Multithreading, regex, API request assignment.

1. Write a normal python program to calculate squares and cubes of [1,2,3,4,5,6,7]. Then implement the same with multithreading without synchronization and with synchronization. And observe the time difference in all 3 codes. (Note: after each operation of square and cube for a number put sleep of 1 sec for better understanding.)

SOL:

Normal python prog

```
L1= [1,2,3,4,5,6,7]

L2=[]

L3=[]

For I in l1:

        L2.append(i*i)

        L3.append(i*i*i)

Print(L2)

Print(L3)
```

multithreading without synchronization

```
import threading

import time

def print_cube(num):

        print("Cube: {}".format(num * num * num))

        time.sleep(1)

def print_square(num):

        print("Square: {}".format(num * num))

        time.sleep(1)

if __name__ == "__main__":

        l1=[1,2,3,4,5,6,7]

        for i in l1:

            t1 = threading.Thread(target=print_square, args=(i,))

            t2 = threading.Thread(target=print_cube, args=(i,))
        # starting thread 1
          t1.start()
        # starting thread 2
          t2.start()

        # wait until thread 1 is completely executed
          t1.join()
        # wait until thread 2 is completely executed
          t2.join()

        print("yup.threads completely executed")
```

multithreading with synchronization

```
import threading
import time

def increment():

    l1=[1,2,3,4,5,6,7]
    for num in l1:
        print("Cube: {}".format(num * num * num))
        print("Square: {}".format(num * num))

def thread_task():

        for _ in range(2):
                increment()

def main_task():
        # creating threads
        t1 = threading.Thread(target=thread_task)
        t2 = threading.Thread(target=thread_task)

        # start threads
        t1.start()
        t2.start()

        # wait until threads finish their job
        t1.join()
        t2.join()

if __name__ == "__main__":
        l1=[1,2,3,4,5,6,7]
        for i in l1:
                main_task()
                time.sleep(1)
                 print("Cube: {}".format(num * num * num))
                  print("Square: {}".format(num * num))
```

2. Write a Python program to remove the parenthesis area in a string. Sample data: ["example (.in)", "w3resource", "github (.com)", "stackoverflow (.us1)"] Expected Output: example w3resource github stackoverflow

SOL:

```
import re
data =  ["example (.in)", "w3resource", "github (.com)", "stackoverflow (.us1)"]
for i in data:
   modified_string = re.sub(r"\([^()]*\)", "", i)
   print(modified_string)
```

3. Write a python program to hit GET api: https://restcountries.com/v3.1/alpha/pe and print:

a. All 3 languages with full names.

b. Capital, area, population.

SOL:

```
import requests

response = requests.get("https://restcountries.com/v3.1/alpha/pe ")

URL = "https://restcountries.com/v3.1/alpha/pe"

PARAMS = {'languages':values}

r = requests.get(url = URL, params = PARAMS)

print(r)

data = r.json()

capital = data[capital]

area = data[area]

population = data[population]

print(capital,area,population)
```