

1IEE14 - Laboratorio 8

Instrucciones para el laboratorio:

- Materiales permitidos: Wiki del curso, apuntes de clase, consultar foros, tutoriales o documentación de python online.
- Está prohibido el uso de cualquier modelo de lenguaje como ChatGPT o alguno similar. A cualquier alumno que se le detecte que ha consultado un modelo de lenguaje se le pondrá nota 0(cero) en el laboratorio.
- Usted debe subir a Paideia 1 solo archivo comprimido(.zip o .rar) con el nombre L9_CODIGOPUCP.zip o L9_CODIGOPUCP.rar. Este archivo comprimido debe tener archivos de python para cada pregunta. Por ejemplo:
Para la pregunta 1, se espera encontrar archivos con nombre pregunta1_cliente.py, pregunta1_servidor.py. Lo mismo para las pregunta 2 y 3. Note que se le está pidiendo archivos de python para cada pregunta, no se aceptarán soluciones en jupyter notebook.
- Si al momento de ejecutar cualquiera de sus programas le sale el error **Address already in use**, eso se debe a que el puerto que usaba sigue abierto y no está disponible para su uso. Cuando ocurra esto, puede cambiar de puerto, no es obligatorio usar solamente el puerto 5000.
- El horario máximo permitido para subir el archivo es a las 10:00:00 pm. Pasada esa hora, habrá una penalidad de 2 puntos por cada minuto extra que se demore en entregar su archivo.

1. (6 pts) Desarrolle un sistema cliente-servidor empleando Sockets donde el cliente monitoree el estado del servidor como se muestra a continuación.

```
PS D:\PUCP\2023-1\Arqui\Lab09> python .\p2_serv.py
Esperando conexiones ...
Conexión entrante de ('127.0.0.1', 51982)
Conexión entrante de ('127.0.0.1', 51985)
Conexión entrante de ('127.0.0.1', 51986)
Conexión entrante de ('127.0.0.1', 51987)
Cerrando el servidor ...
PS D:\PUCP\2023-1\Arqui\Lab09> python .\p2_serv.py
Esperando conexiones ...
Conexión entrante de ('127.0.0.1', 52001)
Conexión entrante de ('127.0.0.1', 52002)
Conexión entrante de ('127.0.0.1', 52008)
Conexión entrante de ('127.0.0.1', 52009)
Conexión entrante de ('127.0.0.1', 52010)
Conexión entrante de ('127.0.0.1', 52011)
Cerrando el servidor ...
PS D:\PUCP\2023-1\Arqui\Lab09>

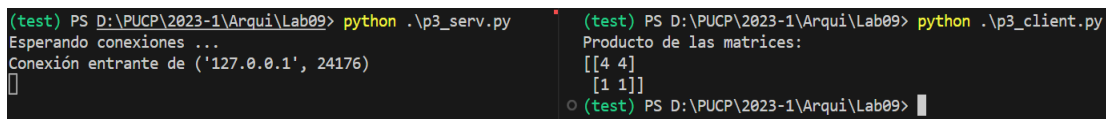
PS D:\PUCP\2023-1\Arqui\Lab09> python .\p2_client.py
[-] El servidor no responde
[*] El servidor está operativo
[*] El servidor está operativo
[*] El servidor está operativo
[*] El servidor está operativo
[*] El servidor está operativo
[-] El servidor no responde
[-] El servidor no responde
[*] El servidor está operativo
[*] El servidor está operativo
[*] El servidor está operativo
[*] El servidor está operativo
[*] El servidor está operativo
[*] El servidor está operativo
[-] El servidor no responde
```

Básicamente el cliente se conecta al servidor y comprueba de manera periódica si este se encuentra en funcionamiento. Los mensajes informativos deben ser implementados como se muestran en el gráfico.

- a) (1.5 pts) Implementar el servidor.
- b) (4 pts) Implementar el cliente. Debe comprobar el estado del servidor cada 5 segundos.

- c) (0.5 pts) Personalizar los mensajes informativos que proporciona el cliente. Si el servidor está operativo, que muestre los mensajes en verde y si no responde que los muestre en rojo.

2. (7.5 pts) Implemente un sistema cliente-servidor que calcule la multiplicación de dos matrices. El cliente debe generar dos matrices cuadradas de 2x2 del mismo tamaño y enviarlas al servidor. Por otro lado, el servidor debe recibir las matrices enviadas por el cliente, realizar la multiplicación y enviar el resultado al cliente. Finalmente, el cliente debe recibir el resultado e imprimirlo.



```
(test) PS D:\PUCP\2023-1\Arqui\Lab09> python .\p3_serv.py
Esperando conexiones ...
Conexión entrante de ('127.0.0.1', 24176)
█

(test) PS D:\PUCP\2023-1\Arqui\Lab09> python .\p3_client.py
Producto de las matrices:
[[4 4]
 [1 1]]
○ (test) PS D:\PUCP\2023-1\Arqui\Lab09> █
```

Nota: Se recomienda usar el módulo Pickles para el envío de matrices en la conexión cliente-servidor.

- a) (4.5 pts) Implementar el servidor.
b) (3 pts) Implementar el cliente.

3. (6.5 pts) Implemente un sistema de chat simple, para eso escriba 2 archivos de Python: pregunta3_cliente.py y pregunta3_servidor.py

El funcionamiento de la aplicación de chat debe ser como se describe a continuación:

- El cliente y el servidor se comunicarán a través del puerto 5000 usando sockets.
- Cuando los programas están conectados ,va a funcionar como un chat simple: La persona que está en el servidor escribe algo y el programa cliente lo recibe y lo muestra en pantalla, luego la persona que está en el cliente responde algo tipeando texto el cual es recibido por el servidor, y la conversación continúa de esa manera.
- Para simplificar la solución del problema, la comunicación será siempre por turnos: En el primer turno, el servidor tiene que decir algo y el cliente está esperando recibir turno; en el segundo turno, ahora le toca al cliente decir algo y el servidor está a la espera de recibir texto; en el tercer turno le toca nuevamente al servidor, y así sucesivamente. Cada programa imprime el caracter '>' para indicar que es su turno y que está a la espera de recibir texto del usuario.
- La conversación terminará cuando cualquiera de los dos usuarios en su turno mande la palabra salir, momento en el cual las conexiones deben cerrarse.

La siguiente captura de pantalla muestra un ejemplo del sistema de chat funcionando:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL REPL
○ (venv) PS D:\Ciclo 2023-1\lab9_telecom_solus> python chat_cliente.py
Connectado a servidor.
Recibí: Hola cliente, cómo estás?
> Yo bien, confirmame de que recibes mis mensajes por favor.
Recibí: Confirmado, si te leo!
● > Perfecto, gracias
  
```

```
(venv) PS D:\Ciclo 2023-1\lab9_telecom_solus> python chat_servidor.py
Server started and listening on port 5000
Connectado a cliente: ('127.0.0.1', 51360)
> Hola cliente, cómo estás?
Recibido: Yo bien, confirmame de que recibes mis mensajes por favor.
> Confirmado, si te leo!
Recibido: Perfecto, gracias
>  
```

Figura que muestra cómo debe funcionar la aplicación del chat. Para hacer el problema más simple, la conversación es por turnos.