ACTIVIDAD EVALUATIVA EJE 3

Laura Juliana Ramírez Barrera

Fundación Universitaria del Área Andina

Ingeniería De Sistemas – Virtual

[Modelos De Programación II - IS - 202460-6a - 61](https://areandina.instructure.com/courses/51454)

Deivys Morales

15 de agosto de 2024

INTRODUCCIÓN

Durante el desarrollo de diversas aplicaciones y sistemas de red, los hilos y sockets juegan un papel fundamental para garantizar un gran rendimiento y la capacidad de comunicación entre diferentes procesos. Mientras los hilos permiten la ejecución de múltiples tareas al tiempo los sockets establecen la comunicación entre distintos dispositivos en una red, logrando facilitar el intercambio de datos entre un cliente y un servidor. En este trabajo lograremos ver como funcionan mediante lenguaje de programación Python.

OBJETIVOS

El objetivo principal de este trabajo es entender cómo se implementan los hilos y sockets en lenguaje de programación Python mediante la construcción de un juego, entre el cliente y el servidor donde el cliente enviara los datos al servidor mediante la conexión de un socket.

TAREA

Construir una aplicación en python que permite aplicar el tema de hilos y socket, la aplicación consiste en:

• Construir un juego en el que se generan números aleatorios a través de hilos por parte del cliente.

• Establecer una comunicación entre el cliente y el servidor para que el servidor adivine los números generados por el cliente.

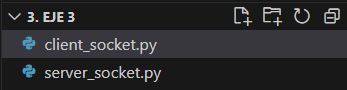
• Contabilizar los aciertos y desaciertos cuando finalice la aplicación.

• Cuando el servidor lleve 3 desaciertos seguidos debe salir un mensaje de “Perdiste”.

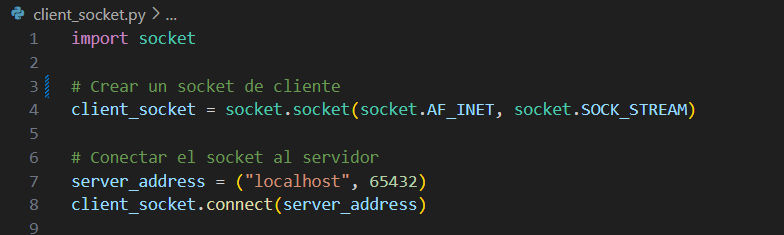
• Terminar la aplicación cuando el cliente envíe la palabra “terminar”

SOLUCIÓN

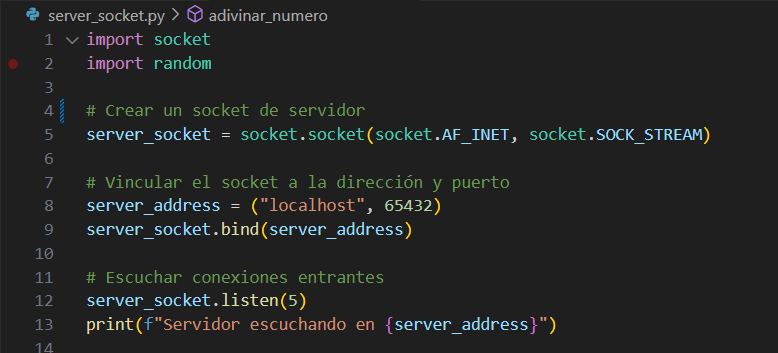
Primero creamos dos archivos, client\_socket.py quien enviará los mensajes y server\_socket.py quien recibirá los mensajes.



Dentro de client\_socket.py comenzamos creando el socket del cliente, para posteriormente establecer conexión con el servidor.



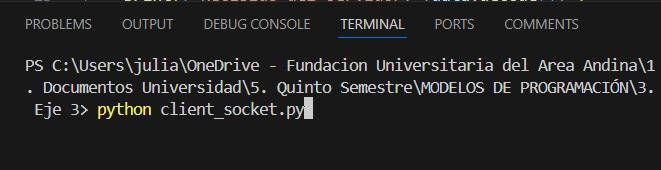
Esto mismo lo hacemos en el archivo de server\_socket.py que al definir el metodo .listen(5) hara que acepte las conexiones entrantes con un maximo de 5 conexiones.



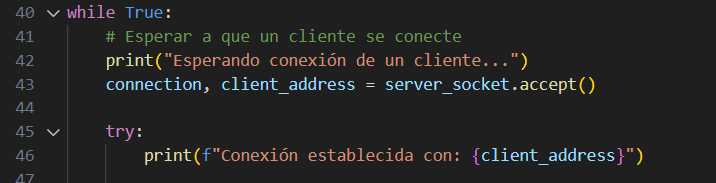
Al ejecutar efectivamente el servidor inicia, esperando la conexión por parte del cliente.

Texto

Descripción generada automáticamente



Se establece la conexión con el servidor.



Texto

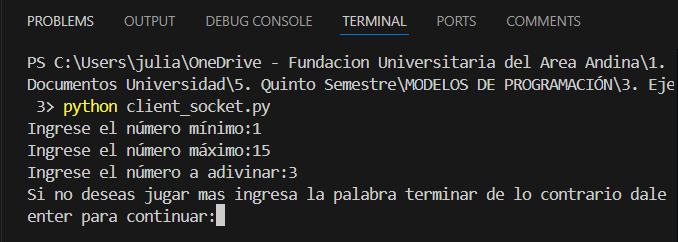
Descripción generada automáticamente

Comenzamos definiendo las variables en client\_socket.py, en este caso como el juego lo que hará será adivinar el numero que el usuario ingrese. Para eso le pediremos que ingrese un número mínimo, un número máximo y el número que desea que el servidor adivine.

Captura de pantalla de un celular

Descripción generada automáticamente

Así mismo le preguntamos si desea continuar jugando, si no lo desea debe ingresar la palabra terminar.

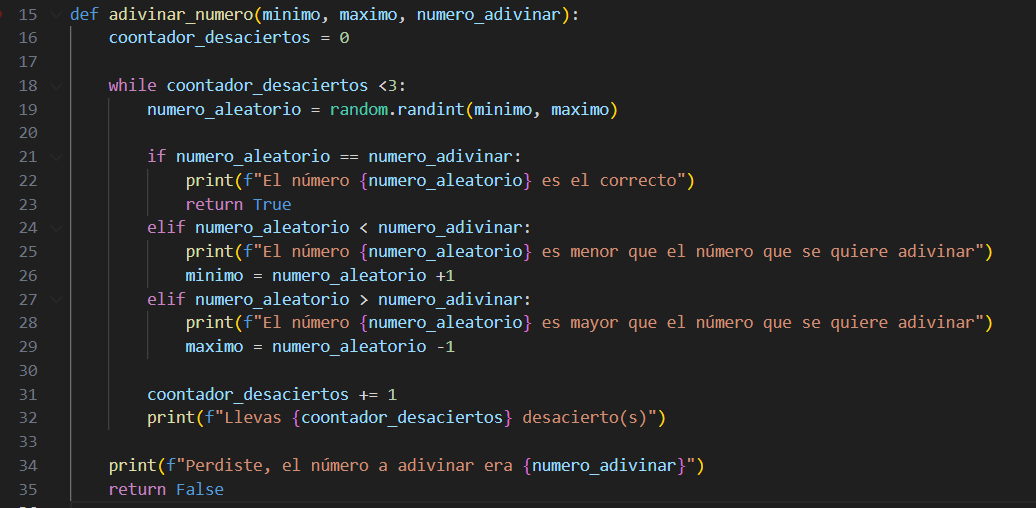


Los datos que ingrese el cliente pasaran al servidor para que inicie el juego

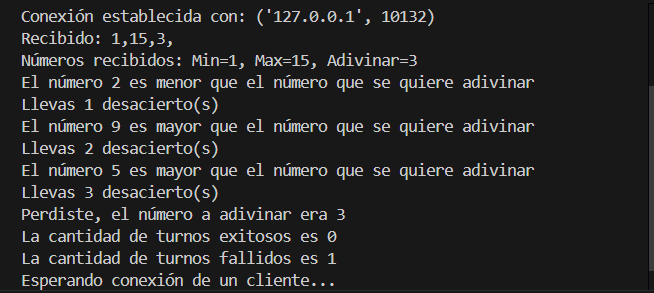
Texto

Descripción generada automáticamente

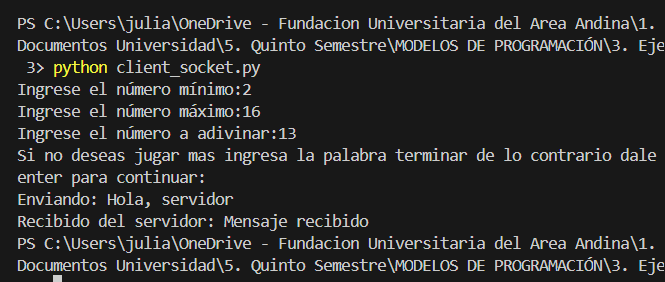
Dentro de server\_socket.py definimos la función adivinar número y por medio del condicional if empezara verificando si el numero aleatorio es igual al numero a adivinar si es menor o si es mayor. De igual forma al colocar el contador de desaciertos contara cuantos desaciertos lleva el servidor y mediante el bucle while, solo le estamos dando 3 oportunidades para que adivine el número.



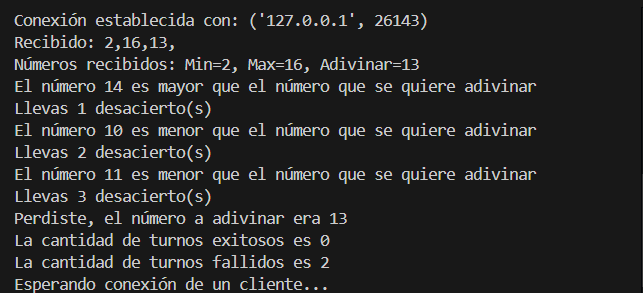
En el primer “turno” del servidor podemos observar que no logro adivinar el número, por ende, nos imprime el número que debió adivinar y la cantidad de turnos fallidos que lleva.



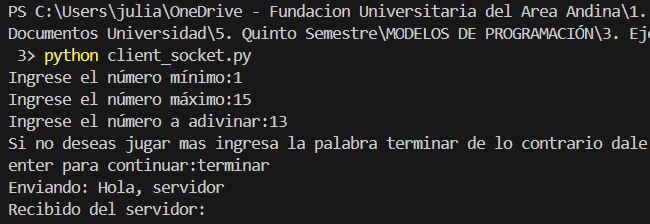
Lo intentamos nuevamente sin terminar aun el juego



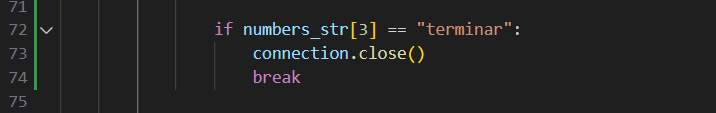
Y el servidor sigue sin tener éxito en adivinar el número, al no terminar el juego nos contabiliza la cantidad de turnos fallidos que lleva el servidor



Continuamos jugando hasta que le damos la orden de terminar el juego.



Al decidir terminar el juego, el socket se desconecta perdiendo así la conexión entre el cliente y el servidor.



Texto

Descripción generada automáticamente

CONCLUSIONES

Mediante el desarrollo de este eje, me di cuenta de que el uso de hilos y sockets nos permiten desarrollar y mejorar significativamente la capacidad y el rendimiento de las aplicaciones, ya que podemos gestionar múltiples conexiones o procesos en paralelo sin bloquear el flujo principal del programa. A través de este eje comprendí que la integración de hilos y socket nos permite la creación no solo de juegos sino también de servidores web, algo fundamental hoy en día. Empresas como Uber utilizan sockets para la comunicación en tiempo real entre los conductores y los usuarios, en las referencias de este trabajo dejare un pequeño instructivo creado por esta compañía donde mencionan como realizan su proceso. Zoom es otra aplicación que utiliza sockets para hacer videollamada, siendo el socket la videollamada y el cliente los conferencistas que se conectaran a la videollamada, de igual forma la cantidad de usuarios dependen del plan adquirido con zoom esto también indica que utilizan sockets con un tope máximo de usuarios.

REFERENCIAS

Python. *Programación con Sockets.*

[https://docs.python.org/es/3/howto/sockets.html](https://docs.python.org/es/3/howto/sockets.html%20)  septiembre 2024

W3Schools. *Random Module*

<https://www.w3schools.com/python/module_random.asp>

Uber. *Uso de la app*

<https://www.uber.com/co/es/about/how-does-uber-work/>

Zoom. Reuniones virtuales

<https://www.zoom.com/es/products/virtual-meetings/>

Repositorio, Juliana Ramirez. [EJE-3-HILOS-Y-SOCKE](https://github.com/JulianaRamirez/EJE-3-HILOS-Y-SOCKET)T

https://github.com/JulianaRamirez/EJE-3-HILOS-Y-SOCKET