06_Lists

January 26, 2018

1 Lists

1.0.1 Introduction

```
In [1]: entries = [1, 2, 3]
        entries

Out[1]: [1, 2, 3]

In [2]: entries = [1, 2, 3, "Python"]
        entries

Out[2]: [1, 2, 3, 'Python']

In [3]: entries[2] + 1

Out[3]: 4

In [4]: entries = [1, 2, 3, ["R", "Python"]] # nested list

In [5]: entries

Out[5]: [1, 2, 3, ['R', 'Python']]
```

Unlike strings, lists are mutable. This means that you can change elements in it

```
In [6]: entries[0] = 100
        entries

Out[6]: [100, 2, 3, ['R', 'Python']]

In [7]: entries[3] # gives you the sublist

Out[7]: ['R', 'Python']

In [8]: # What if I wanted to access "Python"?

        entries[3][1]

Out[8]: 'Python'
```

1.0.2 Traversing a List

```
In [9]: # we can use a for or while loop

        for entry in entries:
            print(entry)
```

```
100
2
3
['R', 'Python']
```

1.0.3 List Operators

```
In [10]: # Concatenate two lists using the +
         a = [1, 2, 3]
         b = [4, 5, 6]

         a + b
```

```
Out[10]: [1, 2, 3, 4, 5, 6]
```

```
In [11]: # Repeat a list n times using the *

         a*4
```

```
Out[11]: [1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3]
```

1.0.4 List Slicing

```
In [12]: entries

Out[12]: [100, 2, 3, ['R', 'Python']]

In [13]: entries[1:3] #prints element at index 1 up to and excluding element at index 3

Out[13]: [2, 3]

In [14]: entries[:3]

Out[14]: [100, 2, 3]

In [15]: entries[0:5:2] # lists every other element [start:end:step]

Out[15]: [100, 3]

In [16]: entries[::2]

Out[16]: [100, 3]

In [17]: entries[:-1] # without the last element

Out[17]: [100, 2, 3]

In [18]: entries[::-1] # list in reverse order

Out[18]: [['R', 'Python'], 3, 2, 100]
```

1.0.5 List Methods

```
In [19]: # append or add an element
         entries.append(300)
         entries

Out[19]: [100, 2, 3, ['R', 'Python'], 300]

In [20]: # extend appends a list and not just one element

         a = [4, 2, 0, 8]
         b = [10, 5, 2, 6]

         a.extend(b)
         a

Out[20]: [4, 2, 0, 8, 10, 5, 2, 6]

In [21]: # sorting a list
         a.sort()
         a

Out[21]: [0, 2, 2, 4, 5, 6, 8, 10]

In [22]: a.sort(reverse = True)
         a

Out[22]: [10, 8, 6, 5, 4, 2, 2, 0]

In [23]: # deleting elements from a list

         print(a)
         del a[0]
         print(a)

[10, 8, 6, 5, 4, 2, 2, 0]
[8, 6, 5, 4, 2, 2, 0]


In [24]: # what if you know what you want to delete? Suppose that
         # I want to delet the 2's

         a.remove(2)
         print(a)

[8, 6, 5, 4, 2, 0]


In [25]: # The pop() function deletes an element at a certain index,
         # and returns that element

         x = a.pop(2)
         print(x)
         print(a)
```

```
5
[8, 6, 4, 2, 0]
```

1.0.6 Useful Functions for Lists

```
In [26]: a = [4, 5, 61, 89]

In [27]: len(a)

Out[27]: 4

In [28]: max(a)

Out[28]: 89

In [29]: sum(a)

Out[29]: 159
```

Rewrite the function the computes the average of a list of numbers entered by the user using a list numlist = [] # always initialize the list

while True: inp = input("Enter a number: ") if inp == "done": break value = float(inp) numlist.append(value)

average = sum(numlist)/len(numlist) print("The average is:", average)

1.0.7 Lists and Strings

A string is a sequence of characters and a list is a sequence of values, but a list of characters is not the same as a string.

```
In [30]: # convert from a string to a list of characters

         word = "Hello World"
         word.split()

Out[30]: ['Hello', 'World']

In [31]: words = "Hello,world,this,is,python"
         words.split(",") #specify delimeter if not space.

Out[31]: ['Hello', 'world', 'this', 'is', 'python']

In [32]: a

Out[32]: [4, 5, 61, 89]
```