1.) Al ejecutar los archivos en “localhost” funciona bien.
Al introducir un tercer compañero podemos ver como el programa funciona con el primer solicitante que se lanza debido a que, en cuanto el aceptador recibe la conexión, devuelve el mensaje de que es el aceptador y cierra la conexión de inmediato sin que el segundo solicitante tenga la oportunidad de realizar la conexión con el aceptador (debido a que ya ha terminado su ejecución).

2.) El mecanismo de finalización de la conexión es enviar un asterisco (*).
Probándolo en la red local funciona perfectamente entre dos máquinas al igual que si lo probamos en “localhost”.
Al probarlo con un tercer compañero, la conexión se establece y tiene sentido entre el aceptador y el primer solicitante que le llega al puerto de conexión. El segundo solicitante, no recibe ni puede enviar mensajes.
Al cerrar la conexión entre el aceptador y el primer solicitante, en el segundo solicitante aparece ua traza de excepciones debido a que el aceptador ya ha cerrado el socket.
En cuanto al timeout, hemos introducido una espera de 10 segundos usando el método “this.socket.setSoTimeout(10000);” en el constructor de “MiSocketStream”. Como este método lanzará una excepción en cuanto se acabe el tiempo de espera, dentro del método run (tanto de Aceptador como de Solicitante) tenemos que poner un bloque “try-catch” en el hilo que se encarga de recibir mensajes.