05_strings

January 26, 2018

1 Strings

1.0.1 Introduction

```
In [1]: # A string is a sequence of characters

        fruit = "banana"
        fruit[0]

Out[1]: 'b'

In [2]: fruit[-2] #second letter from the right

Out[2]: 'n'

In [3]: len(fruit)

Out[3]: 6
```

1.0.2 Traversal through a string with a loop

```
In [4]: index = 0 # begining of the string

        while index < len(fruit):
            print(fruit[index])
            index = index + 1

b
a
n
a
n
a


In [5]: # or we can use a for loop

        for x in fruit:
            print(x)
```

```
b
a
n
a
n
a
```

1.0.3 String Slicing

The slicing operator [n:m] returns the part of the string from the n^{th} character to the m^{th}, including the n^{th} and **execluding** the m^{th}.

```
In [6]: print(fruit)
        print(fruit[2:4]) # including the character at index 2 up to and excluding element wit
```

```
banana
na
```

```
In [7]: fruit[:3]
```

```
Out[7]: 'ban'
```

```
In [8]: fruit[3:] #including element with index 3 up to the end of the string
```

```
Out[8]: 'ana'
```

```
In [9]: fruit[3:3]
```

```
Out[9]: ''
```

1.0.4 String are immutable

You can't change an existing strings

```
In [10]: name = "Jilary"
         #name[0] = "H" # gives an error because you can't change strings.
```

Write a function count_a(word) that counts and returns the number of "a" in a given string.

```
In [11]: def count_a(word):
             count = 0
             for char in word:
                 if(char == "a" or char =="A"):
                     count = count + 1
             print(count)
```

```
In [12]: count_a("America")
```

```
2
```

1.0.5 String Methods

Python has the function `dir()` that lists the methods available for an object.

```
In [13]: fruit = "banana"
```

```
In [14]: dir(fruit)
```

```
Out[14]: ['__add__',
          '__class__',
          '__contains__',
          '__delattr__',
          '__dir__',
          '__doc__',
          '__eq__',
          '__format__',
          '__ge__',
          '__getattribute__',
          '__getitem__',
          '__getnewargs__',
          '__gt__',
          '__hash__',
          '__init__',
          '__init_subclass__',
          '__iter__',
          '__le__',
          '__len__',
          '__lt__',
          '__mod__',
          '__mul__',
          '__ne__',
          '__new__',
          '__reduce__',
          '__reduce_ex__',
          '__repr__',
          '__rmod__',
          '__rmul__',
          '__setattr__',
          '__sizeof__',
          '__str__',
          '__subclasshook__',
          'capitalize',
          'casefold',
          'center',
          'count',
          'encode',
          'endswith',
          'expandtabs',
          'find',
          'format',
          'format_map',
          'index',
          'isalnum',
          'isalpha',
          'isdecimal',
          'isdigit',
          'isidentifier',
          'islower',
          'isnumeric',
          'isprintable',
          'isspace',
          'istitle',
          'isupper',
          'join',
          'ljust',
          'lower',
          'lstrip',
          'maketrans',
          'partition',
          'replace',
          'rfind',
          'rindex',
          'rjust',
          'rpartition',
          'rsplit',
          'rstrip',
          'split',
          'splitlines',
          'startswith',
          'strip',
          'swapcase',
          'title',
          'translate',
          'upper',
          'zfill']
```

```
In [15]: print(fruit.upper())
```

```
BANANA
```

```
In [16]: fruit.find("a") #locates where the first occurance of a given character is
```

```
Out[16]: 1
```

```
In [17]: fruit.find("nan") # locates where a substring strarts
```

```
Out[17]: 2
```

```
In [18]: fruit.find("a", 2) # start searching from index 2 for the letter "a"

Out[18]: 3

In [19]: line = " Hello, World! "
         line.strip() # gets rid of white spaces from the begining and end

Out[19]: 'Hello, World!'

In [20]: line = line.strip()
         line

Out[20]: 'Hello, World!'

In [21]: line.startswith("H")

Out[21]: True

In [22]: line.startswith("hello")

Out[22]: False

In [23]: line.lower().startswith("hello") # change all to lower case, and then search

Out[23]: True
```