Introduction to Python

CMPT 498/820 Machine Learning
Tutorial 1

Najeeb Khan

September 14, 2016

1 Python

- Python is an interpreted programming language
- Whitespace in Python is important
- No curly braces! indentation is used to structure the code.
- Comments start with a #

2 Tools

2.1 Anaconda

Available at: https://docs.continuum.io/anaconda/install

2.2 PyCharm Edu

Available at: https://www.jetbrains.com/pycharm-edu/

3 Hello World in Python

There are many ways one can run a Python program, for example,

- using a command line interface
- using an Integerated Development Environment (IDE)
- using Python Notebooks

This document was created using Python Notebook. To run Python Notebook, navigate to the directory you want to save your notebooks in and type the following in a terminal: `$> jupyter notebook`

4 Python Keywords

There are 33 keywords in Python 3.5

4.1 Importing Packages

4.1.1 import

```
In [1]: import keyword
        print(keyword.kwlist[1:8])
```

```
['None', 'True', 'and', 'as', 'assert', 'break', 'class']
```

4.1.2 from

```
In [2]: from keyword import kwlist
        print(kwlist[1:8])
```

```
['None', 'True', 'and', 'as', 'assert', 'break', 'class']
```

4.1.3 as

```
In [3]: from keyword import kwlist as words
        print(len(words))
```

```
33
```

4.2 Conditonals

4.2.1 if

```
In [4]: score=90
        if score > 70:
            print('Congrats!')
        print('HAPPY OR SAD, THIS WILL BE PRINTED ANYWAY!')
```

```
Congrats!
HAPPY OR SAD, THIS WILL BE PRINTED ANYWAY!
```

4.2.2 else

```
In [5]: score=60
        if score > 70:
            print('Congrats!')
        else:
            print('Good luck next term!')
        print('HAPPY OR SAD, THIS WILL BE printED ANYWAY!')
```

```
Good luck next term!
HAPPY OR SAD, THIS WILL BE printED ANYWAY!
```

4.2.3 elif

```
In [6]: score=85
        if score > 90:
            print('You are the best (probably)!')
        elif score > 80:
            print('Good job!')
        else:
            print('Good luck next term!')
        print('Python doesn\'t have a switch statement!')
```

```
Good job!
Python doesn't have a switch statement!
```

4.3 Loops

4.3.1 in

```
In [7]: student='john'
        if student in ['steve','alex','bob','john']:
            print(student + ' is enrolled!')
```

```
john is enrolled!
```

4.3.2 for

```
In [8]: for s in ['steve','alex','bob','john']:
            print(s)
```

```
steve
alex
bob
john
```

```
In [9]: for id in range(5):
            print(id)
```

```
0
1
2
3
4
```

```
In [10]: print('ID \t Name')
         for (id,s) in  zip(range(4), ['steve','alex','bob','john']):
             print(str(id)+'\t '+s)
```

```
ID          Name
0         steve
1         alex
2         bob
3         john
```

4.3.3 while

```
In [11]: life=3
         while life > 0:
             print('I am alive with '+str(life)+' stars')
             life-=1
         print('And finally I\'m dead!')
         print('Python doesn\'t have an increment/decrement operator')
```

```
I am alive with 3 stars
I am alive with 2 stars
I am alive with 1 stars
And finally I'm dead!
Python doesn't have an increment/decrement operator
```

4.3.4 break

```
In [12]: for s in ['steve','alex','bob','john']:
             if s == 'bob':
                 print('AHAA! We\'ve found '+s+'!' )
                 break
             else:
                 print('Nope! It is '+s+'!')
```

```
Nope! It is steve!
Nope! It is alex!
AHAA! We've found bob!
```

4.3.5 continue

```
In [13]: for s in ['steve','alex','bob','john']:
             if s == 'bob':
                 continue
             print('Welcome '+s+'!' )
         print('We just ignored bob!')
```

```
Welcome steve!
Welcome alex!
Welcome john!
We just ignored bob!
```

4.4 Organizing Code

4.4.1 def

```
In [14]: def product(a,b):
             """Computes the product of two numbers using addition."""
             p=0
             for i in range(a):
                 p=p+b
             print(p)

         # This is how you call a function
         product(4,5)
```

```
20
```

4.4.2 return

```
In [15]: def product(a,b):
             """Computes the product of two numbers using addition."""
             p=0
             for i in range(a):
                 p=p+b
             return p

         # This is how you return a value from a function
         result = product(4,5)
```

4.4.3 pass

Just a placeholder operation

4.4.4 class

```
In [16]: class student:
             """A class representing a student."""
             def __init__(self, name, course):
                 """Initializer for the student class."""
                 self.name=name
                 self.course=course

             def printName(self):
                 print(self.name)

             def aStaticMethod():
                 print('A static method')
```

```
         s=student('foo','CS')

         s.printName()
         student.printName(s)
         student.aStaticMethod()
```

```
foo
foo
A static method
```

4.5 Exception Handeling

4.5.1 try, except, finally

```
In [17]: f=open('onionRings.txt')
```

```
        ---------------------------------------------------------------------------------

        FileNotFoundError                         Traceback (most recent call last)

        <ipython-input-17-57b6e9fbdd29> in <module>()
    ----> 1 f=open('onionRings.txt')


        FileNotFoundError: [Errno 2] No such file or directory: 'onionRings.txt'
```

```
In [18]: try:
             f=open('onionRings.txt')
         except FileNotFoundError:
             print('Sorry! Onion rings not found >_<')
             print('Would you like french fries instead?')
         finally:
             print('You\'ll have to pay for the pop anyway!')
```

```
Sorry! Onion rings not found >_<
Would you like french fries instead?
You'll have to pay for the pop anyway!
```

4.5.2 assert, raise

```
In [19]: age = int( input('Please enter your age: ') )
         assert(age > 21)
         print('Welcome!')
```

```
Please enter your age: 20
```

```
        ------------------------------------------------------------------------------

        AssertionError                            Traceback (most recent call last)

        <ipython-input-19-d6c1e4e071d0> in <module>()
          1 age = int( input('Please enter your age: ') )
    ----> 2 assert(age > 21)
          3 print('Welcome!')


        AssertionError:
```

4.6 Types

4.6.1 True, False, None

4.7 Logical

4.7.1 and, or, not

4.8 Scope

4.8.1 with, nonlocal, global, del

4.9 Miscellaneous

4.9.1 lambda, is, yield

5 Sequence types

5.1 Tuples ( )

Tuples and strings are immutable sequence types

```
In [20]: t=('abc', 123, [1,2,3])
         print(t[2])
```

```
[1, 2, 3]
```

5.2 Strings " "

```
In [21]: s="hello world"
         print(s[6])
```

```
w
```

```
In [22]: newS = s.replace('world','saskatoon')
         print(s)
         print(newS)
```

```
hello world
hello saskatoon
```

5.3 Lists [ ]

Lists are mutable types.

```
In [23]: a=[1, 2, [3,4], 'abc', (0,'d'), 7]
         print(a[3])
```

```
abc
```

```
In [24]: a.append('hi')
         a.remove([3,4])
         print(a)
```

```
[1, 2, 'abc', (0, 'd'), 7, 'hi']
```

5.4 Indexing Sequence types

```
In [25]: print(a[1:3])
```

```
[2, 'abc']
```

```
In [26]: print(a[:3])
```

```
[1, 2, 'abc']
```

```
In [27]: print(a[-2])
```

```
7
```

```
In [28]: print(a[3:])
```

```
[(0, 'd'), 7, 'hi']
```

5.5 Dictionaries

Dictionaries store key value pairs.

```
In [29]: d = {'a': 65, 'b': 66, 'c': 67}
         print(d['b'])
```

```
66
```

6 List Comprehension

List comprehension is one of the most powerful concept of Python.

Get a list of the successful student IDs given a list of marks.

```
In [30]: successful = []
         marks=[90, 45, 92, 85, 94, 39, 71, 86]
         for ID in range( len(marks) ):
             if marks[ID] > 70:
                 successful.append(ID)
         print(successful)
```

```
[0, 2, 3, 4, 6, 7]
```

```
In [31]: successful = [ID for ID in range(len(marks)) if marks[ID] > 70]
         print(successful)
```

```
[0, 2, 3, 4, 6, 7]
```

```
In [32]: successful = [(ID,marks[ID]) for ID in range(len(marks)) if marks[ID] > 70
         print(successful)
```

```
[(0, 90), (2, 92), (3, 85), (4, 94), (6, 71), (7, 86)]
```

7 Files I/O

```
In [33]: s=['This is line 1.','This is line 2','This is line 3']
         f=open('myFile.txt','w') # open a file in write mode
         for l in s:              # write the lines to file
             f.write(l+'\n')
         f.close()                # close the file
```

```
In [34]: with open('myFile.txt','r') as f: # using 'with' terminates f after use
             s=f.read()
             #s=f.readlines()
         print(s)
```

```
This is line 1.
This is line 2
This is line 3
```