****

****

**Instituto Politécnico Nacional**

**ESCOM “ESCUELA SUPERIOR DE CÓMPUTO”**

*DESARROLLO DE SISTEMAS DISTRIBUIDOS*

*TAREA 5: CHAT MULTICAST*

PROFE: PINEDA GUERRERO CARLOS

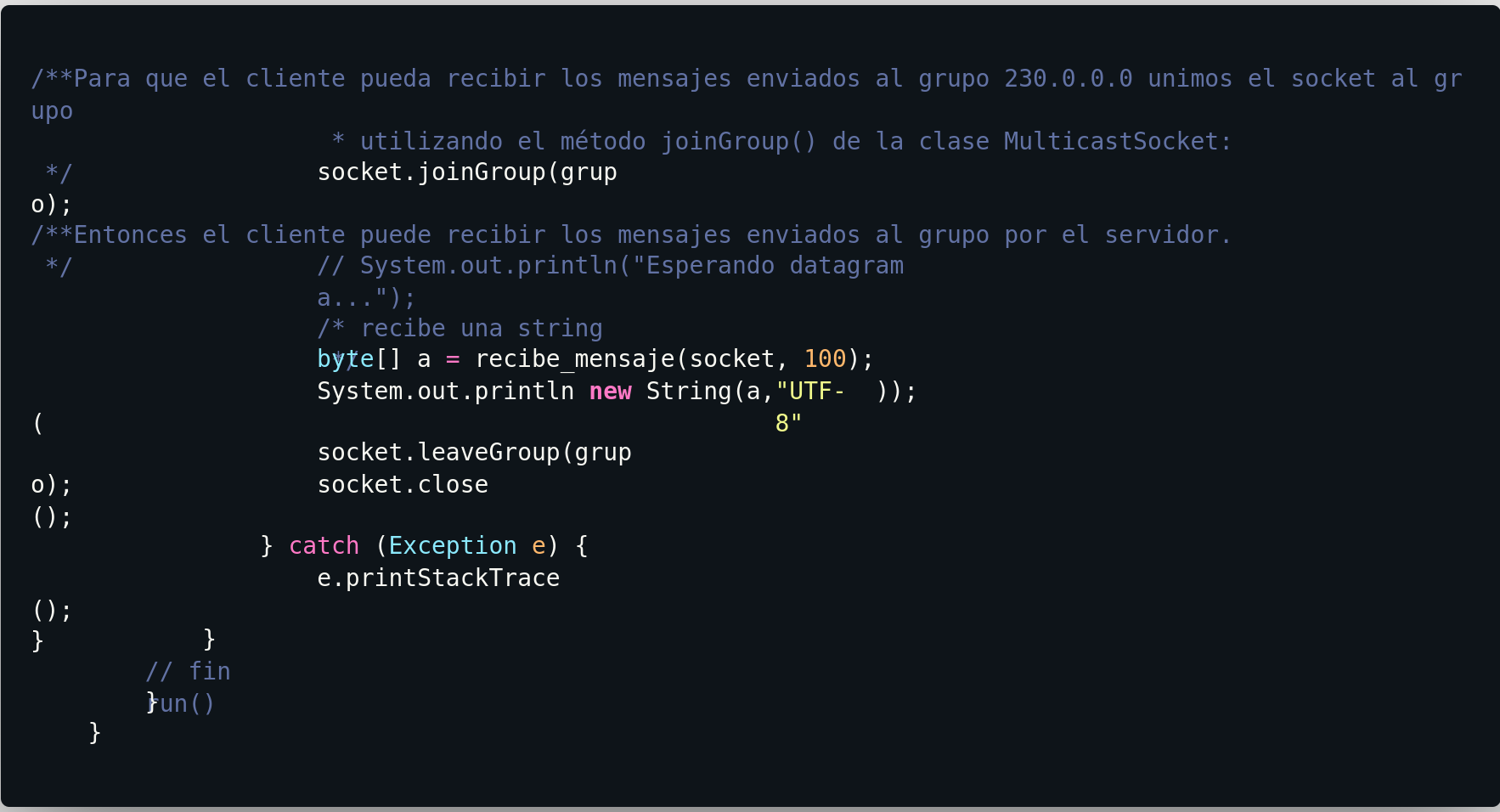
ALUMNO: Rojas Alvarado Luis Enrique

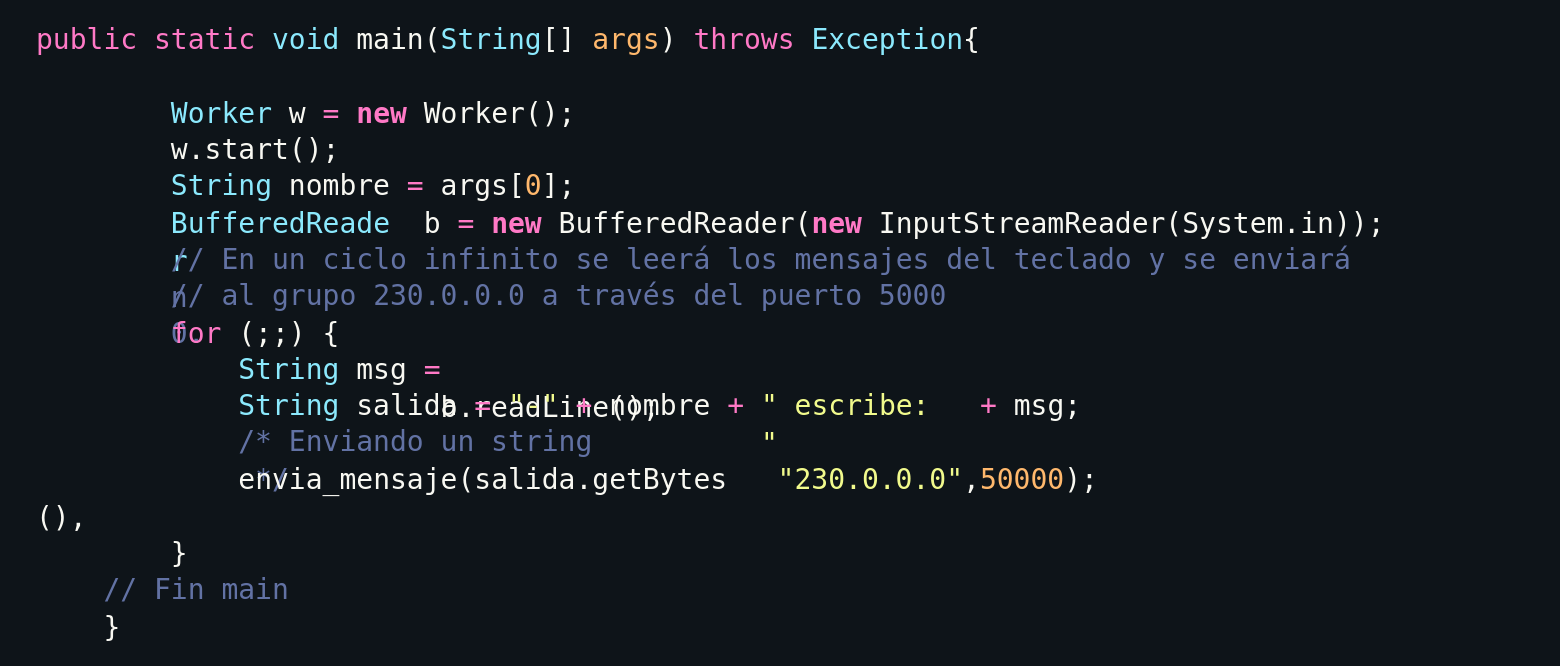
BOLETA: 2014010995

GRUPO: 4CM5

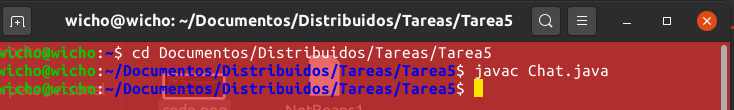
**DESARROLLO:**

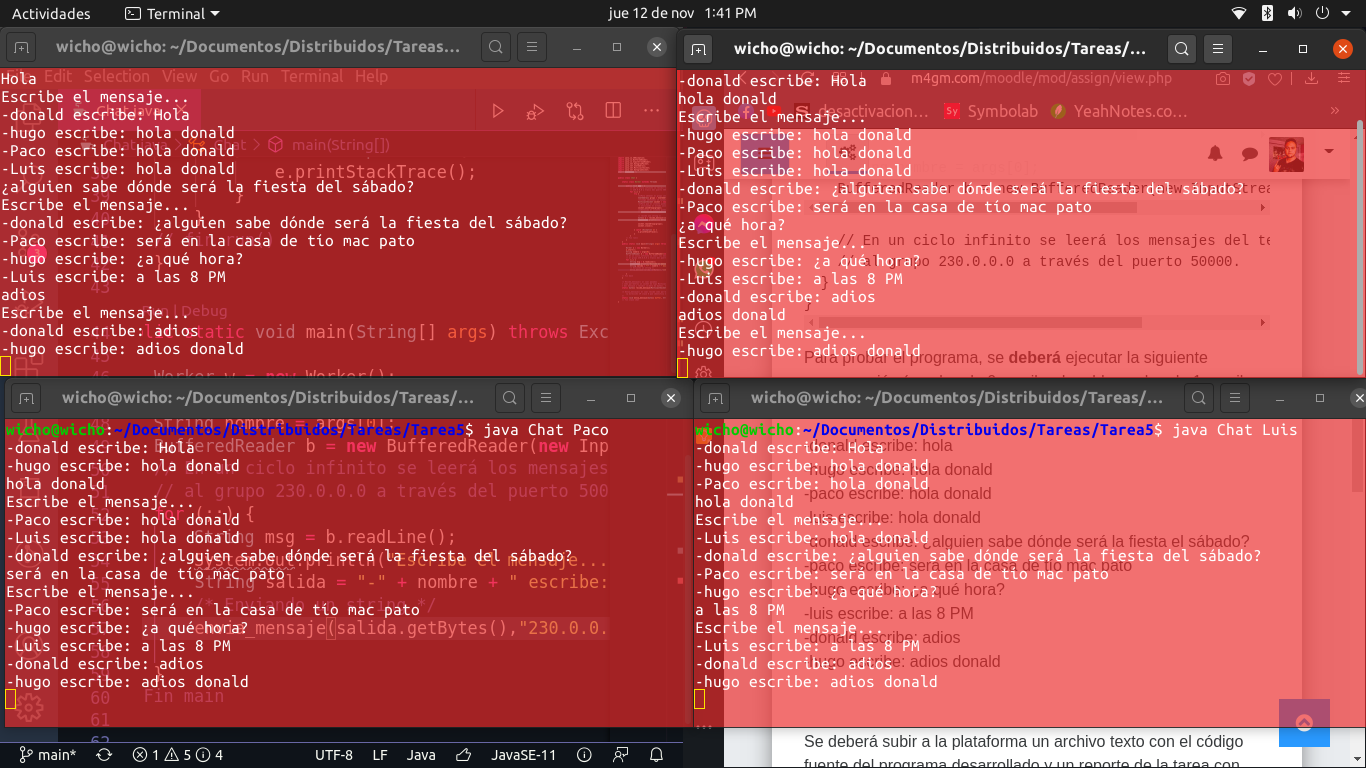
Para ésta práctica se necesita implementar un chat multicast con las funciones envia\_mensaje y recibe\_mensaje ya facilitadas por el profesor en clases pasadas, solo tenemos que completar la clase worker, donde recibiremos los mensajes, por lo que crearemos las instancias para crear el grupo y el socket multicast, así como la string de 100 caracteres para capturar de la mejor manera un mensaje. Y en el método main, en un ciclo infinito enviaremos el mensaje apoyandonos del método readline() en el grupo y socket ya definido anteriormente

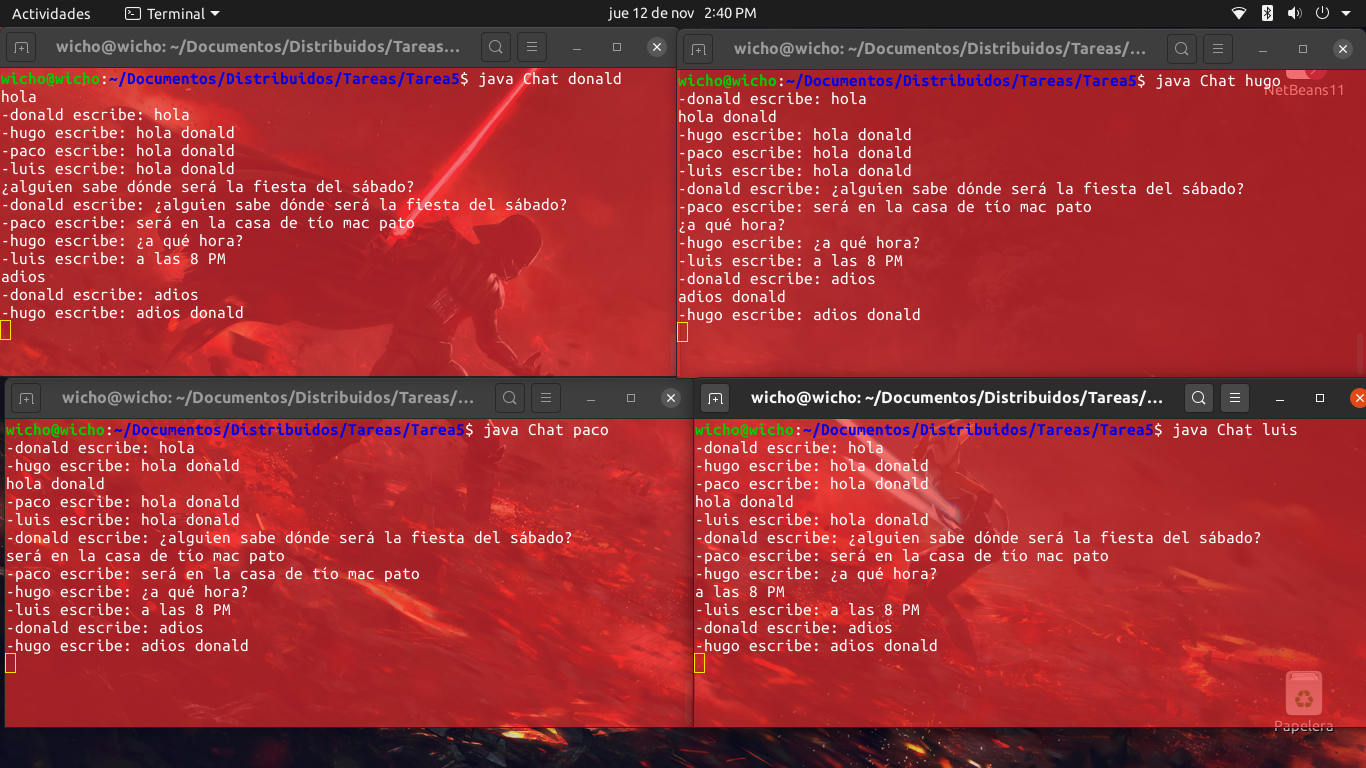


Recuperamos en la línea de comandos el nombre y lo adjuntamos al mensaje, posteriormente invocamos al método envia\_mensaje y quedará listo.

Compilando el programa:

Ejecutando el programa y haciendo la conversación sugerida por el profesor:





**CONCLUSIONES**

Para ésta práctica se pudo observar el comportamiento distribuido de un programa de chat multicast, en el cual, a diferencia del unicast no se tiene que cuidar la linealidad de la transmisión de mensajes, ya que estando dentro del grupo, es más fácil que se comuniquen sin tener una secuencialidad, respetando un orden, aquí todos reciben el dato (en este caso mensaje/cadena/string), que en unicast, un servidor (o un nodo en específico) se encargaba de “almacenar” los datos que se enviaban y trabajar con un hilo cada conexión o cliente, además de que tiene un especial cuidado con el paralelismo o concurrencia.