Introduction of Data Sciene in Python

Week 1 In-Video Assignments

```
def add_numbers(x, y, z):
  return x+y+z

print(add_numbers(1, 2, 3))
```

```
6
```

```
def do_math(a, b, kind = None):
  if (kind=='add'):
    return a+b
  else:
    return a-b

out1 = do_math(1, 2)
print("out1 : ", out1)
out2 = do_math(1, 2, kind = "add")
print("out2 : ", out2)
```

```
out1 :  -1
    out2 :  3
```

```
x = 'Dr. Christopher Brooks'

# To get the output Christopher

print(x[4:15])
```

```
Christopher
```

```
'''can you write a function and
apply it using map() to get a list
of all faculties and last names'''

lst =['0', '1', '5', '2', '3']
lst.sort()
lst

people = ['Dr. Christopher Brooks', 'Dr. Kevyn Collins-Thompson', 'Dr. VG Vinod Vydiswaran

def split_title_and_name(person):
    firstname = person.split()[0]
    lastname  = person.split()[-1]
    return '{} {}'.format(firstname, lastname)

list(map(split_title_and_name, people))
```

```
['Dr. Brooks', 'Dr. Collins-Thompson', 'Dr. Vydiswaran', 'Dr. Romero']
```

```
# Convert this function using lambda

people = ['Dr. Christopher Brooks', 'Dr. Kevyn Collins-Thompson', 'Dr. VG Vinod Vydiswaran

def split_title_and_name(person):
    return person.split()[0] + ' ' + person.split()[-1]

#option 1
for person in people:
    print(split_title_and_name(person) == (lambda x: x.split()[0] + ' ' + x.split()[-1])(p

#option 2
list(map(split_title_and_name, people)) == list(map(lambda person: person.split()[0] + ' '
```

```
True
True
True
True
True
```

```
# Convert usong List Comprehension

# Example 1
def times_tables():
    lst = []
    for i in range(10):
        for j in range (10):
            lst.append(i*j)
    return lst

times_tables() == [j*i for i in range(10) for j in range(10)]

'''Imagine you work at an service
provider and the user ids are all
two letters and two numbers (eg.
aa07). your task at an organisation
you might to be hold a record on
the billing activity of an each
user'''


lowercase = 'abcdefghijklmnopqrstuvwxyz'
digits = '0123456789'

correct_answer = [a+b+c+d for a in lowercase for b in lowercase for c in digits for d in d

print(correct_answer[:50], end = ' ') # Display first 50 ids
```

```
['aa00', 'aa01', 'aa02', 'aa03', 'aa04', 'aa05', 'aa06', 'aa07', 'aa08', 'aa09'
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Regular_Expressions.txt to Regular_Expressions (1).txt
```

```
import re, sys
fname = sys.argv[1]
# with open(Regular_Expressions.txt, 'r') as file:
#    fstring = file.read()
fh = open("Regular_Expressions.txt")
fstring = fh.read()
num_list = re.findall('[0-9]+',fstring)
num_list = map(lambda x:(x), num_list)
print(sum(num_list))

fh.close()



# 1. Import regex
# 2. Read file
# 3. Create
# 4. Look for integers re.findall('[0-9]+', line)
# 5. Convert strings to integers
# 6. Sum integers

# 1. Import regex
import re

# 2. Read file
fhandle = open('Regular_Expressions.txt')

# 3. Create list
numlist = list()

# 4. Look for integers re.findall('[0-9]+', line)
for line in fhandle:
    line = line.rstrip()
    # Create lists of numbers
    num = re.findall('[0-9]+', line)

    # print num
    # confirm that numbers are collated

    # print num
    # shows max 3 in a list

    # Skip blank lists
    if len(num) < 1:
        continue

    elif len(num) == 1:
        # 5. Convert strings to integers
        num1 = int(num[0])
        numlist.append(num1)
    elif len(num) == 2:
        num1 = int(num[0])
        num2 = int(num[1])
        numlist.append(num1)
```

```
        numlist.append(num2)
    else:
        num1 = int(num[0])
        num2 = int(num[1])
        num3 = int(num[2])
        numlist.append(num1)
        numlist.append(num2)
        numlist.append(num3)

# 6. Sum integers in a list
sum_num_integer = sum(numlist)
print(len(numlist))
print(sum_num_integer)
```

```
---------------------------------------------
-----------------------------
TypeError
Traceback (most recent call last)
<ipython-input-46-709f0684f3fc> in <module>()
     50
     51 # 6. Sum integers in a list
---> 52 sum_num_integer = sum(numlist)
     53 print(len(numlist))
     54 print(sum_num_integer)

TypeError: 'int' object is not callable
```

SEARCH STACK OVERFLOW

```
import re

fh =open("Regular_Expressions.txt")

sum , count = 0 , 0

for line in fh:
    f = re.findall('[0 - 9]+', line)
    for num in f:
        count += 1
        sum   += int(num)

print(sum)
print(count)
```

```
      ---------------------------------------------
      ----------------------------
      ValueError
```

```
import socket

mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect(('data.pr4e.org', 80))
cmd = 'GET http://data.pr4e.org/intro-short.txt HTTP/1.0\r\n\r\n'.encode()
mysock.send(cmd)

while True:
    data = mysock.recv(512)
    if len(data) < 1:
        break
    print(data.decode(),end='')

mysock.close()
```

```
HTTP/1.1 200 OK
Date: Fri, 19 Jun 2020 13:54:53 GMT
Server: Apache/2.4.18 (Ubuntu)
Last-Modified: Sat, 13 May 2017 11:22:22 GMT
ETag: "1d3-54f6609240717"
Accept-Ranges: bytes
Content-Length: 467
Cache-Control: max-age=0, no-cache, no-store, must-revalidate
Pragma: no-cache
Expires: Wed, 11 Jan 1984 05:00:00 GMT
Connection: close
Content-Type: text/plain

Why should you learn to write programs?

Writing programs (or programming) is a very creative
and rewarding activity.  You can write programs for
many reasons, ranging from making your living to solving
a difficult data analysis problem to having fun to helping
someone else solve a problem.  This book assumes that
everyone needs to know how to program, and that once
you know how to program you will figure out what you want
to do with your newfound skills.
```

```
import urllib.request, urllib.parse, urllib.error
from bs4 import BeautifulSoup
import ssl

ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.veify_mode = ssl.CERT_NONE
```

```
      ---------------------------------------------
      ----------------------------
      AttributeError
      Traceback (most recent call last)
      <ipython-input-10-75d78a64c1cf> in <module>()
            5 ctx = ssl.create_default_context()
            6 ctx.check_hostname = False
      ----> 7 ctx.veify_mode = ssl.CERT_NONE

      AttributeError: 'SSLContext' object has no
      attribute 'veify_mode'
```

```
link = "http://py4e-data.dr-chuck.net/comments_603495.html"
html = urllib.request.urlopen(link, context = ctx).read()
soup = BeautifulSoup(html, 'html.parser')


tags = soup('span')
total , count = 0 , 0
for tag in tags:
    count += 1
    total += int(tag.contents[0])
print("The total lines are {} and the total sum is {}".format(count,total))
```

```
  The total lines are 50 and the total sum is 2206
```

```
import urllib.request, urllib.parse, urllib.error
from bs4 import BeautifulSoup
import ssl

ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.veify_mode = ssl.CERT_NONE
```

```
  ---------------------------------------------
      ----------------------------
      AttributeError
      Traceback (most recent call last)
      <ipython-input-13-30f81ddaf41e> in <module>()
            5 ctx = ssl.create_default_context()
            6 ctx.check_hostname = False
      ----> 7 ctx.veify_mode = ssl.CERT_NONE

      AttributeError: 'SSLContext' object has no
      attribute 'veify_mode'
```

SEARCH STACK OVERFLOW

```
link  = input("Entrer URL ::> ")
count = int(input("Enter Count ::> "))
pos   = int(input("Enter Position ::> "))

print("Retriving the link...", link)
```

```
for i in range(0,count):
    html  = urllib.request.urlopen(link).read()
    soup  = BeautifulSoup(html)
    tags  = soup("a")

    links = tags[pos].get("href")

output = tags[pos].contents[0]
print(output)
```

```
Entrer URL ::> http://py4e-data.dr-chuck.net/known_by_Nell.html
Enter Count ::> 7
Enter Position ::> 18
Retriving the link... http://py4e-data.dr-chuck.net/known_by_Nell.html
Abrar
```

```
import urllib
from bs4 import BeautifulSoup

url = input('Enter - ')
count = int(input('Enter count: '))
position = int(input('Enter position: '))
for i in range(count+1):
    html = urllib.request.urlopen(url).read()
    soup = BeautifulSoup(html, 'html.parser')
    # Retrieve all of the anchor tags
    tags = soup.find_all('a')
    print("Retrieving: ", url)
    tag = tags[position-1]
    url = tag.get('href', None)
```

```
Enter - http://py4e-data.dr-chuck.net/known_by_Nell.html
Enter count: 7
Enter position: 18
Retrieving:  http://py4e-data.dr-chuck.net/known_by_Nell.html
Retrieving:  http://py4e-data.dr-chuck.net/known_by_Melis.html
Retrieving:  http://py4e-data.dr-chuck.net/known_by_Zachary.html
Retrieving:  http://py4e-data.dr-chuck.net/known_by_Beatriz.html
Retrieving:  http://py4e-data.dr-chuck.net/known_by_Darien.html
Retrieving:  http://py4e-data.dr-chuck.net/known_by_Asim.html
Retrieving:  http://py4e-data.dr-chuck.net/known_by_Sabriyah.html
Retrieving:  http://py4e-data.dr-chuck.net/known_by_Analyse.html
```

```
import urllib
import json
import xml.etree.ElementTree as ET

url  = input("Enter URL - ")
u    = urllib.request.urlopen(url)
data = u.read()
xml_data = ET.fromstring(data)
search_str = "comments/comment"
count_tags = xml_data.findall(search_str)
```

```
total = 0
for tags in count_tags:
    c = tags.find("count")
    total += int(c.text)

print(total)
```

```
Enter URL - http://py4e-data.dr-chuck.net/comments_603497.xml
2459
```

```
import json
import urllib
url = input("Enter URL - ")
u = urllib.request.urlopen(url)
data = u.read()
data_load = json.loads(data)

total = 0
for tags in data_load["comments"]:
    total += tags["count"]

print(total)
```

```
Enter URL - http://py4e-data.dr-chuck.net/comments_603498.json
2632
```

```
import urllib.request, urllib.parse, urllib.error
import json
import ssl

api_key = False
# If you have a Google Places API key, enter it here
# api_key = 'AIzaSy___IDByT70'
# https://developers.google.com/maps/documentation/geocoding/intro

if api_key is False:
    api_key = 42
    serviceurl = 'http://py4e-data.dr-chuck.net/json?'
else :
    serviceurl = 'https://maps.googleapis.com/maps/api/geocode/json?'

# Ignore SSL certificate errors
ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE

while True:
    address = input('Enter location: ')
    if len(address) < 1: break

    parms = dict()
    parms['address'] = address
    if api_key is not False: parms['key'] = api_key
    url = serviceurl + urllib.parse.urlencode(parms)

    print('Retrieving', url)
```

```
uh = urllib.request.urlopen(url, context=ctx)
data = uh.read().decode()
print('Retrieved', len(data), 'characters')

try:
    js = json.loads(data)
except:
    js = None

if not js or 'status' not in js or js['status'] != 'OK':
    print('==== Failure To Retrieve ====')
    print(data)
    continue

print(json.dumps(js, indent=4))

lat = js['results'][0]['geometry']['location']['lat']
lng = js['results'][0]['geometry']['location']['lng']
print('lat', lat, 'lng', lng)
location = js['results'][0]['formatted_address']
print(location)
```

```
Enter location: University of Sao Paulo
Retrieving http://py4e-data.dr-chuck.net/json?address=University+of+Sao+Paulo&k
Retrieved 1703 characters
{
    "results": [
        {
            "address_components": [
                {
                    "long_name": "S\u00e3o Paulo",
                    "short_name": "S\u00e3o Paulo",
                    "types": [
                        "administrative_area_level_2",
                        "political"
                    ]
                },
                {
                    "long_name": "Butanta",
                    "short_name": "Butanta",
                    "types": [
                        "administrative_area_level_4",
                        "political"
                    ]
                },
                {
                    "long_name": "State of S\u00e3o Paulo",
                    "short_name": "SP",
                    "types": [
                        "administrative_area_level_1",
                        "political"
                    ]
                },
                {
                    "long_name": "Brazil",
                    "short_name": "BR",
                    "types": [
                        "country",
                        "political"
                    ]
                }
            ],
            "formatted_address": "Butanta, S\u00e3o Paulo - State of S\u00e3o P
            "geometry": {
                "location": {
                    "lat": -23.5613991,
                    "lng": -46.7307891
                },
                "location_type": "GEOMETRIC_CENTER",
                "viewport": {
                    "northeast": {
                        "lat": -23.5600501197085,
                        "lng": -46.72944011970849
                    },
                    "southwest": {
                        "lat": -23.5627480802915,
                        "lng": -46.73213808029149
                    }
                }
            }
```