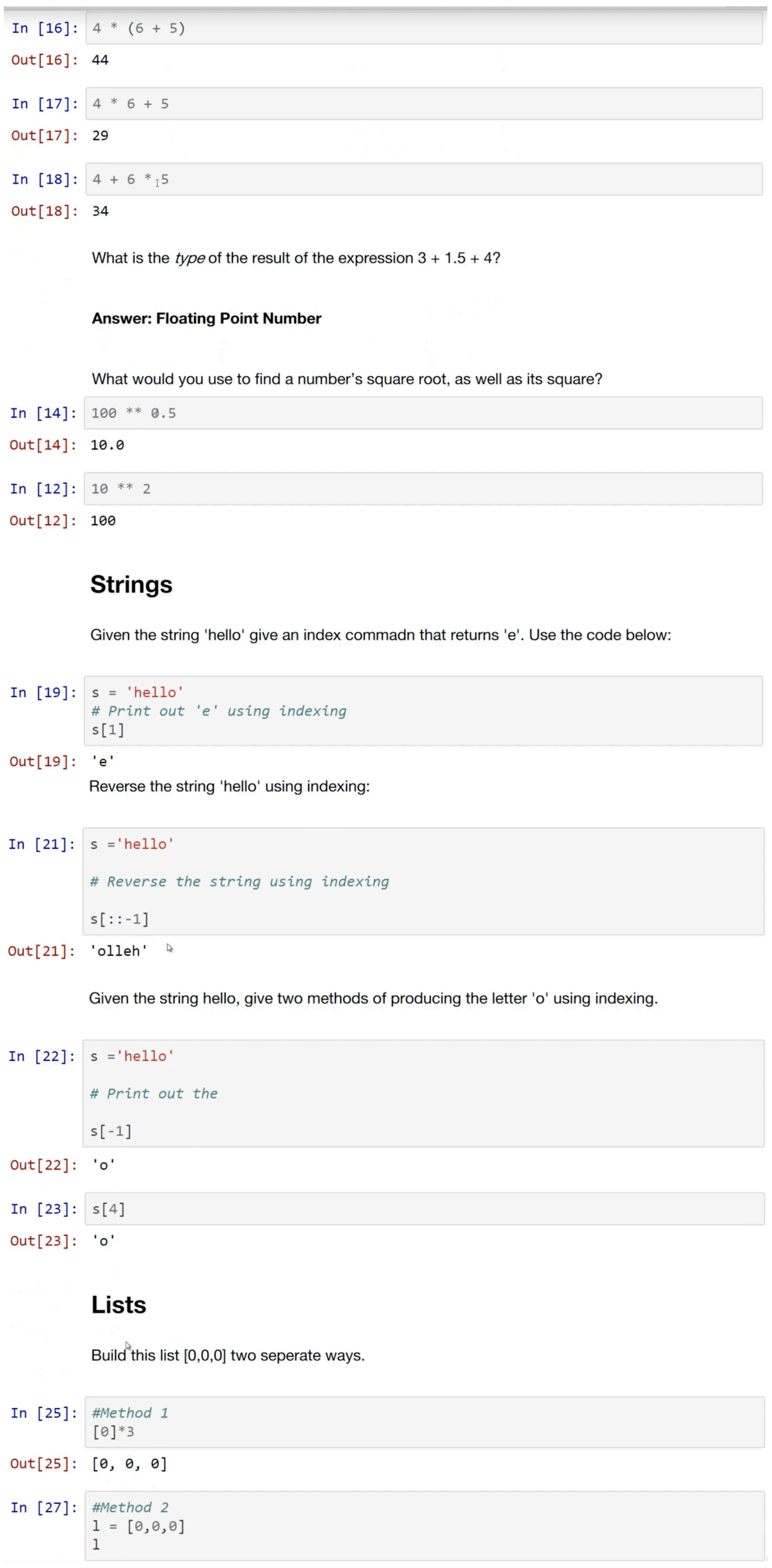
    - **-> then multiple keys and indices**
        - -> the other has two keys
        - -> the third is the key, index, nested key and then three more keys left

- **-> dictionaries can't be sorted, because normal dictionaries are mappings not sequences**
    - -> the order of the mappings doesn't matter
    - -> there is a subclass of dictionaries called ordered dictionaries

- **-> tuples are immutable and lists aren't**
    - -> tuples look like coordinates
    - -> sets don't allow for duplicate items

- **-> booleans**
    - -> there is a section on comparison operators
    - -> then there are also comparison operators
    - -> he gives examples of these in the notebook
    - -> comparisons between different elements of a list

- -> the next lecture is Python comparison operators



```
In [16]: 4 * (6 + 5)
Out[16]: 44

In [17]: 4 * 6 + 5
Out[17]: 29

In [18]: 4 + 6 * 5
Out[18]: 34
```

What is the *type* of the result of the expression 3 + 1.5 + 4?

Answer: Floating Point Number

What would you use to find a number's square root, as well as its square?

```
In [14]: 100 ** 0.5
Out[14]: 10.0

In [12]: 10 ** 2
Out[12]: 100
```

Strings

Given the string 'hello' give an index commadn that returns 'e'. Use the code below:

```
In [19]: s = 'hello'
         # Print out 'e' using indexing
         s[1]
Out[19]: 'e'
```

Reverse the string 'hello' using indexing:

```
In [21]: s ='hello'

         # Reverse the string using indexing

         s[::-1]
Out[21]: 'olleh'
```

Given the string hello, give two methods of producing the letter 'o' using indexing.

```
In [22]: s ='hello'

         # Print out the

         s[-1]
Out[22]: 'o'

In [23]: s[4]
Out[23]: 'o'
```

Lists

Build this list [0,0,0] two seperate ways.

```
In [25]: #Method 1
         [0]*3
Out[25]: [0, 0, 0]

In [27]: #Method 2
         l = [0,0,0]
         l
```

```
Out[27]: [0, 0, 0]
```

Reassign 'hello' in this nested list to say 'goodbye' item in this list:

```
In [28]: l = [1,2,[3,4,'hello']]
```

```
In [31]: l[2][2] = 'goodbye'
```

```
In [32]: l
```

```
Out[32]: [1, 2, [3, 4, 'goodbye']]
```

Sort the list below:

```
In [33]: l = [3,4,5,5,6]
```

```
In [38]: #Method 1
         sorted(l)
```

```
Out[38]: [3, 4, 5, 5, 6]
```

```
In [40]: #Method 2
         l.sort()
         l
```

```
Out[40]: [3, 4, 5, 5, 6]
```

Dictionaries

Using keys and indexing, grab the 'hello' from the following dictionaries:

```
In [41]: d = {'simple_key':'hello'}
         # Grab 'hello'
```

```
In [42]: d['simple_key']
```

```
Out[42]: 'hello'
```

```
In [43]: d = {'k1':{'k2':'hello'}}
         # Grab 'hello'
```

```
In [44]: d['k1']['k2']
```

```
Out[44]: 'hello'
```

```
In [45]: # Getting a little tricker
         d = {'k1':[{'nest_key':['this is deep',['hello']]}]}
```

```
In [51]: # This was harder than I expected...
         d['k1'][0]['nest_key'][1][0]
```

```
Out[51]: 'hello'
```

```
In [52]: # This will be hard and annoying!
         d = {'k1':[1,2,{'k2':['this is tricky',{'toughie':[1,2,['hello']]}]}]}
```

```
In [61]: # Phew
         d['k1'][2]['k2'][1]['toughie'][2][0]
```

```
Out[61]: 'hello'
```

Can you sort a dictionary? Why or why not?

Answer: No! Because normal dictionaries are *mappings* not a sequence.

Tuples

What is the major difference betwen tuples and lists?

Tuples are immutable!

How do you create a tuple?

```
In [63]: t = (1,2,3)
```

Sets

What is unique about a set?

Answer: They don't allow for duplicate items!

Use a set to find the unique values of the list below:

```
In [64]: l = [1,2,2,33,4,4,11,22,3,3,2]
```

```
In [65]: set(l)

Out[65]: {1, 2, 3, 4, 11, 22, 33}
```

Booleans

For the following quiz questions, we will get a preview of comparison operators:

Operator	Description	Example
==	If the values of two operands are equal, then the condition becomes true.	(a == b) is not true.
!=	If values of two operands are not equal, then condition becomes true.	
<>	If values of two operands are not equal, then condition becomes true.	(a <> b) is true. This is similar to != operator.
>	If the value of left operand is greater than the value of right operand, then condition becomes true.	(a > b) is not true.
<	If the value of left operand is less than the value of right operand, then condition becomes true.	(a < b) is true.
>=	If the value of left operand is greater than or equal to the value of right operand, then condition becomes true.	(a >= b) is not true.
<=	If the value of left operand is less than or equal to the value of right operand, then condition becomes true.	(a <= b) is true.

What will be the resulting Boolean of the following pieces of code (answer fist then check by typing it in!)

```
In [66]: # Answer before running cell
         2 > 3

Out[66]: False
```

```
In [67]: # Answer before running cell
         3 <= 2

Out[67]: False
```

```
In [68]: # Answer before running cell
         3 == 2.0

Out[68]: False
```

```
In [69]: # Answer before running cell
         3.0 == 3

Out[69]: True
```

```
In [70]: # Answer before running cell
         4**0.5 != 2

Out[70]: False
```

Final Question: What is the boolean output of the cell block below?

```
In [71]: # two nested lists
         l_one = [1,2,[3,4]]
         l_two = [1,2,{'k1':4}]

         #True or False?
         l_one[2][0] >= l_two[2]['k1']

Out[71]: False
```

Great Job on your first assessment!

Back to top