DSE5002 Module 5 Pair Programming

Beija Richardson 3/20/25

Intro to some Python Ideas

Basic Variables

Python is dynamically typed, like R, so Python decides how to store variables when you declare them.

Python has floating points, integers, complex numbers, boolean and strings as the basic type elements

It does not have a factor like R does

The basic math operations work the same way they do in R

In [56]:
```
a=5
b=2
print(a+b)
```

```
7
```

In [57]:
```
#We can use the type function to see what type of variable on object is
type(a)
```

Out[57]:
```
int
```

In [58]:
```
a=5.0
type(a)
```

Out[58]:
```
float
```

Question

Why is a an integer in one case and a float in the other?

In [60]:
```
a="5"
type(a)
```

Out[60]:
```
str
```

Storage classes in Python, such as str and more complex types have built in member functions that operate on items in the class

The dir() function will show the variables in the class, which have underscores added, and functions which do not

In [62]:
```
dir(a)
```

Out[62]:
```
['__add__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__getnewargs__',
 '__getstate__',
 '__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',
 'isascii',
 'isdecimal',
 'isdigit',
 'isidentifier',
 'islower',
 'isnumeric',
 'isprintable',
 'isspace',
 'istitle',
```

```
 'isupper',
 'join',
 'ljust',
 'lower',
 'lstrip',
 'maketrans',
 'partition',
 'removeprefix',
 'removesuffix',
 'replace',
 'rfind',
 'rindex',
 'rjust',
 'rpartition',
 'rsplit',
 'rstrip',
 'split',
 'splitlines',
 'startswith',
 'strip',
 'swapcase',
 'title',
 'translate',
 'upper',
 'zfill']
```

Looking at a string, there is along list of available functions

We'll define a more interesting string and see what some of these functions do.

Notice how the member functions are called, as the variable name, a period and then the function name and parenthesis

Not all functions in Python are member functions, this is just showing how to call member functions once you find them using dir()

In [64]:
```
a="Joe chased a leaf,"
print(a.upper())
print(a.lower())
print(a.title())
```

```
JOE CHASED A LEAF,
joe chased a leaf,
Joe Chased A Leaf,
```

In [65]:
```
a.split()
```

Out[65]:
```
['Joe', 'chased', 'a', 'leaf,']
```

The key idea here is that dir() can show you a bunch of useful functions available

In [67]:
```
a.__len__
```

Out[67]:
```
<method-wrapper '__len__' of str object at 0x000001E291903170>
```

In [68]:
```
#this is an iPython "magic" command to see the user defined variables in use at the
# it is a lift from the Matlab system
# "magic" commands are utility commands in iPython or Jupyter, not part of python

%who
```

```
Boston_roll      a       b       dataframe_columns       dataframe_hash  dtypes_str
get_dataframes   getpass         hashlib
import_pandas_safely     infile  is_data_frame   json    np      pd      x
```

In [69]:
```
%whos
```

```
Variable               Type         Data/Info
-------------------------------------------
Boston_roll            DataFrame               _id         PI<...>182242 rows x 66 c
olumns]
a                      str          Joe chased a leaf,
b                      int          2
dataframe_columns      function     <function dataframe_colum<...>ns at 0x000001E28C
F982C0>
dataframe_hash         function     <function dataframe_hash at 0x000001E28CF98860>
dtypes_str             function     <function dtypes_str at 0x000001E28CF98220>
get_dataframes         function     <function get_dataframes at 0x000001E28CF98D60>
getpass                module       <module 'getpass' from 'C<...>conda3\\Lib\\getpa
ss.py'>
hashlib                module       <module 'hashlib' from 'C<...>conda3\\Lib\\hashl
ib.py'>
import_pandas_safely   function     <function import_pandas_s<...>ly at 0x000001E28C
F99260>
infile                 str          C:\Users\Luke\Documents\C<...>_Assessment_Roll_2
024.csv
is_data_frame          function     <function is_data_frame at 0x000001E28CF98900>
json                   module       <module 'json' from 'C:\\<...>\Lib\\json\\__init
__.py'>
np                     module       <module 'numpy' from 'C:\<...>ges\\numpy\\__init
__.py'>
pd                     module       <module 'pandas' from 'C:<...>es\\pandas\\__init
__.py'>
x                      matrix       [[1 2 3]\n [4 5 6]\n [7 8 9]]
```

To see more about magic commands, see

https://ipython.readthedocs.io/en/stable/interactive/magics.html

Thee are 4 types of built-in data structures in basic python

list, tuple, dictionary and set

We'll talk about them next week

Numpy

Numpy stands for numerical python, it has array and vector data types defined within it

To create matrices and do matric operations in Python, people typically use Numpy

Many other structures and common python packages are built on top of numpy classes

To use Numpy, we have to import the package. Note that the package must already be installed in your environment

np is the classic appreciation or alias for numpy

In [73]:
```
import numpy as np
```

Numpy matrices

-must all have the same type of element

- are indexed by the row and column number

-but like most other language, the first row is 0, and the first column is also 0 with R indices start with 1, in python they start with 0

-think of the indices in python as the distance from the start of an array or matrix

In [75]:
```
x=np.matrix('1 2 3; 4 5 6; 7 8 9')
x
```

Out[75]:
```
matrix([[1, 2, 3],
        [4, 5, 6],
        [7, 8, 9]])
```

In [76]:
```
x[1,0]
```

Out[76]:
```
4
```

In [77]:
```
Action/Question

What is the index of the "8" in this matrix?

Test your answere
```

```
  Cell In[77], line 3
    What is the index of the "8" in this matrix?
                 ^
SyntaxError: invalid syntax
```

import numpy as np

x=np.matrix('1 2 3; 4 5 6; 7 8 9')

x

x[2,1]

= 8

In [ ]:
```
type(x)
```

In [ ]:
```
# dtype is an attribute of a numpy array that indicates the type of elements in the

x.dtype
```

In [ ]:
```
x.size
```

In [ ]:
```
x.shape
```

In [ ]:
```
dir(x)
```

Action

Add a cell and try some different operations from the member functions, see what they do

Specialized Matrices

In [ ]:
```
x=np.identity(6)
x
```

In [ ]:
```
x=np.ones((4,5))
x
```

In [ ]:
```
x=np.zeros((3,6))
x
```

To learn more about using Numpy to do linear algebra see:

https://numpy.org/doc/stable/user/absolute_beginners.html

Another option is

https://www.kaggle.com/code/legendadnan/numpy-tutorial-for-beginners-data-science

Pandas

Pandas is a libary that implies a data frame, much like the dataframes in R, or a data table in SQL

Each row is an observation, each column is a variable.

The columns are actually 1 dimensional numpy arrays (ie n x 1 matrics, for n rows)

There are an immense number of member functions to let us carry out operations on Pandas dataframes

Slicing, sorting and selecting work much like they do in R

We typically import pandas as pd

In [ ]:
```
import pandas as pd
```

edit this line so it contains the full path name for the Boston Asssessment_Roll_2024.cvs

infile="C:\Users\Luke\Documents\Class5002\

In [ ]:
```
#import the file,  place it into a Pandas dataframe

Boston_roll=pd.read_csv(infile)
```

In [ ]:
```
# We have a head function, just as in R

Boston_roll.head(5)
```

In [ ]:
```
#We can access columns by name, rowns by number
# Notice that 0:5 gives us all values from 0 to 4,  5 is not included

Boston_roll["CITY"][0:5]
```

In [ ]:
```
# we can index using two numbers, note the need to use the .iloc notation to do thi

Boston_roll.iloc[5:10,7]
```

In [ ]:
```
#let's look at the member functions

dir(Boston_roll)
```

Just a few options, eh?

Pandas can do a lot of different things....

We will see a lot more of Pandas later

If you want to work ahead a bit

https://www.kaggle.com/learn/pandas

https://pandas.pydata.org/docs/getting_started/tutorials.html