Bước 2: chạy lệnh ./tcpserv01 & ; các hàm lần lượt được gọi là:
1. bzero(&servaddr, sizeof(servaddr));
2. bind(listenfd, (SA *) &servaddr, sizeof(servaddr));
3. listen(listenfd, LISTENQ);
4. accept(listenfd, (SA *) &cliaddr, &clilen);
Chương trình sẽ dừng ở hàm Accept();

Bước 3:
tcp 0 0 *:9877 *:* LISTEN
Tiến trình đang nghe trên cổng localhost:9877 theo giao thức tcp

Bước 4:

Địa chỉ 127.0.0.1 là địa chỉ IP loopback của máy local, dùng để tạo các IP connection ở trên cùng 1 máy.

Vì chạy server và client trên cùng 1 máy nên ta dùng địa chỉ này.

```
client                  |server
--------------------------------
Socket()                |
bzero()                 |
Inet_pton()             |
Connect()               |  Listen()
str_cli()               |
Fgets()                 |
Writen()                |   Fork()->close(listenfd)->str_echo(connfd)
Readline()              |
Fputs()                 |
```

Bước 5:
Chạy lệnh "netstat -a"
2 dòng mới xuất hiện liên quan đến việc thực hiện kết nối giữa client và sever:
tcp 0 0 localhost:59792 localhost:9877 ESTABLISHED
tcp 0 0 localhost:9877 localhost:59792 ESTABLISHED

Bước 6:

```
PID      PPID   TT     STAT    COMMAND                 WCHAN
22038   22036 pts/6    S       -bash wait4
17870   22038 pts/6    S         ./tcpserv01               wait_for_connect
19315   17870 pts/6    S         ./tcpserv01               tcp_data_wait
19314   22038 pts/6    S       ./tcpcli01 127.0.0.1      read_chan
```

PID(process id): id của tiến trình
PPID(parrent process id): id của tiến trình cha
STAT (status): Trạng thaí tiến trình (S: sleep)

tiến trình wait_for_connect là cha, tcp_data_wait là con, bởi vì sau khi thiết lập kết nối thành công thì cha mới fork ra một process để gửi nhận data

Cả hai tiến trình đều ở trạng thái ngủ vì chưa có dữ liệu truyền qua kết nối đó

Bước 7:
Chạy lệnh "netstat -a | grep 9877"
tcp 0 0 *:9877 *:* LISTEN
tcp 0 0 localhost:59792 localhost:9877 TIME_WAIT
1. Khi ta nhập kí tự EOF,hàm fgets sẽ trả lại null và hàm str_cli sẽ kết thúc (returns)
2. Khi hàm str_cli kết thúc và trở lại hàm main của client, gọi exit(0) .
3. Khi exit(0) được gọi, client socket sẽ bị đóng bởi kernel. Nó sẽ gửi một tín hiệu FIN tới server,và server TCP trả lời bằng một tín hiệu ACK. Lúc này server socket đang ở trang thái CLOSE_WAIT và client socket ở trạng thái FIN_WAIT_2.
4. Khi server TCP nhận được FIN, server còn được đặt vào trong hàm readline và readline trả về 0. Vì vậy hàm str_echo trả về server con chính.
5. Server con bị đóng bởi exit(0).
6. Các descriptors mở cửa server con đều đã bị đóng, client nhận một tín hiệu FIN từ server và gửi một tín hiệu ACK tới server rồi chuyển sang trạng thái TIME_WAIT, server nhận được ACK từ client và kết nối bị ngắt hoàn toàn.

Bước 8:
Gõ lệnh "ps -t pts/6 -o pid,ppid,tty,stat,args,wchan"

PID	PPID	TT	STAT	COMMAND	WCHAN
22038	22036	pts/6	S	-bash	read_chan
17870	22038	pts/6	S	./tcpserv01	wait_for_connect
19315	17870	pts/6	Z	[tcpserv01 <defu	do_exit

Tiến trình con của server rơi vào trạng thái Z (zombie), là trạng thái mà ví dụ đã gửi 1 tín hiệu SIGCHLD lên tiến trình cha nhưng chưa được xử lí -> chưa ngắt hoàn toàn, do đó cần phải xử lí tín hiệu SIGCHLD để ngắt hoàn toàn tiến trình con.