Data Science Cheat Sheet

Python Basics

BASICS, PRINTING AND GETTING HELP

`x = 3` - Assign **3** to the variable **x**

`print(x)` - Print the value of **x**

`type(x)` - Return the type of the variable **x** (in this case, **int** for integer)

`help(x)` - Show documentation for the **str** data type

`help(print)` - Show documentation for the **print()** function

READING FILES

```
f = open("my_file.txt","r")
file_as_string = f.read()
```

- Open the file **my_file.txt** and assign its contents to **s**

```
import csv
f = open("my_dataset.csv","r")
csvreader = csv.reader(f)
csv_as_list = list(csvreader)
```

- Open the CSV file **my_dataset.csv** and assign its data to the list of lists **csv_as_list**

STRINGS

`s = "hello"` - Assign the string **"hello"** to the variable **s**

```
s = """She said,
"there's a good idea."
"""
```

- Assign a multi-line string to the variable **s**. Also used to create strings that contain both **"** and **'** characters

`len(s)` - Return the number of characters in **s**

`s.startswith("hel")` - Test whether **s** starts with the substring **"hel"**

`s.endswith("lo")` - Test whether **s** ends with the substring **"lo"**

`"{} plus {} is {}".format(3,1,4)` - Return the string with the values **3**, **1**, and **4** inserted

`s.replace("e","z")` - Return a new string based on **s** with all occurances of **"e"** replaced with **"z"**

`s.split(" ")` - Split the string **s** into a list of strings, separating on the character **" "** and return that list

NUMERIC TYPES AND MATHEMATICAL OPERATIONS

`i = int("5")` - Convert the string **"5"** to the integer **5** and assign the result to **i**

`f = float("2.5")` - Convert the string **"2.5"** to the float value **2.5** and assign the result to **f**

`5 + 5` - Addition

`5 - 5` - Subtraction

`10 / 2` - Division

`5 * 2` - Multiplication

`3 ** 2` - Raise **3** to the power of **2** (or 3^2)

`27 ** (1/3)` - The **3**rd root of **27** (or $\sqrt[3]{27}$)

`x += 1` - Assign the value of **x + 1** to **x**

`x -= 1` - Assign the value of **x - 1** to **x**

LISTS

`l = [100,21,88,3]` - Assign a list containing the integers **100**, **21**, **88**, and **3** to the variable **l**

`l = list()` - Create an empty list and assign the result to **l**

`l[0]` - Return the first value in the list **l**

`l[-1]` - Return the last value in the list **l**

`l[1:3]` - Return a slice (list) containing the second and third values of **l**

`len(l)` - Return the number of elements in **l**

`sum(l)` - Return the sum of the values of **l**

`min(l)` - Return the minimum value from **l**

`max(l)` - Return the maximum value from **l**

`l.append(16)` - Append the value **16** to the end of **l**

`l.sort()` - Sort the items in **l** in ascending order

`" ".join(["A","B","C","D"])` - Converts the list [**"A"**, **"B"**, **"C"**, **"D"**] into the string **"A B C D"**

DICTIONARIES

`d = {"CA":"Canada","GB":"Great Britain", "IN":"India"}` - Create a dictionary with keys of **"CA"**, **"GB"**, and **"IN"** and corresponding values of of **"Canada"**, **"Great Britain"**, and **"India"**

`d["GB"]` - Return the value from the dictionary **d** that has the key **"GB"**

`d.get("AU","Sorry")` - Return the value from the dictionary **d** that has the key **"AU"**, or the string **"Sorry"** if the key **"AU"** is not found in **d**

`d.keys()` - Return a list of the keys from **d**

`d.values()` - Return a list of the values from **d**

`d.items()` - Return a list of **(key, value)** pairs from **d**

MODULES AND FUNCTIONS

The body of a function is defined through indentation.

`import random` - Import the module **random**

`from math import sqrt` - Import the function **sqrt** from the module **math**

```
def calculate(addition_one,addition_two,
exponent=1,factor=1):
  result = (value_one + value_two) ** exponent * factor
  return result
```

- Define a new function **calculate** with two required and two optional named arguments which calculates and returns a result.

`addition(3,5,factor=10)` - Run the **addition** function with the values **3** and **5** and the named argument **10**

BOOLEAN COMPARISONS

`x == 5` - Test whether **x** is equal to **5**

`x != 5` - Test whether **x** is not equal to **5**

`x > 5` - Test whether **x** is greater than **5**

`x < 5` - Test whether **x** is less than **5**

`x >= 5` - Test whether **x** is greater than or equal to **5**

`x <= 5` - Test whether **x** is less than or equal to **5**

`x == 5 or name == "alfred"` - Test whether **x** is equal to **5** or **name** is equal to **"alfred"**

`x == 5 and name == "alfred"` - Test whether **x** is equal to **5** and **name** is equal to **"alfred"**

`5 in l` - Checks whether the value **5** exists in the list **l**

`"GB" in d` - Checks whether the value **"GB"** exists in the keys for **d**

IF STATEMENTS AND LOOPS

The body of if statements and loops are defined through indentation.

```
if x > 5:
  print("{} is greater than five".format(x))
elif x < 0:
  print("{} is negative".format(x))
else:
  print("{} is between zero and five".format(x))
```

- Test the value of the variable **x** and run the code body based on the value

```
for value in l:
  print(value)
```

- Iterate over each value in **l**, running the code in the body of the loop with each iteration

```
while x < 10:
  x += 1
```

- Run the code in the body of the loop until the value of **x** is no longer less than **10**