SECTION 20: ADVANCED PYTHON PROJECTS AND DATA STRUCTURES - 41 minutes, 7 parts

2/7 Advanced Strings

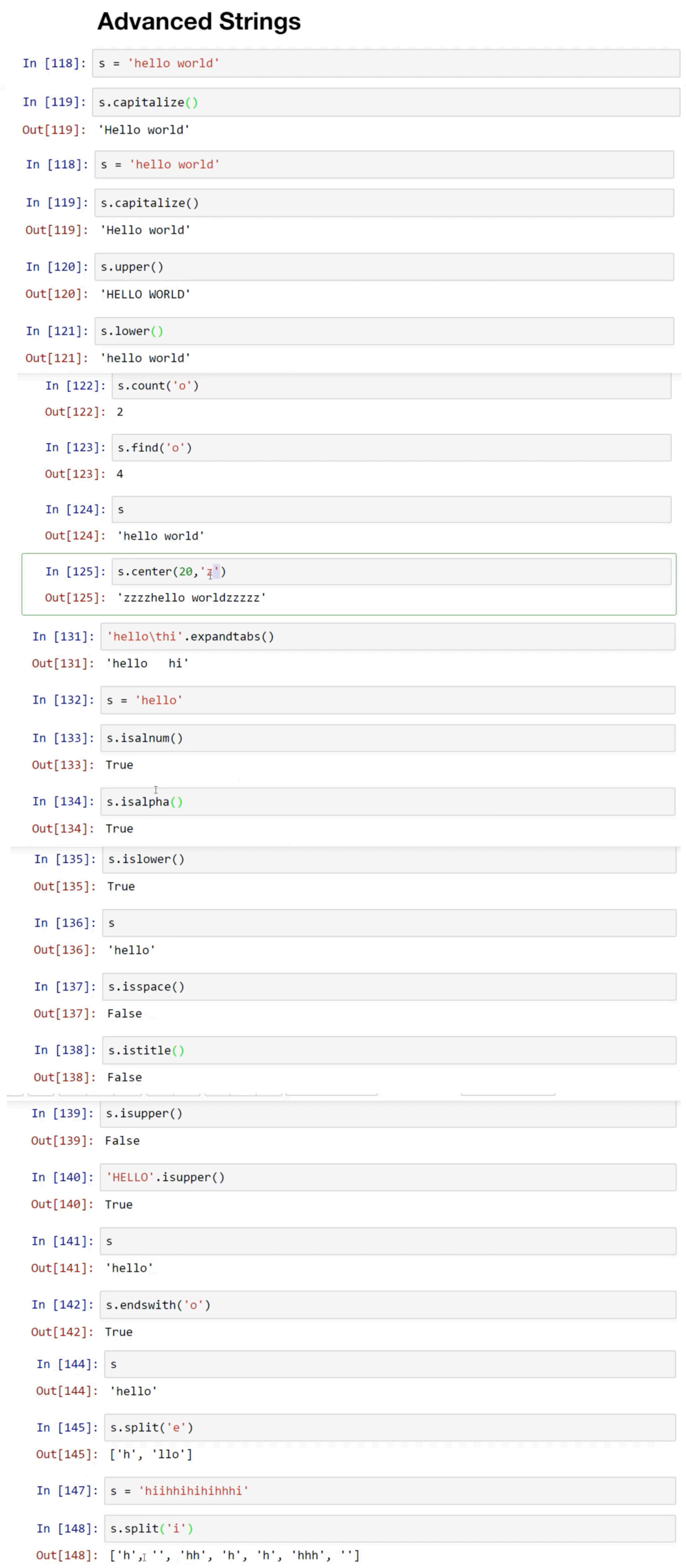
- **-> string objects**
  - -> there are methods for these which can be used to add functionality
  - -> he has created a string and stored it in a variable
  - -> the .capitalize() method <- uppercase first word in the string

- **-> to uppercase or lowercase every word in the string**
  - -> .upper()
  - -> .lower()

- **-> location and counting methods**
  - -> the .count() method can be used with the argument of the letter which you are searching for
  - -> this returns the number of times the letter appears in the string
  - -> .find() <- this returns the index of the argument where this happens

  - **-> formatting methods**
    - **-> the center() method**
      - -> place the string centred between a provided string of a certain length
      - -> s.center(20,'z')
        - -> this means take the string stored by the `s` variable and place it in the centre of 20s's (10 to the left of it and 10 to the right)

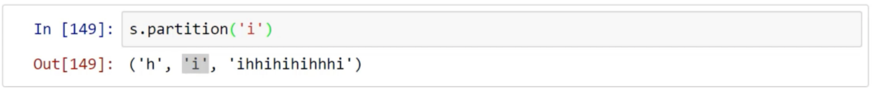
    - **-> the expand tabs method**
      - -> print(`hello\thi`)
      - -> this adds a tab after 'hello' when the string is printed
      - -> \t <- this is for the tab

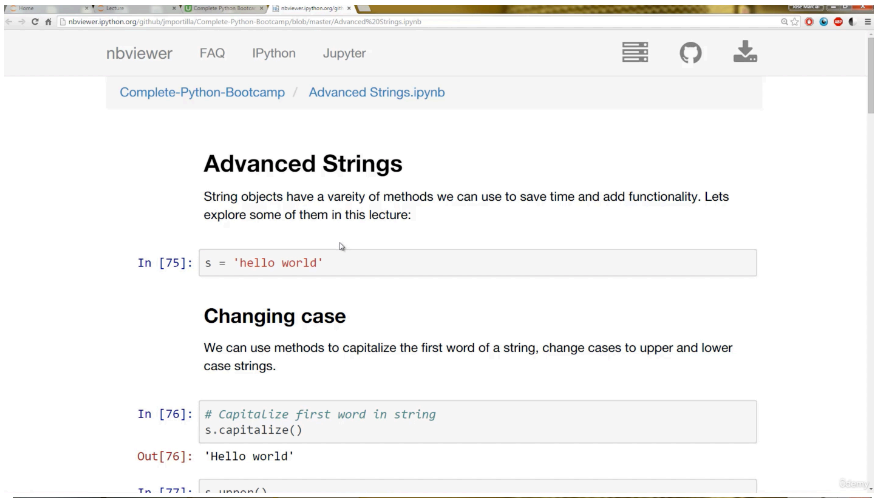


Advanced Strings

```
In [118]: s = 'hello world'

In [119]: s.capitalize()
Out[119]: 'Hello world'

In [118]: s = 'hello world'

In [119]: s.capitalize()
Out[119]: 'Hello world'

In [120]: s.upper()
Out[120]: 'HELLO WORLD'

In [121]: s.lower()
Out[121]: 'hello world'

In [122]: s.count('o')
Out[122]: 2

In [123]: s.find('o')
Out[123]: 4

In [124]: s
Out[124]: 'hello world'

In [125]: s.center(20,'z')
Out[125]: 'zzzzhello worldzzzzz'

In [131]: 'hello\thi'.expandtabs()
Out[131]: 'hello   hi'

In [132]: s = 'hello'

In [133]: s.isalnum()
Out[133]: True

In [134]: s.isalpha()
Out[134]: True

In [135]: s.islower()
Out[135]: True

In [136]: s
Out[136]: 'hello'

In [137]: s.isspace()
Out[137]: False

In [138]: s.istitle()
Out[138]: False

In [139]: s.isupper()
Out[139]: False

In [140]: 'HELLO'.isupper()
Out[140]: True

In [141]: s
Out[141]: 'hello'

In [142]: s.endswith('o')
Out[142]: True

In [144]: s
Out[144]: 'hello'

In [145]: s.split('e')
Out[145]: ['h', 'llo']

In [147]: s = 'hiihhihihihhhi'

In [148]: s.split('i')
Out[148]: ['h', '', 'hh', 'h', 'h', 'hhh', '']
```

            - -> parsing after text data
            - -> .expandtabs() <- this is not used as often
    - **-> .isalnum <- this checks if the characters in s are alphaneumeric**
        - -> we can also check if they are alphabetic
        - -> this is for natural language processing
        - -> variable_name.islower() <- to return a boolean that checks if the string is lowercase or not
        - -> s.isspace() <- this returns True if all characters in s are whitespace
        - -> s.istitle() <- this returns True if s is a title case string
        - -> s.isupper() <- this returns False if all letters in s are uppercase
            - -> you can replace s with for example 'ANENTIRESTRING'
        - -> s.endswith('o') <- this checks if the string stored in the variable s ends in an 'o' character
            - -> this is the same as s[-1]=='o'
            - -> this asks if the last character of the string stored in variable `s` equal to 'o'


```
In [149]: s.partition('i')

Out[149]: ('h', 'i', 'ihhihihihhhi')
```


nbviewer   FAQ   IPython   Jupyter

Complete-Python-Bootcamp / Advanced Strings.ipynb

Advanced Strings

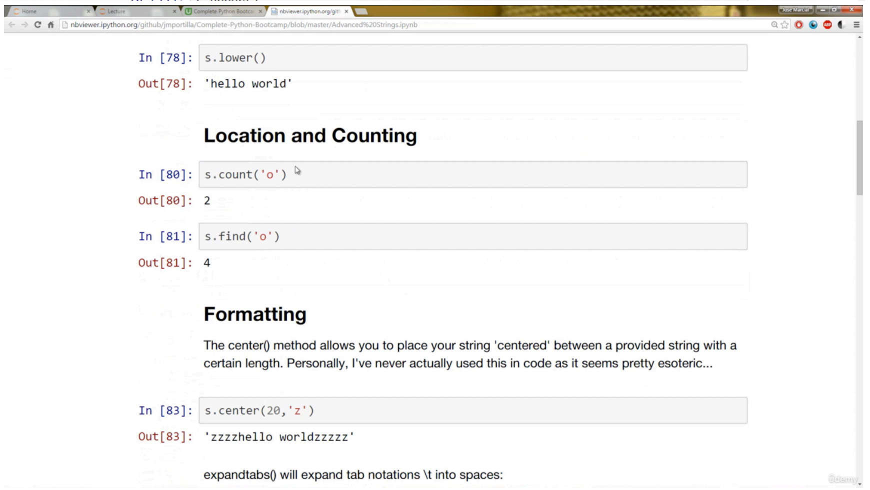
String objects have a vareity of methods we can use to save time and add functionality. Lets explore some of them in this lecture:

```
In [75]: s = 'hello world'
```

Changing case

We can use methods to capitalize the first word of a string, change cases to upper and lower case strings.

```
In [76]: # Capitalize first word in string
         s.capitalize()

Out[76]: 'Hello world'
```


```
In [78]: s.lower()

Out[78]: 'hello world'
```

Location and Counting

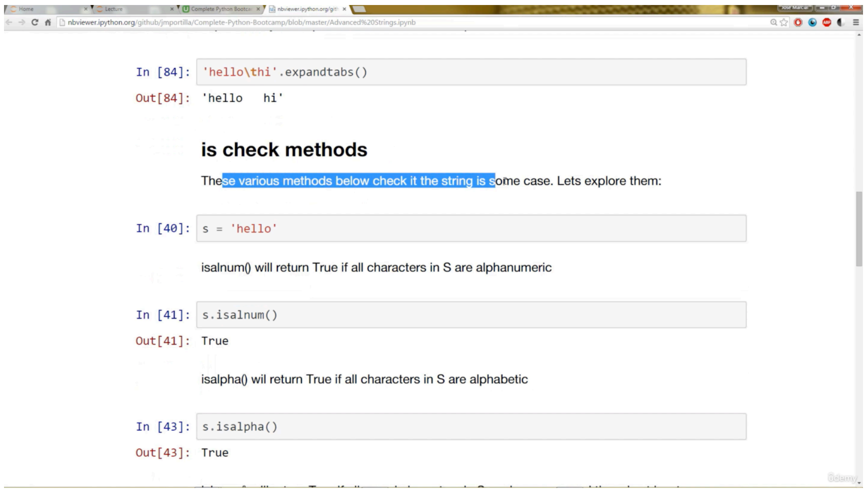
```
In [80]: s.count('o')

Out[80]: 2

In [81]: s.find('o')

Out[81]: 4
```

Formatting

The center() method allows you to place your string 'centered' between a provided string with a certain length. Personally, I've never actually used this in code as it seems pretty esoteric...

```
In [83]: s.center(20,'z')

Out[83]: 'zzzzhello worldzzzzz'
```

expandtabs() will expand tab notations \t into spaces:


```
In [84]: 'hello\thi'.expandtabs()

Out[84]: 'hello   hi'
```

is check methods

These various methods below check it the string is some case. Lets explore them:

```
In [40]: s = 'hello'
```

isalnum() will return True if all characters in S are alphanumeric

```
In [41]: s.isalnum()

Out[41]: True
```

isalpha() wil return True if all characters in S are alphabetic

```
In [43]: s.isalpha()

Out[43]: True
```

- **-> built-in regular expression operations**
    - -> builtin methods for operating on strings
    - **-> splitting**
        - -> this splits the string at a certain element
        - -> then returns a list of the results from this
        - -> s.split('e')
            - -> this takes the string stored in the `s` variable and returns an array
            - -> the first element is everything before 'e' in the string and the second is everything after it
            - -> if there were n instances of 'e' in `s`, then the array this method returns will be 1xn
    - **-> partition**
        - -> s.partition('i')

        - -> this returns a 1x3 array
        - -> the first element of this is everything before where 'i' is in the `s` string
        - -> the middle element of this is the partition element ('i')
        - -> then the last element of this array is everything after the partition in the 's' string
        - -> the middle element is called the 'separator'

    - -> partition only works at the first instance the string is found
        - -> splitting is for every instance

- **-> review**
    - -> boolean checks in Python
    - -> methods for changing case <- .capitalise(), .upper() and .lower()
    - -> methods for location and counting <- .count() and .find()
    - -> methods for esoteric formatting <- .center() and .expandtabs()
    - -> is check methods <- .isalnum(), .islower(), .isspace(), .istitle() and .endswith()
    - -> builtin regular expressions for string objects <- .split(), .partition()
    - -> being able to use these and understand them in someone else’s code