



Universidad Nacional Autónoma de
México

Facultad de Ingeniería

División de Ingeniería Mecánica e Industrial

Laboratorio de Cómputo de Ingeniería Mecatrónica (1964)



Profesor: Miguel Serrano Reyes

Semestre 2021-2

Práctica No. 1

Nombre de la práctica

Programación Concurrente

Nombre del Estudiante:

Dimas Ramírez Luis Daniel

Actividad 1

Comentar a detalle todas las líneas del ejercicio
“practica_01_concurrencia_sincrono.py”

```
F:\Cienia de datos (Python)\Spyder\practica_01_concurrencia_sincrono.py
practica_01_concurrencia_sincrono.py x practica_01_concurrencia_hilos.py x practica_02_API's.py x Entrega_practica_01.py x practica_0

1 #####
2 ##### ACTIVIDAD 1 #####
3 #####
4 #####
5 #####
6 #####
7
8 import requests
9 import time
10
11 def descargar_sitio(url, sesion): #Primero se define la función para descargar el sitio
12     with sesion.get(url) as respuesta: #Con este comando obtenemos la url del sitio
13         print("Leyendo {} de {}".format(len(respuesta.content), url))
14         #Nos muestra la longitud del contenido de la url y la url.
15
16 def descargar_todo(sitios): #Esta función va a contener los sitios
17     with requests.Session() as sesion:
18         for url in sitios: #Este ciclo descarga la url de los que este contenido en
19             descargar_sitio(url, sesion) #sitios
20
21 if __name__ == '__main__': #Indica al script "el método main" que se ejecutara primero
22     sitios=[ "https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html",
23             "https://imagenenlaciencia.blogspot.com/2014/08/chamacos-mendigos.html"]*100
24     #En las líneas de arriba, hacemos una lista con dos sitios web, el cual está
25     #multiplicado por 100
26     tiempo_inicio= time.time() #Con la biblioteca time obtenemos el tiempo de inicio
27     descargar_todo(sitios) #descargamos todos los sitios contenidos en la lista de arriba
28     duracion = time.time()-tiempo_inicio #con esta resta obtenemos el tiempo que se
29         #tardo en realizar la descarga
30     print('Se descargaron {} sitios en {} segundos'.format(len(sitios), duracion))
31     #finalmente este nos muestra cuantos sitios se descargaron y el tiempo que se requirio
32     #####
33     #####
34     #####
35
36 #####
```

Name	Type	Size	Value
duracion	float	1	25.3706693649292
sitios	list	200	['https://matplotlib.org/stabl...
tiempo_inicio	float	1	1628692701.4163544

Variable explorer Help Plots Files

Console 1/A x

```
practica_01_concurrencia_sincrono.py
Leyendo 110192 de https://
imagenenlaciencia.blogspot.com/2014/08/chamacos-
mendigos.html
Leyendo 74668 de https://matplotlib.org/stable/
api/_as_gen/matplotlib.pyplot.plot.html
Leyendo 110192 de https://
imagenenlaciencia.blogspot.com/2014/08/chamacos-
mendigos.html
Leyendo 74668 de https://matplotlib.org/stable/
api/_as_gen/matplotlib.pyplot.plot.html
Leyendo 110192 de https://
imagenenlaciencia.blogspot.com/2014/08/chamacos-
mendigos.html
Se descargaron 200 sitios en 25.3706693649292
segundos

In [4]:
```

Actividad 2

Comentar a detalle todas las líneas del ejercicio
“practica_01_concurrencia_hilos.py”

```
F:\Cienia de datos (Python)\Spyder\practica_01_concurrencia_hilos.py
practica_01_concurrencia_sincrono.py x practica_01_concurrencia_hilos.py x practica_02_API's.py x Entrega_practica_01.py x practica_0

5 import requests
6 import threading #A diferencia de la otra actividad, aqui necesitamos la biblioteca
7 import concurrent.futures # threading y concurrent.futures
8
9 # En esta actividad utilizaremos subprocesos para poder hacer más eficientes
10 #las descargas , la eficiencia se proporcional al número de núcleos de cada equipo
11 thread_local= threading.local()
12
13 def get_session(): #definimos la función para obtener la sesión
14     if not hasattr(thread_local, "session"): #aquí definimos que si no tiene el atributo
15         thread_local.session = requests.Session() # sesion, le creamos una sesión
16     return thread_local.session# y aquí le asigna la sesión
17
18 def descargar_sitio(url):#Se define la función para descargar el sitio
19     session = get_session() #Con este comando obtenemos la sesión del sitio
20     with session.get(url) as respuesta: #Con este comando obtenemos la url del sitio
21         print("Leyendo {} de {}".format(len(respuesta.content), url))
22         #Nos muestra la longitud del contenido de la url y la url.
23
24 def descargar_todo(sitios): #creamos función para descargar los sitios pero con hilos
25     with concurrent.futures.ThreadPoolExecutor(max_workers=5) as ejecutor:
26         #en la linea de arriba definimos 5 subprocesos queremos para hacer más eficiente
27         ejecutor.map(descargar_sitio, sitios) #la descarga y aquí ejecuta la descarga de
28         #la url tantas veces como sitios haya
29
30 if __name__ == '__main__': #Indica al script "el método main" que se ejecutara primero
31     sitios=[ "https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html",
32             "https://imagenenlaciencia.blogspot.com/2014/08/chamacos-mendigos.html"]*100
33
34     #En las líneas de arriba, hacemos una lista con dos sitios web, el cual está
35     #multiplicado por 100
36     tiempo_inicio= time.time()#Con la biblioteca time obtenemos el tiempo de inicio
37     descargar_todo(sitios)#descargamos todos los sitios contenidos en la lista de arriba
38     duracion_hilos = time.time()-tiempo_inicio#con esta resta obtenemos el tiempo que se
39         #tardo en realizar la descarga
40     print('Se descargaron {} sitios en {} segundos'.format(len(sitios), duracion_hilos))
41
42 #####
```

Name	Type	Size	Value
duracion_hilos	float	1	5.671797037124634
sitios	list	200	['https://matplotlib.org/sta...
thread_local	_local	1	_local object of _thread
tiempo_inicio	float	1	1628693093.086764

Variable explorer Help Plots Files

Console 2/A x

```
practica_01_concurrencia_hilos.py
Leyendo 110192 de https://
imagenenlaciencia.blogspot.com/2014/08/chamacos-
mendigos.html
Leyendo 74668 de https://matplotlib.org/stable/
api/_as_gen/matplotlib.pyplot.plot.html
Leyendo 74668 de https://matplotlib.org/stable/
api/_as_gen/matplotlib.pyplot.plot.html
Leyendo 110192 de https://
imagenenlaciencia.blogspot.com/2014/08/chamacos-
mendigos.html
Leyendo 110192 de https://
imagenenlaciencia.blogspot.com/2014/08/chamacos-
mendigos.html
Se descargaron 200 sitios en 5.671797037124634
segundos

In [2]:
```

IPython console History

Actividad 3

Ejecutar 10 veces el primer código y guardas los tiempos en una lista.
Ejecutar 10 veces el segundo código, guarda los tiempos en otra lista.
Comparar ambos tiempos con un boxplot.

```
Fi\Ciencia de datos (Python)\Spyder\Entrega_practica_01.py
practica_01_concurrencia_sincrono.py practica_01_concurrencia_hilos.py Entrega_practica_01.py practica_02_API's.py practica_05_s...

35 print("Leyendo {} de {}".format(len(respuesta2.content), url))
36
37 def descargar_todo2(sitios):
38     with concurrent.futures.ThreadPoolExecutor(max_workers=5) as ejecutor:
39         ejecutor.map(descargar_sitio2, sitios)
40
41 #####
42 ##### ACTIVIDAD 3 #####
43 #####
44
45 duracion_sincrono = []
46 for i in range(10):
47     tiempo_inicio1= time.time()
48     descargar_todo2(sitios)
49     duracion_sincrono.append(time.time()-tiempo_inicio1)
50
51 duracion_hilos = []
52 for m in range(10):
53     tiempo_inicio2= time.time()
54     descargar_todo2(sitios)
55     duracion_hilos.append(time.time()-tiempo_inicio2)
56
57 print(duracion_hilos)
58 print(duracion_sincrono)
59
60 tiempos = [duracion_sincrono, duracion_hilos]
61 plt.boxplot(tiempos, patch_artist= True)
62 plt.xticks([1,2], ["Concurencia sincrona", "Concurencia hilos"])
63 plt.title("Comparación entre concurrencias")
64 plt.ylabel("Tiempo de concurrencia")
65 plt.show()
66 #####
67 #####
68 #####
69
```

Built-in mutable sequence.

If no argument is given, the constructor creates a new empty list. The argument must be an iterable if specified.

Variable explorer Help Plots Files

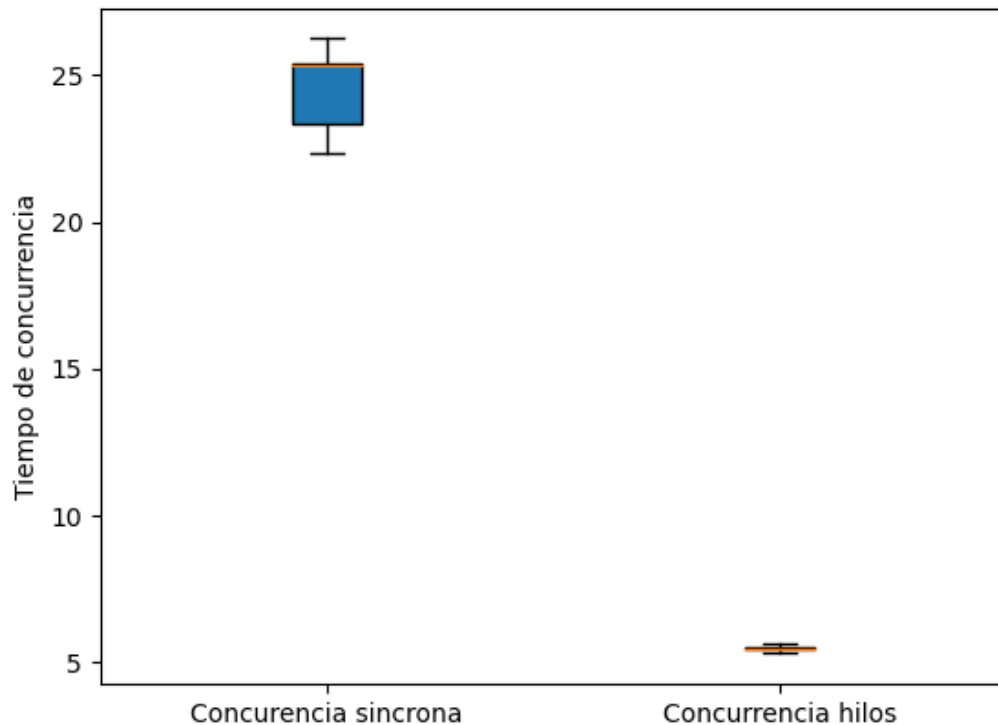
Console 4/A

```
... plt.title("Comparación entre concurrencias")
... plt.ylabel("Tiempo de concurrencia")
... plt.show()

In [23]: tiempos = [duracion_sincrono, duracion_hilos]
... plt.boxplot(tiempos, patch_artist= True)
... plt.xticks([1,2], ["Concurencia sincrona", "Concurencia hilos"])
... plt.title("Comparación entre concurrencias")
... plt.ylabel("Tiempo de concurrencia")
... plt.show()

In [24]:
```

Comparación entre concurrencias



Evidentemente la concurrencia con hilos es mucho mejor. Tiene menor dispersión de datos y además el tiempo en el que se ejecuta es casi cinco veces menor. La media de tiempo para la concurrencia con hilos es de 5.47 segundos mientras que para la concurrencia síncrona es de 24.59