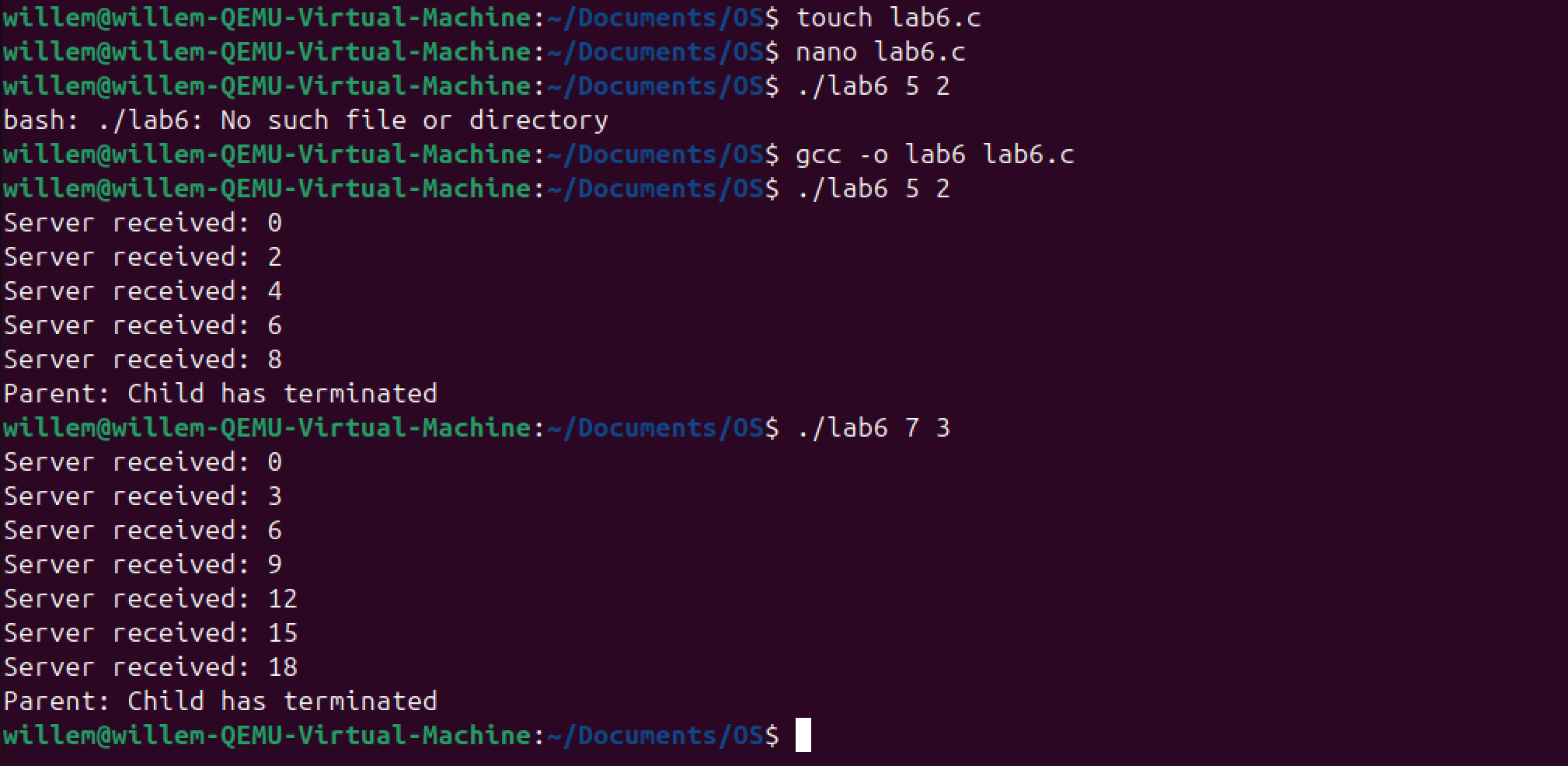

```
willem@willem-QEMU-Virtual-Machine:~/Documents/OS$ touch lab6.c
willem@willem-QEMU-Virtual-Machine:~/Documents/OS$ nano lab6.c
willem@willem-QEMU-Virtual-Machine:~/Documents/OS$ ./lab6 5 2
bash: ./lab6: No such file or directory
willem@willem-QEMU-Virtual-Machine:~/Documents/OS$ gcc -o lab6 lab6.c
willem@willem-QEMU-Virtual-Machine:~/Documents/OS$ ./lab6 5 2
Server received: 0
Server received: 2
Server received: 4
Server received: 6
Server received: 8
Parent: Child has terminated
willem@willem-QEMU-Virtual-Machine:~/Documents/OS$ ./lab6 7 3
Server received: 0
Server received: 3
Server received: 6
Server received: 9
Server received: 12
Server received: 15
Server received: 18
Parent: Child has terminated
willem@willem-QEMU-Virtual-Machine:~/Documents/OS$ 
```

Q2:

a. Which of the calls above are blocking and which are not?

Blocking:
connect() blocks until a connection is established
accept() blocks until a client connects
read() blocks until there is data to read
write() blocks if the network buffer is full

Not blocking:
socket() just creates a descriptor
bind() just associates a socket with an address and port
close() closes a socket descriptor
listen() sets up a queue, but doesn’t wait for it to full

b. Is this a form of direct communications or indirect communications?

This is direct communication because the child and parent are directly communicating with eachother and no middleman

c. What is the failure flag returned from connect() that indicates the server is not ready?

If connect fails, it will return ECONNREFUSED which means the connection was refused.

d. How would you change your program to communicate between processes in different machines?

You would use an actual IP address of a machine rather than a local IP address, and make sure the port being used is available