python basics

variables

```
x = 2

string = ‘hello world’
```

Python is a dynamically typed language. Variable types aren’t explicitly mentioned like in C/C++.

data types

int: 1, -3, 42

float: 8.6, 3.0, 31e12

complex: 3+2j, -4+7j

bool: True, False

string: ‘turing’, “bourne again”, “a”, “b”

list

A list is made of a mixture of other types as it’s elements. This can include numbers, strings, dictionaries and even other lists.

```
x = [1, “sdfsdf”, 6.7, [1, 2, 3]] # list containing different elements

y = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

l = [] # empty list

l.append(24) # l = [24]

l.append(7.8) # l = [24, 7.8]
```

list

```
x = ["first", "second", "third", "fourth"]

x[0] # 'first'

x[2] #'third'

x[-1] # 'fourth'

x[-2] # 'third'

x[0:3] # ['first', 'second', 'third']
```

tuples

Similar to lists but they are immutable i.e. they can’t be changed once they have been created.

```
()
```

```
(1,)
```

```
(1, 2, 3, 4, 5, 6, 7, 8, 12)
```

```
(1, "two", 3L, 4.0, ["a", "b"], (5, 6))
```

strings

```
s1 = ‘this is a string.’

s2 = “this is also a string.”

s3 = s1 + s2 # ‘this is a string.this is also a string.’

s1.split() # [‘this’, ‘is’, ‘a’, ‘string.’]

s1.replace(‘a’, ‘the’) #  ‘this is the string.’

x = ‘add here: {} and here: {}’

x.format(‘word’, ‘something’) # x = ‘add here: word and here: something’
```

dictionaries

Dictionaries are a collection of key-value pairs.

```
x = {1: "one", 2: "two"} # 1 is the key and “one” is the value

x[3] = “three” # adding a new key-value pair to the dictionary
```

You can access values in a dictionary by it’s key.

```
x[2] # gives “two”

x[4] # gives error
```

if-elif-else

```
x = 5

if x < 5:

    y = -1

elif x > 5:

    y = 1

else:

    y = 0

print(x, y)
```

while loop

```
x = 1

while x < 5:

    print(x)

    x = x + 1
```

for loop

Different from for loops in C, C++ or Java.

A ‘for loop’ in python iterates over the elements of any iterable type.

strings, lists, dictionaries etc. are all iterable types.

```
l = [1, 2, 6, 7, 9]

for element in l:

    print(element)
```

functions in python

Defining a function:

```
def func(a, b, c):
    ...do something...
    return some_value
```

Calling functions:

```
func(1, 2, 3)
func(a=1, b=2, c=3)
```

file handling

```
f = open(“file_name.txt”, “r”) # r stands for read mode

line = f.readline()

f.close()


f.open(“file_name”, “wb”) # w stands for write, b for binary

f.write(data)

f.close()
```

modules

Any external functionality that is not a part of the core python language can be brought in using modules.

A module is basically a collection of programs.

```
import requests

from bs4 import BeautifulSoup
```

web scraping

what is web scraping?

Web Scraping is a technique employed to extract large amounts of data from websites.

BeautifulSoup is a python library for pulling data out of HTML and XML files.

It is used extensively in web scraping.

Basic Syntax of HTML

```
<!DOCTYPE html>

<html>

<body>

    <h1> My first heading</h1>

    <p> My first paragraph</p>

</body>

</html>
```

Accessing HTML content using requests

```
import requests

URL = ”example.com”

r = requests.get(URL).content

print(r)
```

Parsing the HTML content

```
import requests

from bs4 import BeautifulSoup

URL = ”example.com”

r = requests.get(URL).content

soup = BeautifulSoup(r, ‘lxml’)

print(soup.prettify())
```

BeautifulSoup example

Demo script

resources

- Python documentation: https://docs.python.org/3/
- BeautifulSoup: https://www.crummy.com/software/BeautifulSoup/bs4/doc/
- For the final script:
  - For asynchronous programming: https://aiohttp.readthedocs.io/en/stable/
  - For multiproccessing: https://docs.python.org/2/library/multiprocessing.html

- All codes and these slides: github.com/LUGM