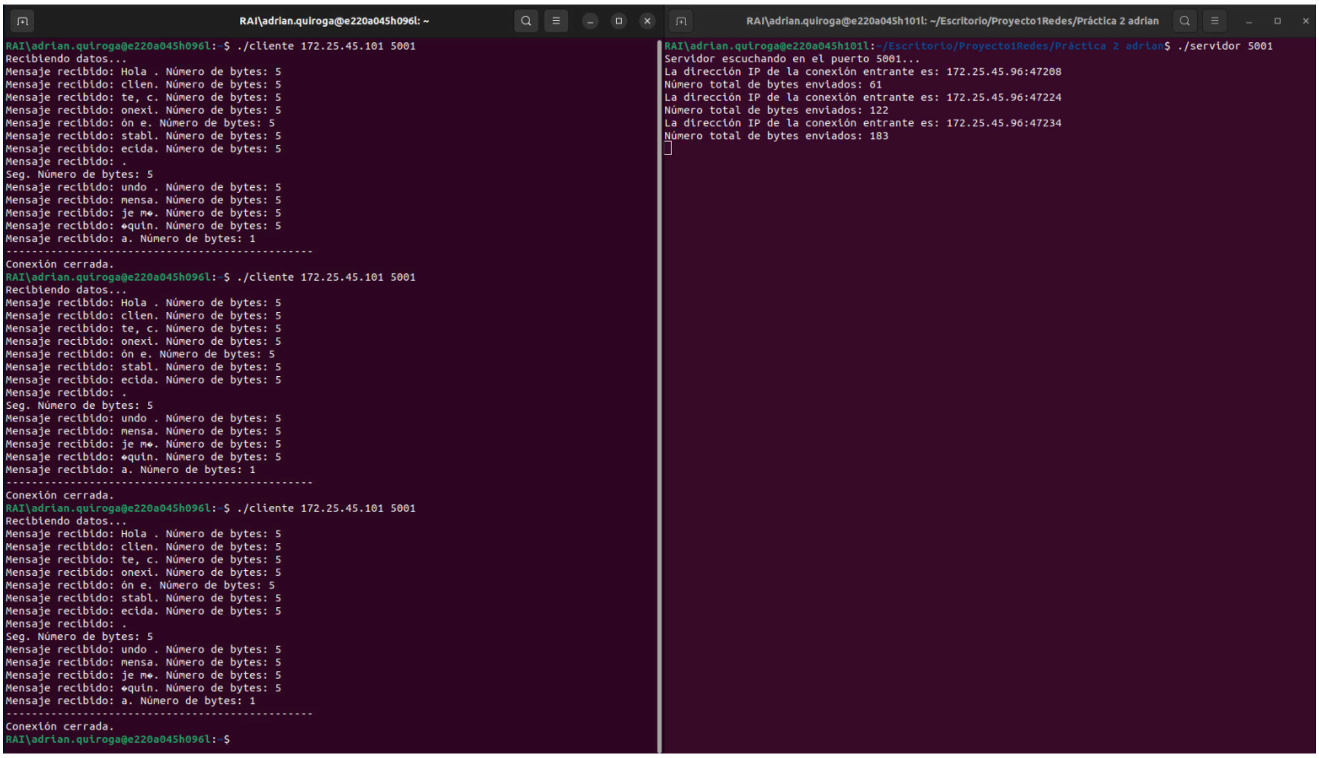
Cuestiones Redes

Apartados 1.c y 1.d



```
RAI\adrian.quiroga@e220a045h096l:~$ ./cliente 172.25.45.101 5001
Recibiendo datos...
Mensaje recibido: Hola . Número de bytes: 5
Mensaje recibido: clien. Número de bytes: 5
Mensaje recibido: te, c. Número de bytes: 5
Mensaje recibido: onexi. Número de bytes: 5
Mensaje recibido: ón e. Número de bytes: 5
Mensaje recibido: stabl. Número de bytes: 5
Mensaje recibido: ecida. Número de bytes: 5
Mensaje recibido: .
Seg. Número de bytes: 5
Mensaje recibido: undo . Número de bytes: 5
Mensaje recibido: mensa. Número de bytes: 5
Mensaje recibido: je m�. Número de bytes: 5
Mensaje recibido: �quin. Número de bytes: 5
Mensaje recibido: a. Número de bytes: 1
---------------------------------------------
Conexión cerrada.
RAI\adrian.quiroga@e220a045h096l:~$ ./cliente 172.25.45.101 5001
Recibiendo datos...
Mensaje recibido: Hola . Número de bytes: 5
Mensaje recibido: clien. Número de bytes: 5
Mensaje recibido: te, c. Número de bytes: 5
Mensaje recibido: onexi. Número de bytes: 5
Mensaje recibido: ón e. Número de bytes: 5
Mensaje recibido: stabl. Número de bytes: 5
Mensaje recibido: ecida. Número de bytes: 5
Mensaje recibido: .
Seg. Número de bytes: 5
Mensaje recibido: undo . Número de bytes: 5
Mensaje recibido: mensa. Número de bytes: 5
Mensaje recibido: je m�. Número de bytes: 5
Mensaje recibido: �quin. Número de bytes: 5
Mensaje recibido: a. Número de bytes: 1
---------------------------------------------
Conexión cerrada.
RAI\adrian.quiroga@e220a045h096l:~$ ./cliente 172.25.45.101 5001
Recibiendo datos...
Mensaje recibido: Hola . Número de bytes: 5
Mensaje recibido: clien. Número de bytes: 5
Mensaje recibido: te, c. Número de bytes: 5
Mensaje recibido: onexi. Número de bytes: 5
Mensaje recibido: ón e. Número de bytes: 5
Mensaje recibido: stabl. Número de bytes: 5
Mensaje recibido: ecida. Número de bytes: 5
Mensaje recibido: .
Seg. Número de bytes: 5
Mensaje recibido: undo . Número de bytes: 5
Mensaje recibido: mensa. Número de bytes: 5
Mensaje recibido: je m�. Número de bytes: 5
Mensaje recibido: �quin. Número de bytes: 5
Mensaje recibido: a. Número de bytes: 1
---------------------------------------------
Conexión cerrada.
RAI\adrian.quiroga@e220a045h096l:~$
```

```
RAI\adrian.quiroga@e220a045h101l:~/Escritorio/Proyecto1Redes/Práctica 2 adrian$ ./servidor 5001
Servidor escuchando en el puerto 5001...
La dirección IP de la conexión entrante es: 172.25.45.96:47208
Número total de bytes enviados: 61
La dirección IP de la conexión entrante es: 172.25.45.96:47224
Número total de bytes enviados: 122
La dirección IP de la conexión entrante es: 172.25.45.96:47234
Número total de bytes enviados: 183
```

Apartado 1.c

Cambios en servidor.c:

```c
   socketCliente = accept(sockserv, (struct sockaddr *)&direccionCliente, &tamano);
        if (socketCliente < 0)
        {
            perror("No se pudo aceptar la conexión\n");
            exit(EXIT_FAILURE);
        }
        char ipCliente[INET_ADDRSTRLEN];
        inet_ntop(AF_INET, &(direccionCliente.sin_addr), ipCliente, INET_ADDRSTRLEN);
        printf("La dirección IP de la conexión entrante es: %s:%d\n", ipCliente, ntohs(direccionCliente.sin_port));

        // Enviar primer mensaje
        valorMensaje = send(socketCliente, mensaje, strlen(mensaje), 0);
        if (valorMensaje < 0)
        {
            perror("Ocurrió un error al enviar el primer mensaje\n");
```

```
            exit(EXIT_FAILURE);
        }

        sleep(1);  // Esperamos 1 segundo para simular que el servidor tarda entre envío y envío

        valorMensaje = send(socketCliente, mensaje2, strlen(mensaje2) + 1, 0); // Aquí acordarse del fin de línea
        if (valorMensaje < 0)
        {
            perror("Ocurrió un error al enviar el segundo mensaje\n");
            exit(EXIT_FAILURE);
        }

        // Cerrar la conexión con el cliente
        close(socketCliente);
```

Desde el servidor enviamos los 2 mensajes. Y le añadimos un delay entre mensaje y mensaje de un segundo para simular que el servidor tarda.

Cambios en cliente.c:

```
 // Intentamos conectarnos al servidor
    if (connect(socketCliente, (struct sockaddr *)&direccionServidor, sizeof(struct sockaddr_in)) < 0) {
        perror("No se pudo conectar al servidor");
        exit(EXIT_FAILURE);
    }

   // sleep(2);

    // Recibimos los mensajes del servidor en una única llamada
    ssize_t n;
    size_t tamano_mensaje = 1000;  // Aumentamos el tamaño del buffer
    printf("Recibiendo datos...\n");

    n = recv(socketCliente, mensajeRecibido, tamano_mensaje, 0);
    if (n > 0) {
        // Añadimos un null terminator para manejarlo como cadena
        mensajeRecibido[n] = '\0';
        printf("Mensaje recibido: %s. Número de bytes: %zd\n", mensajeRecibido, n);
    }
```

Escribimos el sleep(2) para que después de aceptar la conexión espere y le dé tiempo a recibir los mensajes. En caso de no escribirlo, el segundo mensaje no tendría el suficiente tiempo como para recibirlo. Con el sleep nos da como salida: Mensaje recibido: Hola, que tal estás?Segundo mensaje máquina. Número de bytes: 46. Sin el sleep: Mensaje recibido: Hola, que tal estás?. Número de bytes: 21.

1.d

En el archivo servidor.c no escribimos ningún cambio. Y en el archivo cliente.c:

```c
// Intentamos conectarnos al servidor
    if (connect(socketCliente, (struct sockaddr *)&direccionServidor,
sizeof(struct sockaddr_in)) < 0) {
        perror("No se pudo conectar al servidor");
        exit(EXIT_FAILURE);
    }

    // Bucle para recibir datos
    ssize_t n;
    size_t tamano_mensaje = 10;
    printf("Recibiendo datos...\n");

    while ((n = recv(socketCliente, mensajeRecibido, tamano_mensaje, 0)) >
0) {
        // Añadimos un null terminator para manejarlo como cadena
        mensajeRecibido[n] = '\0';
        printf("Mensaje recibido: %s. Número de bytes: %zd\n",
mensajeRecibido, n);
    }
```

Vamos cambiando el tamaño del mensaje y va imprimiendo diferentes resultados. Con un tamaño de 10 bytes:

```
Recibiendo datos...
Mensaje recibido: Hola, que . Número de bytes: 10
Mensaje recibido: tal estás. Número de bytes: 10
Mensaje recibido: ?. Número de bytes: 1
Mensaje recibido: Segundo me. Número de bytes: 10
Mensaje recibido: nsaje máq. Número de bytes: 10
Mensaje recibido: uina. Número de bytes: 5
```

Con un tamaño de 5 bytes:

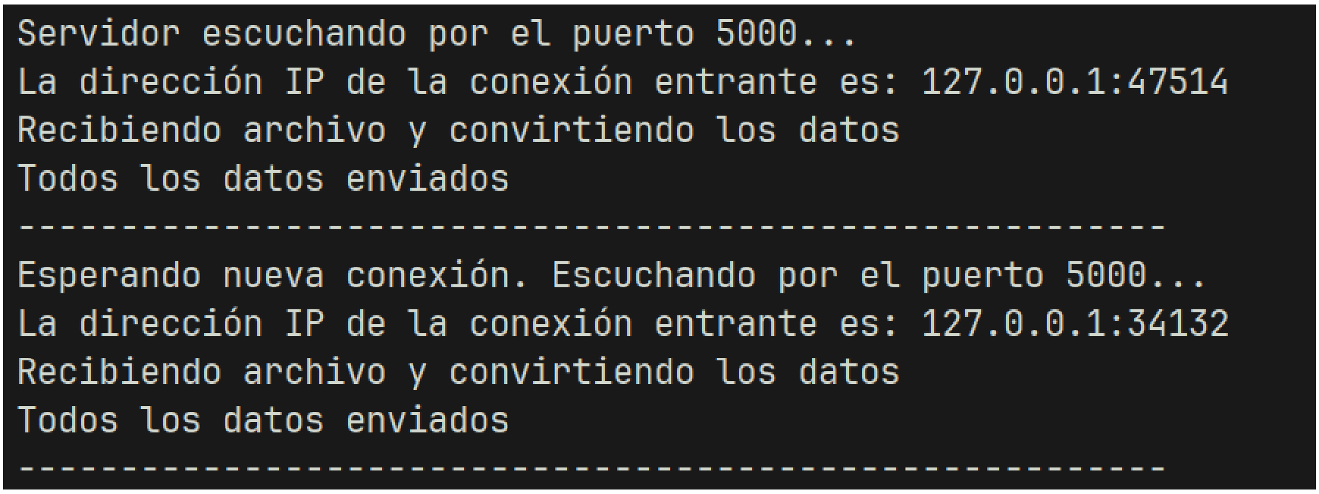
```
Recibiendo datos...
Mensaje recibido: Hola,. Número de bytes: 5
Mensaje recibido: que . Número de bytes: 5
Mensaje recibido: tal e. Número de bytes: 5
Mensaje recibido: stás. Número de bytes: 5
Mensaje recibido: ?. Número de bytes: 1
```

Mensaje recibido: Segun. Número de bytes: 5
Mensaje recibido: do me. Número de bytes: 5
Mensaje recibido: nsaje. Número de bytes: 5
Mensaje recibido: máq. Número de bytes: 5
Mensaje recibido: uina. Número de bytes: 5

Si introducimos un sleep para que de tiempo a enviar otro mensaje desde otra terminal, el servidor manejará las solicitudes de manera secuencias, primero respondiendo a una y después a otra

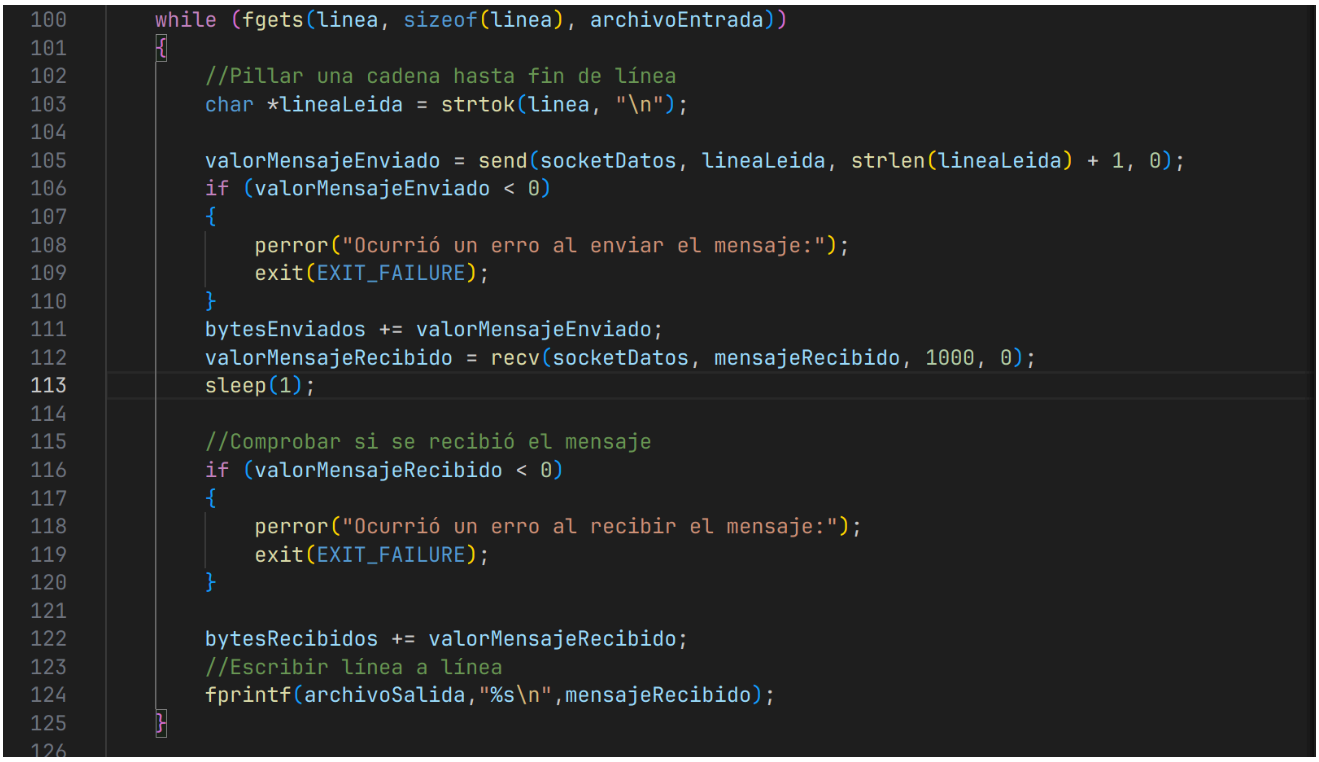
Apartado 3

Si introducimos un sleep para que de tiempo a enviar otro mensaje desde otra terminal, el servidor manejará las solicitudes de manera secuencias, primero respondiendo a una y después a otra



```
Servidor escuchando por el puerto 5000...
La dirección IP de la conexión entrante es: 127.0.0.1:47514
Recibiendo archivo y convirtiendo los datos
Todos los datos enviados
--------------------------------------------------------
Esperando nueva conexión. Escuchando por el puerto 5000...
La dirección IP de la conexión entrante es: 127.0.0.1:34132
Recibiendo archivo y convirtiendo los datos
Todos los datos enviados
--------------------------------------------------------
```

Cambios en el código: (Añadir un sleep(1)):



```c
    while (fgets(linea, sizeof(linea), archivoEntrada))
    {
        //Pillar una cadena hasta fin de línea
        char *lineaLeida = strtok(linea, "\n");

        valorMensajeEnviado = send(socketDatos, lineaLeida, strlen(lineaLeida) + 1, 0);
        if (valorMensajeEnviado < 0)
        {
            perror("Ocurrió un erro al enviar el mensaje:");
            exit(EXIT_FAILURE);
        }
        bytesEnviados += valorMensajeEnviado;
        valorMensajeRecibido = recv(socketDatos, mensajeRecibido, 1000, 0);
        sleep(1);

        //Comprobar si se recibió el mensaje
        if (valorMensajeRecibido < 0)
        {
            perror("Ocurrió un erro al recibir el mensaje:");
            exit(EXIT_FAILURE);
        }

        bytesRecibidos += valorMensajeRecibido;
        //Escribir línea a línea
        fprintf(archivoSalida,"%s\n",mensajeRecibido);
    }
```