

Universidad Nacional de Lanús

DESARROLLO DE SOFTWARE EN SISTEMAS DISTRIBUIDOS

**Actividad Sockets**

Alumno: Mathov Camila

DNI: 40946043

**Estrategia de resolución**

La estrategia de resolución fue crear tanto un cliente como un servidor que se conectarán a una IP (127.0.0.1) y puerto (8080) fijos. Ambos utilizan la librería *winsock2* (o la librería *sockets* en el caso del cliente en Python) para mantener una conexión TCP con sockets bloqueantes.

**Funcionamiento general del cliente:**

Debido a que la lógica del programa se encuentra del lado del servidor tanto el cliente en C como el escrito en Python son realmente simples, estos se conectan al servidor y entran en un bucle donde mostraran el mensaje que recibieron del servidor, esperaran por un input del usuario y luego mandaran dicho input al servidor una y otra vez hasta que ocurra un error al recibir o enviar un mensaje (ya sea debido a que el servidor desconecto al cliente o que el servidor ya no está en funcionamiento).

**Funcionamiento general del servidor:**

Para no afectar la escalabilidad del sistema y permitir agregar más opciones de generación sin tener que modificar los clientes es el servidor el que se encarga de toda la lógica del programa. Es por ello que el servidor es el que le envía el menú a los clientes en el momento en que estos se conectan. El mismo luego de enviar el menu espera un mensaje del cliente con la opción que este eligió y convierte dicho mensaje a una opción numérica utilizando la función *strtol()*, en el caso de que el cliente envié un mensaje con letras en lugar de números esta función se encarga de devolver 0.

**En caso de que la opción sea 1 generar un usuario:**

El servidor mandara un mensaje al cliente pidiéndole que ingrese la cantidad de caracteres que quiere utilizar e indicando cuál es el mínimo y máximo de caracteres con el cual trabaja el sistema. Nuevamente utilizando la función *strtol()* se convertirá el mensaje recibido del cliente a el número de caracteres a utilizar, si este número no es válido el servidor volverá a pedir la cantidad de caracteres e indicar los valores esperados hasta que el usuario le envié un mensaje con un input válido. Una vez la cantidad de caracteres enviada por el usuario este dentro de los límites esperados el servidor llamará a la función *generarUsuario()* para crear un nombre de usuario con dicha cantidad de caracteres y se lo enviara al cliente, luego de esto esperara por un input del cliente para continuar y le enviará nuevamente el menú principal.

**En caso de que la opción sea 2 generar una contraseña:**

La lógica es la misma que para generar un usuario (bucle que le pide un input al cliente hasta que este le envié uno valido) lo único que cambia es el rango de caracteres que son válidos y que la función a la que llama para crear la contraseña es *generarContraseña().*

**En caso de que la opción sea 3 salir:**

El servidor saldrá de la función que está utilizando para manejar al cliente, cerrará la conexión con el mismo y esperará por un nuevo cliente.

**En caso de que la opción sea invalida (cualquier mensaje que no sea 1, 2 o 3):**

El servidor enviara el menú nuevamente.

**Funcionamiento de *generarUsuario()***

Está función posee 2 listas, una con vocales y otra con consonantes. Lo primero que hará será guardar aleatoriamente en la variable "random" un 0 o 1 para seleccionar si la primera letra será una vocal o una consonante, luego de esto se iniciara un for donde el índice empezará en 0 y continuara ejecutándose mientras que el mismo sea menor a la longitud enviada por parámetro. Dentro de este for se preguntará si la suma entre el número de índice actual y la variable "random" que fijamos anteriormente es un número par o impar, en caso de que sea par se agregara a la cadena un elemento al azar de la lista vocales, si es impar se agregara un elemento al azar de la lista de consonantes.

**Funcionamiento de *generarContraseña()***

Está función posee una sola lista que contiene las letras del abecedario de la A a la Z tanto en minúscula como en mayúscula y también los números del 0 al 9. Nuevamente se utiliza un for para asignar uno a uno los caracteres de la cadena que guarda la contraseña agregando aleatoriamente uno de los elementos de la lista de caracteres que se menciona anteriormente.

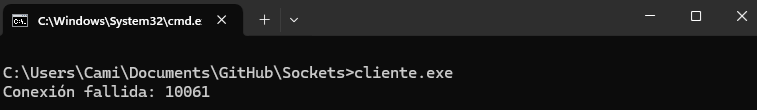
**Repositorio de Github**

<https://github.com/cmathov/Sockets>

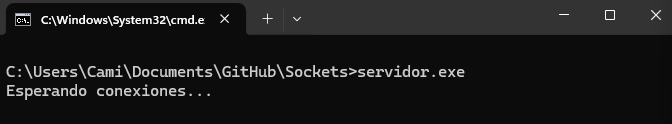
**Pruebas realizadas:**

Pruebas del Servidor en C + Cliente en C

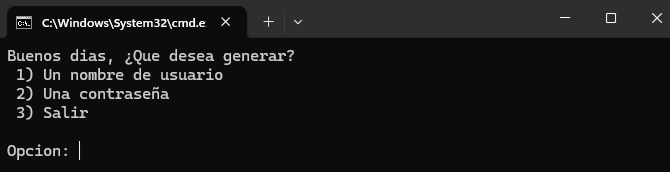
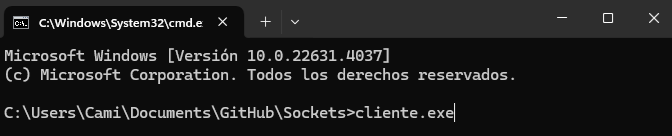
**Iniciando el Cliente sin que el Servidor este iniciado:**



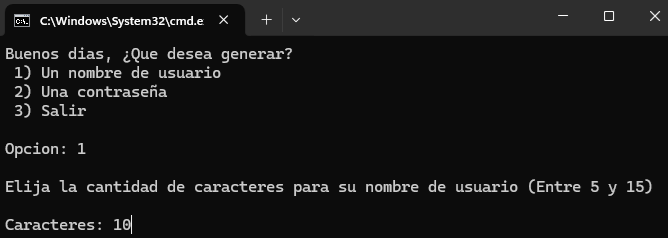
**Iniciando Servidor:**



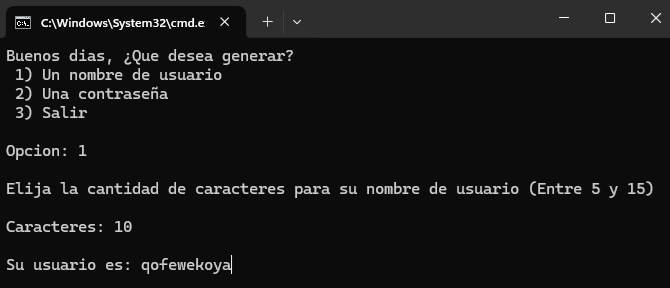
**Iniciando Cliente:**



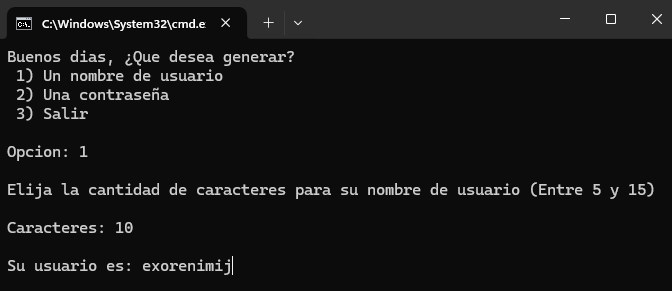
**Seleccionando la opción de generar un nombre de usuario e introduciendo una cantidad correcta de caracteres:**



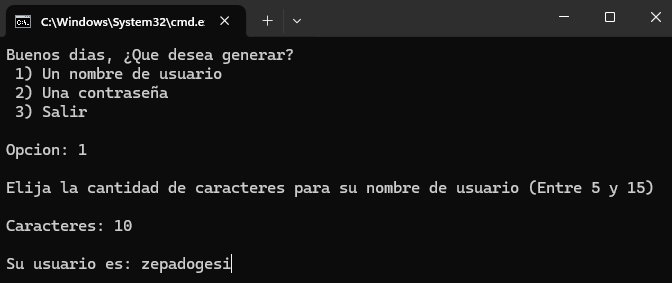
Resultado 1:



Resultado 2:



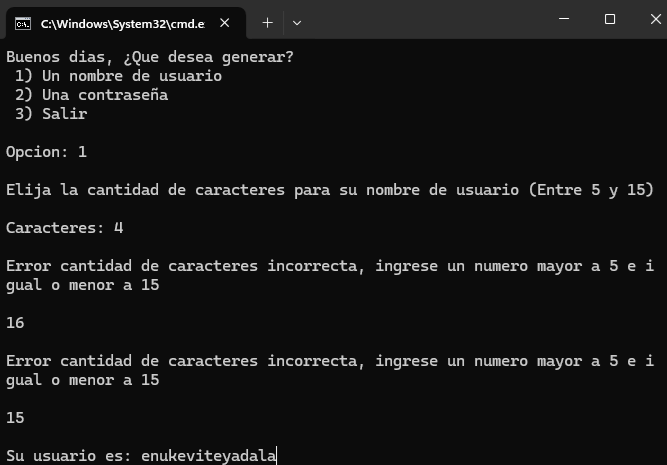
Resultado 3:



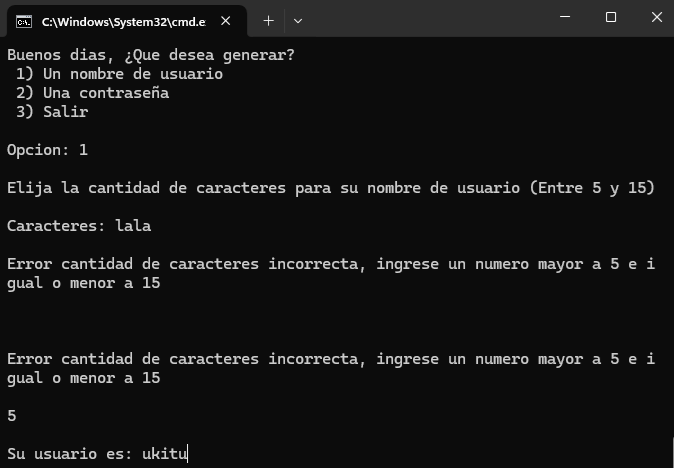
Todos tienen 10 caracteres, son aleatorios y alternan entre vocales y consonantes.

**Seleccionando la opción de generar un nombre de usuario e introduciendo una cantidad incorrecta de caracteres:**

Resultado 1:

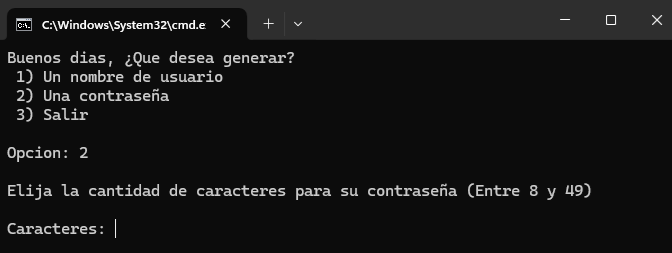


Resultado 2:

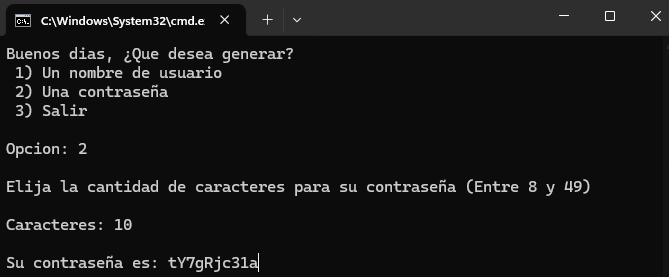


Sea el input 4, 16, vacío o letras el servidor seguirá pidiendo por un input correcto.

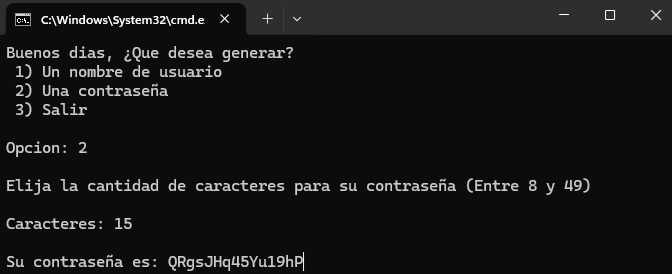
**Seleccionando la opción de generar una contraseña e introduciendo una cantidad correcta de caracteres:**



Resultado 1:

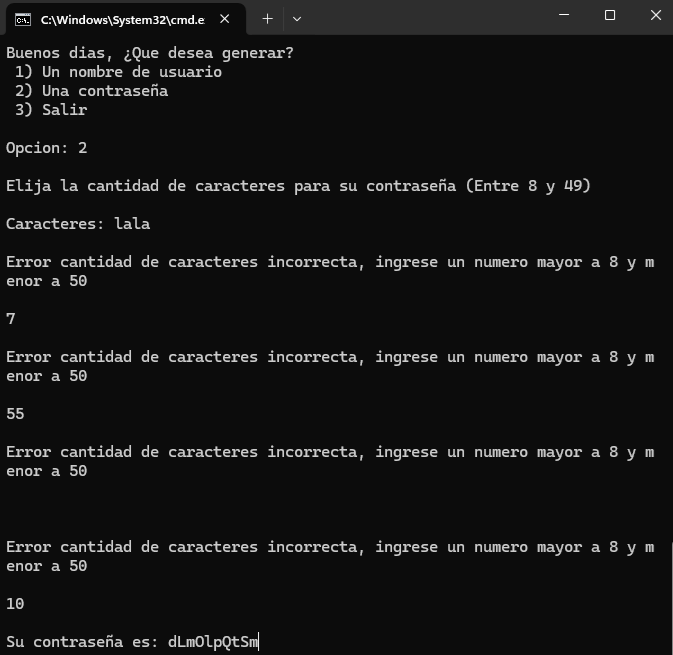


Resultado 2:



Ambos tienen la cantidad correspondiente a los caracteres ingresados por el usuario y están conformados al azar por letras minúsculas, mayúsculas y números.

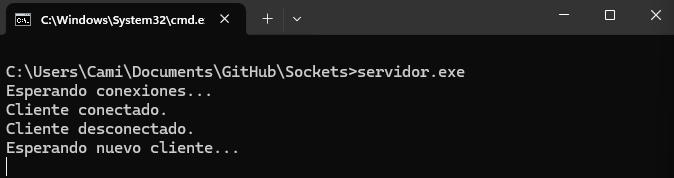
**Seleccionando la opción de generar una contraseña e introduciendo una cantidad incorrecta de caracteres:**

Resultado: 

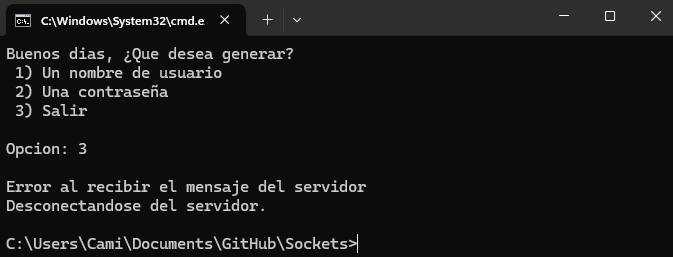
Sea el input 7, 50, vacío o letras el servidor seguirá pidiendo por un input correcto.

**Opción salir:**

Servidor:

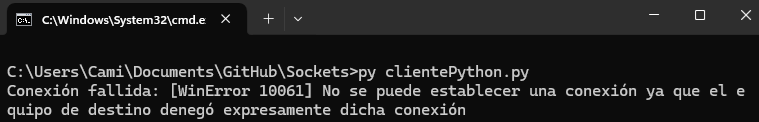


Cliente:

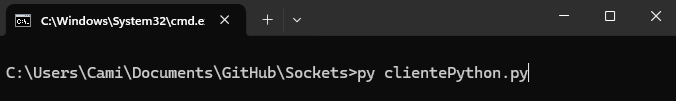


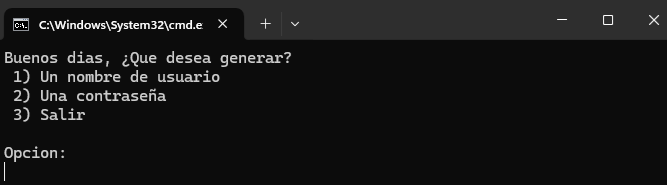
Pruebas del Servidor en C + Cliente en otro lenguaje (Python)

**Iniciando el Cliente sin que el Servidor este iniciado:**

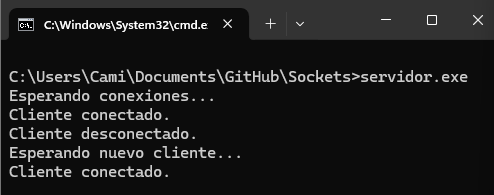


**Iniciando cliente:**

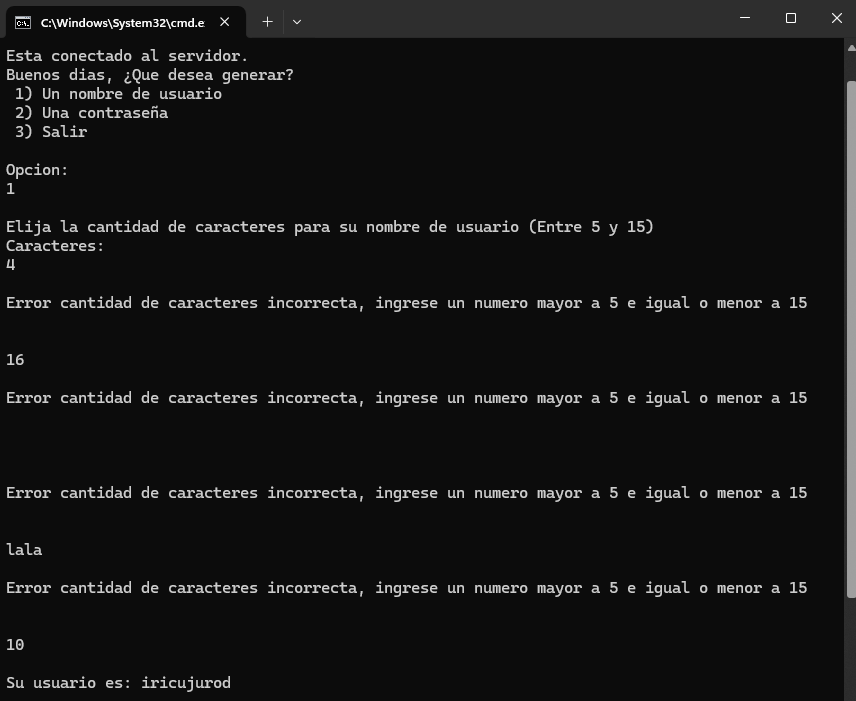
****



Nuevo cliente (en Python) se conectó al servidor:

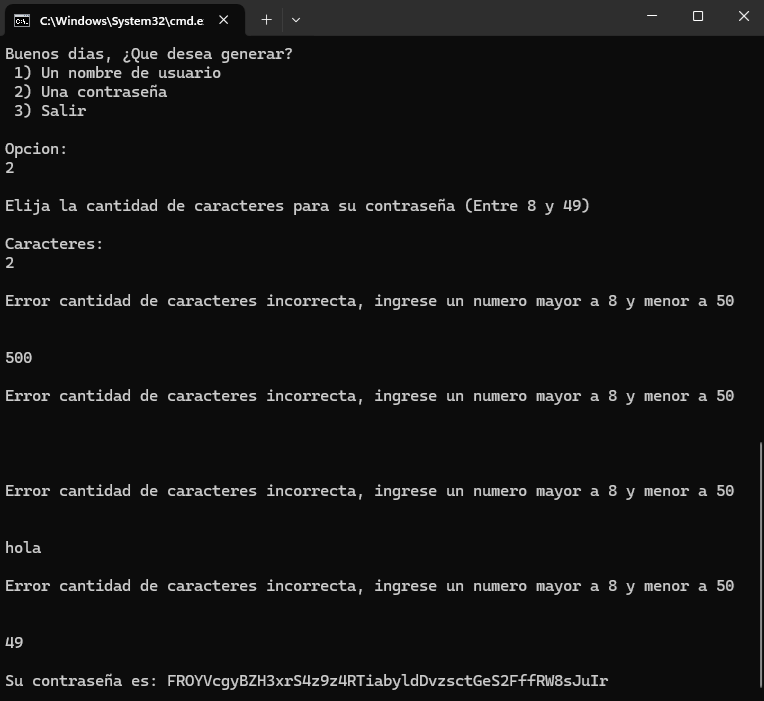


**Seleccionando la opción de generar un nombre de usuario e introduciendo una cantidad incorrecta de caracteres:**



Al igual que el cliente en C sea el input 4, 16, vacío o letras el servidor seguirá pidiendo por un input correcto.

**Seleccionando la opción de generar una contraseña e introduciendo una cantidad incorrecta de caracteres:**



Al igual que el cliente en c sea el input 3, 55, vacío o letras el servidor seguirá pidiendo por un input correcto.

**Opción salir:**

Cliente: A screenshot of a computer

Description automatically generated

Servidor:

