**EXPERIEMNT NO - 09**

Title: Program to demonstrate “multi-Threading” in python.

1. Creating Threads in python.
2. Without creating a class.
3. By extending Thread class.
4. Without extending thread class.
   1. Without creating a class.

Code:

from threading import \*

def new():

    for x in range(7):

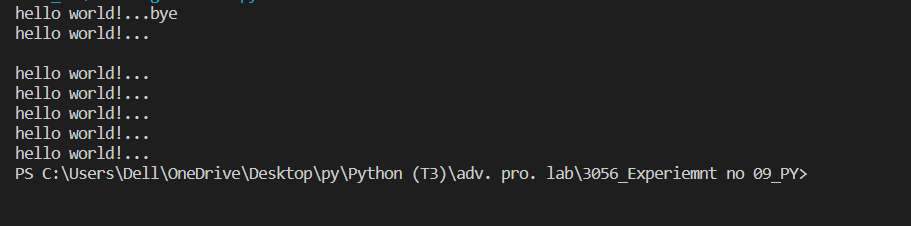
        print("hello world!...")

t1 = Thread(target=new)

t1.start()

print("bye")

output:



Code:

from threading import \*

def new():

    for x in range(7):

        print("hello world!...")

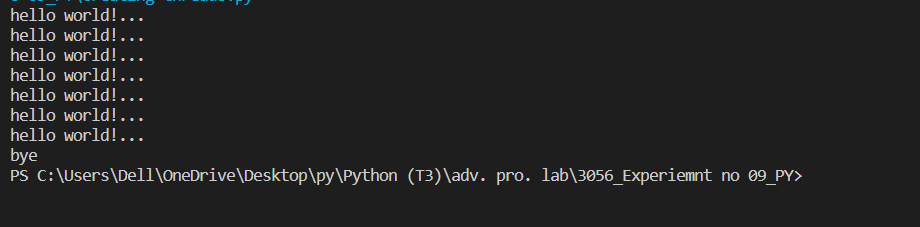
t1 = Thread(target=new)

t1.start()

t1.join()

print("bye")

Output:



Code :

from threading import \*

def new():

    for x in range(6):

        print("hello world...",current\_thread().getName())

t1=Thread(target=new)

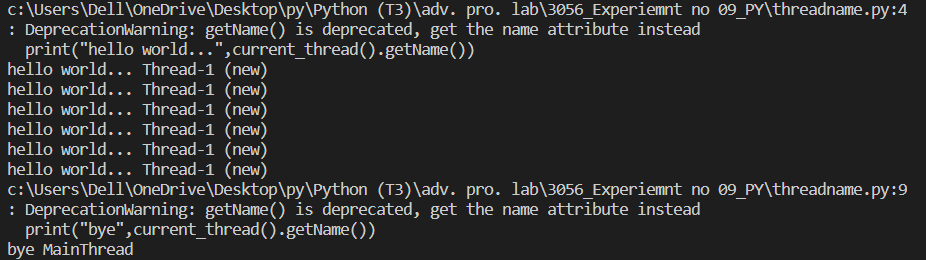
print(current\_thread().getName())

t1.start()

t1.join()

print("bye",current\_thread().getName())

Output:



* 1. By Extending thread class.

Code:

from threading import \*

class A(Thread):

    def run(self):

        for x in range(7):

            print("child =",current\_thread().getName())

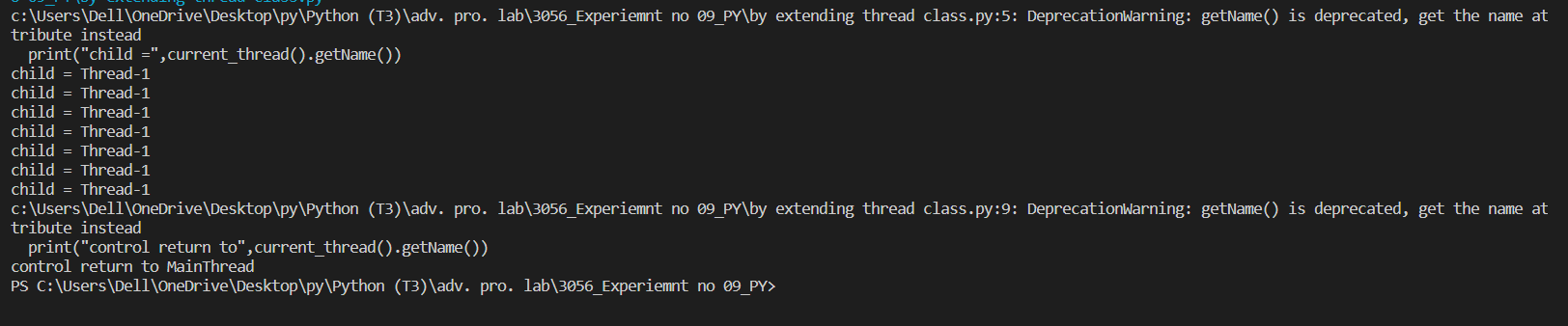
obj=A()

obj.start()

obj.join()

print("control return to",current\_thread().getName())

Output:



* 1. Without extending thread class.

Code:

from threading import \*

class ex:

    def B(self):

        Lst=[2, 1, 'w', 8.7, 'abc']

        for x in Lst:

            print("child printing",x)

myobj = ex()

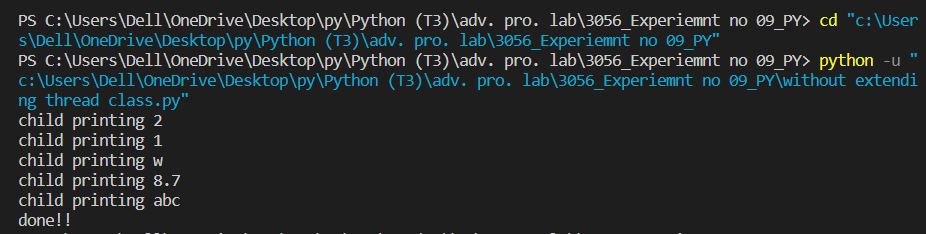
t1= Thread(target=myobj.B)

t1.start()

t1.join()

print("done!!")

Output:



**#Time comparisons:**

***Without multithreading……***

Code:

import time

def d2(n):

    for x in n:

        time.sleep(1)

        print(x%2)

def d3(n):

    for x in n:

        time.sleep(1)

        print(x%3)

n=[2, 4, 3, 6, 7]

s=time.time()

d2(n)

d3(n)

e=time.time()

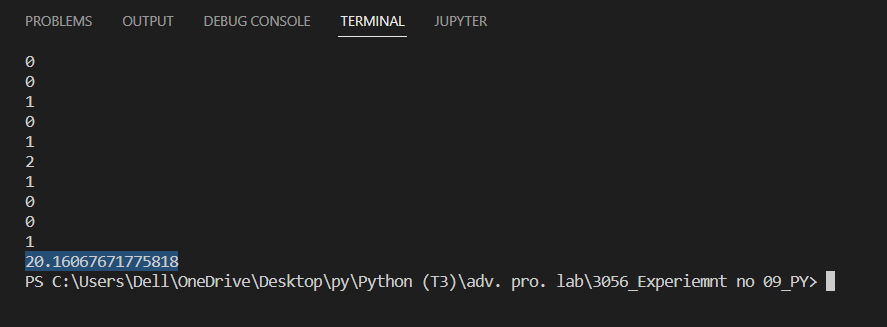
d2(n)

d3(n)

e=time.time()

print(e-s)

output:



Now ***with multithreading……***

Code:

from threading import \*

print("\n")

def d(n):

    for x in n:

        time.sleep(1)

        print(x%2)

def d1(n):

    for x in n:

        time.sleep(1)

        print(x%3)

n=[2, 4, 3, 6, 7]

s=time.time()

t1 = Thread(target=d, args=(n,))

t2 = Thread(target=d1, args=(n,))

t1.start()

t2.start()

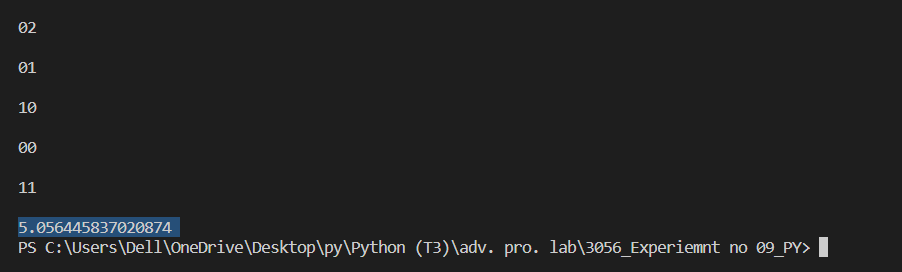
t1.join()

t2.join()

e=time.time()

print(e-s)

Output:



*Multithreading gives enhanced performance with decrease in completion time.*