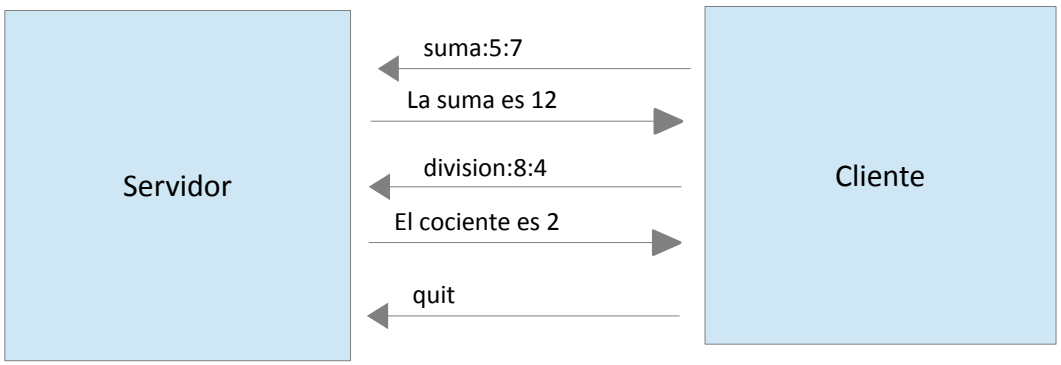
Ejercicios Socket TCP

4. La aplicación en red implementará un servicio "CALCULADORA DE NÚMEROS". Tanto servidor como cliente se ejecutan inicialmente en "*localhost*". De momento el servidor solo atiende a un cliente. La parte del servidor y del cliente será en modo texto.

Parte del servidor

a) Escucha peticiones en el puerto 6000
b) Acepta una conexión
c) Atiende a un solo cliente.
   - Como el cliente puede querer enviar varios mensajes para operar con la calculadora el servidor tendrá que iterar hasta que el cliente marque el final
   - el servidor leerá cada vez un mensaje del cliente (un String). Este mensaje siempre tiene el formato *operación:numero1:numero2*. Cuando operación valga "QUIT" significa que el cliente ya no va a enviar más información
   - el servidor procesa el mensaje y envía el resultado como un String al cliente
d) Cierra la conexión




```
Servidor conectado en puerto local 6000
 - Esperando conexión de cliente  ....
Aceptada conexión de  cliente  desde IP 127.0.0.1
en servidor suma:9:8
La suma es 17
```

Importa el fichero ServidorClienteCalculadoraTexto.AL. Contiene los dos proyectos que necesitas.

La parte del servidor está en ServidorCalculadoraTextoAL .
La clase Calculadora no tienes que modificarla.
Hay que completar la clase Servidor:

- añade los atributos necesarios (socket y flujos)
  - para lectura flujo BufferedReader, para escritura flujo PrintWriter
- el constructor crea el socket de bienvenida (por el que escuchará el servidor) y llama al método iniciar(). Trata las excepciones adecuadamente.
- completa el método iniciar()
  - acepta la conexión del cliente y obtiene los flujos

        - itera mientras el cliente no quiera terminar
            - cuando recibe un mensaje lo parsea y calcula el resultado con ayuda del método calcularResultado()
            - envía el resultado al cliente
    - el método privado calcularResultado() obtiene a partir del tipo de operación y los dos números la cadena resultado que habrá que enviar al cliente

Parte del cliente

a) se conecta al servidor
b) pide al usuario una operación a realizar (+ - * /)
c) pide al usuario los dos números que se necesitan
d) forma la línea de mensaje adecuada y la envía al servidor
e) recibe el resultado del servidor y lo muestra
f) repite el proceso mientras el usuario no teclee como operación 'q'

```
Cliente conectado al servidor de IP localhost/127.0.0.1
Qué operación quieres realizar +  -  *  / q (quit)
suma
Qué operación quieres realizar +  -  *  / q (quit)
+
Dame número
9
Dame número
8
suma:9:8
La suma es 17
Qué operación quieres realizar +  -  *  / q (quit)
```

Completa ClienteCalculadoraTextoAL:

- la clase Teclado no hay que modificarla. Servirá para pedir al usuario tanto la operación como los números (es una clase que encapsula el teclado)
- completa los atributos definiendo flujos (de lectura Scanner, de escritura PrintWriter) y socket para conectar con el servidor
- en el constructor conecta con el servidor, obtén los flujos y llama al método iniciar()
- en el método iniciar() incluye toda la lógica del cliente, leer operación, leer números, enviar mensaje al servidor y mostrar resultado obtenido del servidor
    - !Ojo! No olvides enviar al servidor el mensaje de final de envío de datos para que sepa que el cliente da por terminada la sesión
- cerrar flujos y conexión

Ejecuta el programa probando la parte servidor y cliente en la máquina local.

Por último, haz una ejecución incluyendo la parte cliente en la VM (VirtualBox) (opcional).

5. Modificaremos la parte servidor de la aplicación anterior para convertir al servidor en un servidor multihilo (concurrente) que pueda atender a varios clientes a la vez. La parte cliente es la misma.

Importa el proyecto ServidorCalculadoraTextoMultihiloAL.

- completa el constructor creando el socket de escucha
- en el método iniciar() haz que el servidor itere continuamente, acepte conexiones de clientes y cree un hilo para tratar cada una de estas conexiones
- la clase privada ManejadorConexionCliente será el hilo (la tarea Runnable) que tratará cada conexión. Complétala.
  - completa los atributos definiendo flujos y socket
  - el constructor crea los flujos y el hilo que tratará la conexión
  - incluye en el método run() toda la lógica del cliente (lo que ya has hecho en el ejercicio anterior)

```
Servidor conectado - Esperando conexiones de clientes ....
Aceptada conexión de  cliente 1 desde IP 127.0.0.1
En servidor con cliente 1 - El producto es 20
Aceptada conexión de  cliente 2 desde IP 127.0.0.1
En servidor con cliente 2 - La suma es 9
```

Haz una ejecución lanzando el servidor e:

- iniciando un par de clientes desde línea de comandos ó
- un primer cliente desde la máquina real y un segundo cliente desde la virtual.