Scientific Programming With Python

Collections: Strings, Lists

UNIVERSITY of HOUSTON
DIVISION OF RESEARCH
HEWLETT PACKARD ENTERPRISE DATA SCIENCE INSTITUTE

Collection Data Types

- ints, floats, bool, complex …. are all scalar types
  - Store only one value
- Collection objects can hold more than one value
- Two kinds of collections, based on how the values are accessed
  - Sequence: access by positional index
    - (str)ing, list, tuple
  - Mapped: access by key
    - (dict)ionary

Collections and Strings

A string is a *collection* data type – those are composed of smaller pieces

- **as are `lists`, `tuples`, `dictionaries`**
- **`int`, `float`, `bool` are primitive data types**

A string is a sequential *collection of characters*

- 'Hello World!' `or "Hello World!"`
- `Or an empty string "`

String operations

Addition and multiplications have different meanings:

```
lastname= 'Doe'
firstname = 'John'
fullname = firstname + lastname    → 'JohnDoe'
silly = 3*lastname                 → 'DoeDoeDoe'

firstname-1 or '34'+2      -> are illegal/not aqllowed
```

String Indexing

Index of an item is a position of the item in a string

```
s = 'Python'
```

`s[0] == 'P', s[1]=='y' … s[5] == 'n'`

Interestingly, a negative index is used to specify a position with respect to the “end”

The last item has index -1,

The second to last item has index -2,…

```
s[-1]  == 'n'
s[-3]  == 'h'
```

String Methods

- Strings are objects with attributes and methods.
- `ss = 'PythonGood'`
-
- `ss.upper()` → `PYTHONGOOD`
- `ss.lower()` → `pythongood`
- `ss.count('o')` → `3`
- `ss.find('o')` → `4`
- `ss.rfind('o')` → `8`

String Methods

Methods that return bool: True or False

Method	Description
isalnum()	Returns True if all characters in the string are alphanumeric
isalpha()	Returns True if all characters in the string are in the alphabet
isdecimal()	Returns True if all characters in the string are decimals
isdigit()	Returns True if all characters in the string are digits
isidentifier()	Returns True if the string is an identifier
islower()	Returns True if all characters in the string are lower case
isnumeric()	Returns True if all characters in the string are numeric
isprintable()	Returns True if all characters in the string are printable
isspace()	Returns True if all characters in the string are whitespaces
istitle()	Returns True if the string follows the rules of a title
isupper()	Returns True if all characters in the string are upper case

Source https://www.w3schools.com/

String Methods

Methods that return bool -> True or False

Method	Description
endswith()	Returns True if the string ends with the specified value
startswith()	Returns True if the string starts with the specified value

Source https://www.w3schools.com/

String Methods

Methods that return a modified view of the string

Method	Description
capitalize()	Converts the first character to upper case
casefold()	Converts string into lower case
lower()	Converts string into lower case
upper()	Converts a string into upper case
title()	Converts the first character of each word to upper case
swapcase()	Swaps cases, lower case becomes upper case and vice versa
translate()	Returns a translated string
rjust()	Returns a right justified version of the string
ljust()	Returns a left justified version of the string
zfill()	Fills the string with a specified number of 0 values at the beginning

Source https://www.w3schools.com/

String Functions and Operators

- `ss = 'PythonGood'`
- Length function `len(ss)` → `10`
- String slices `([n:m]-- substring from n to m-1)`
- `ss[0:6]` → `'Python'`
- `ss[6:10]` → `'Good'`
- in and not in (if one string is a substring of other)
- `'n' in 'Python'` → `True`
- `'n' not in 'Python'` → `False`
- `'' in 'Python'` → `True`
- `'Python' in 'Python'` → `True`

String Comparison

```
ss = 'PythonGood'
      ss == 'PythonGood'          True
      ss == 'pythongood'          False
      'Python'< 'Java'  ?
      'Python' < 'Scala'?
      'Python' < 'python' ?
      (lexicographic)
ord() and chr() functions
>>> ord('n')
 97
>>> chr(97)
 'a'
```

Strings are Immutable

- Elements of strings cannot be modified
- `ss = 'PythonGood'`
- `ss[0] = 'p'` → `error`

- However
- `newss = 'p' + ss(1:10)` → 'pythonGood'

Strings Constants

- provided by `string` module

- `string.ascii_lowercase`
- `string.ascii_uppercase`
- `string.digits`
- `string.punctuations`

Lists: Basics

- List is a sequential collection of Python Data Items.
- Like strings, except the list items can be any type, even strings or even other lists
- `pets = ['ant', 'bat', 'cod', 'dog', 'elk']`
- `lst = [0, 1, 'two', 'three', [4, 'five']]`
- `nums = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]`

- A list within a list is nested – inner list can be referred to as sublist

Accessing List Elements

- List items are accessed through indexes
- `pets = ['ant', "bird", 'cod', 'dog', 'elk']`
- `pets[1] = 'bird'`

- `mixlist = [44, 'bird', 12.9, [False, 'dog'], True ]`

- `mixlist[2]` → `12.9`
- `mixlist[3]` → `[False, 'dog']`
- `mixlist[3][1]` → `'dog'`

Common List Operations

- Many operations are similar to strings: `len, in and not in, concatenation (+), repetition, slicing`
- `mixlist = [44, "bird", 12.9, [False, 'dog'], True ]`
- `len(mixlist)` → `5`

- Operations directly work on the top level of the list, not the nested elements.
- `len(mixlist[3])` → `2`
- `len(mixlist[3][1])` → `3`

Lists are Mutable

```
mixlist = [44, 'bird', 12.9, [False,
'dog'], True ]
```

```
mixlist[1] = 'animal'
```

→ `[44, 'animal', 12.9, [False, 'dog'], True ]`

```
mixlist[3:5] = []                #deletion
```

→ `[44, 'animal', 12.9 ]`

List Methods: Adding and Removing elements

- **`lst.append(item):`** Adds item to the end
- `lst.insert(position,item):` Adds item at position
- **`lst.pop(position):`** Removes and returns the item at postions – last item by default
- `lst.sort(), lst.reverse()`

- **`lst.remove(item):`** removes first occurrence of item
- `lst.index(item):` return pos of first occurrence of item
- `lst.count(item):` return # of occurrences of the item

Classwork

- Write a program to take a number K as input, reads K names (one at a time), store them in a list, and then print them.