1. Cree un ejemplo de cliente-servidor del protocolo UDP en el que el servidor escriba “recibido”, cada vez que el cliente mande el mensaje “enviado”. Si el cliente manda otro mensaje, por ejemplo “hola”, el servidor no escribe nada.
2. Usando JSON y el protocolo UDP, simule un proceso de autentificación. El cliente debe mandar el “usuario” y una contraseña. El servidor debe mandar al cliente “Acceso concedido” si el usuario es “admin” y la contraseña “12345”. En caso contrario debe mandar al cliente “Acceso denegado”.
3. Repita el ejercicio anterior usando el protocolo TCP.
4. Vamos a explorar el protocolo HTTP, para ello cree un servidor TCP que muestre por consola lo que el cliente le envíe. En su navegador web favorito escriba la dirección y el puerto de su servidor. Deberá ver un texto similar a:

```
GET / HTTP/1.1
Host: 127.0.0.1:5000
User-Agent: Mozilla/5.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: es-ES,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

   Ahora haga que el servidor escriba lo siguiente como respuesta al cliente:

```
HTTP/1.1 200 OK
Content-Type: text/html

<html><body>Hola mundo</body></html>
```

   La línea en blanco entre el “Content-Type: text/html” y la página html es necesaria. Después de escribirlo debe cerrar la conexión con el cliente usando el método “socket.end()”. Conéctese ahora desde cliente, debería tener un pequeño servidor web que muestra la página “Hola mundo”.
5. Usando datagramas y JSON escriba un servidor que reciba dos números y una operación (que puede ser suma, resta, multiplicación y división) debe devolver al cliente el resultado de la operación. Escriba un cliente que pruebe el funcionamiento del servidor. Las operaciones del cliente se deben pasar por linea de comandos. Por ejemplo:

```
$ node cliente.js 2 + 3
Resultado: 5
```

6. Se va a escribir una aplicación que almacene mensajes enviados por los clientes usando el protocolo UDP. En cada mensaje se debe almacenar el nombre del usuario y el mensaje que ha enviado. Los paquetes que recibe el servidor pueden ser de dos tipos:
   - Tipo lista: Si se recibe un mensaje de este tipo, el servidor debe devolver al cliente la lista de todos los mensajes recibidos.
   - Tipo mensaje: Se se recibe un mensaje de este tipo, el servidor debe almacenar el mensaje con el nombre de usuario y el mensaje.

   El cliente que se va a conectar a este servidor debe funcionar los la línea de comandos. Por ejemplo, cuando se escriba en la línea de comandos:

```
$ node cliente2.js lista
```

   Debe devolver el listado de mensajes almacenados en el servidor:

```
pepe: Hola
lolo: Hola
lucho: Adios
```

   Cuando se escriba en la línea de comandos:

```
$ node cliente2.js mensaje pepe “Adios mundo”
```

   Debe enviar al servidor el mensaje “Adios mundo” de parte del usuario “pepe”.