→ CCPS 844 Data Mining Lab 1

Answer the following questions and submit a PDF file on the D2L.

You can consult the following link to revise/improve your Python programming skills $\,$

https://docs.python.org/3/tutorial/index.html

Let's begin by revising

- · Basic operations
- Data Structures
- · creating/using functions

What is 7 to the power of 4?

```
7**4

☐⇒ 2401

Split this string:

s = "Hi there Sam!"

into a list.

s = 'Hi there Sam!'
```

Given the variables:

s.split(' ')

```
planet = "Earth"
diameter = 12742
```

Use .format() to print the following string:

['Hi', 'there', 'Sam!']

```
The diameter of Earth is 12742 kilometers.

planet = "Earth"
diameter = 12742

print("The diameter of {} is {} kilometer".format(planet,diameter))

The diameter of Earth is 12742 kilometer
```

Given this nested list, use indexing to grab the word "hello"

Given this nest dictionary grab the word "hello". Be prepared, this will be annoying/tricky

What is the main difference between a tuple and a list?

```
# Tuple is immutable
```

Create a function that grabs the email website domain from a string in the form:

```
user@domain.com
```

So for example, passing "user@domain.com" would return: domain.com

```
def domainGet(email):
    domain=email.split("@")
    return domain[1]

domainGet('user@domain.com')
    'domain.com'
```

Create a basic function that returns True if the word 'dog' is contained in the input string. Don't worry about edge cases like a punctuation being attached to the word dog, but do account for capitalization.

```
def findDog(str):
    if "dog" in str:
        return True
    elif "Dog" in str:
        return True
    else:
        return False

findDog('Is there a dog here?')
    True
```

Create a function that counts the number of times the word "dog" occurs in a string. Again ignore edge cases.

```
def countDog(str):
    count=0
    words=str.split()
    for word in words:
        if word=="dog":
            count+=1
    return count

countDog('This dog runs faster than the other dog dude!')
2
```

Use lambda expressions and the filter() function to filter out words from a list that start with the letter 's'. For example:

```
seq = ['soup','dog','salad','cat','great']
```

should be filtered down to:

```
['soup','salad']
seq = ['soup','dog','salad','cat','great']
list(filter(lambda x: x.startswith('s'),seq))
        ['soup', 'salad']
```

Learning Pandas Basics

Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python.

http://pandas.pydata.org/pandas-docs/stable/user_guide/index.html

▼ Data Frame

DataFrame is a 2-dimensional labeled data structure with columns of potentially different types. You can think of it like a spreadsheet or SQL table, or a dict of Series objects. It is generally the most commonly used pandas object. There are several ways to create a DataFrame. One way way is to use a dictionary.

```
#Importing the Pandas library import pandas as pd
```

Create a dataframe df from the dictionary dict

	country	capital	area	population
0	Brazil	Brasilia	8.516	200.40
1	Russia	Moscow	17.100	143.50
2	Pakistan	Islamabad	3.286	1252.00
3	China	Beijing	9.597	1357.00
4	South Africa	Pretoria	1.221	52.98

```
indices = ["BR", "RU", "PK", "CH", "SA"]
```

Set the list indices as index of the dataframe brics

```
df.index=indices
df
```

	country	capital	area	population
BR	Brazil	Brasilia	8.516	200.40
RU	Russia	Moscow	17.100	143.50
PK	Pakistan	Islamabad	3.286	1252.00
СН	China	Beijing	9.597	1357.00
SA	South Africa	Pretoria	1.221	52.98

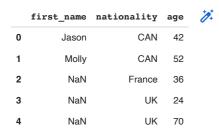
Select records from the dataframe indexed with 'RU' and 'SA'

```
df.loc[["RU", "SA"]]
```

	country	capital	area	population	10-
RU	Russia	Moscow	17.100	143.50	
SA	South Africa	Pretoria	1.221	52.98	

Create a dataframe from raw_data

```
df=pd.DataFrame(raw_data)
df
```



Select all elderly Canadians from the dataframe (age > 50)

elderly= df.loc[df['age']>50]
elderly

	first_name	nationality	age	1
1	Molly	CAN	52	
4	NaN	UK	70	

© Dr. Muhammad Naeem Irfan. Can't be posted on the internet

