Angela Kerlin
CPE464-01

Program 2 Design Work

☆ Part 1

1) %0 M : flag 5

Packet Len	0x5 flag	Handle Len of Sender	Handle name of sender	num destination handles	...
2 bytes	1 byte	1 byte		1 byte	

for each destination handle:

...	Len of handle	handle name	...
	1 byte		

...	text message

- One destination handle

0x1800 (24)	0x5	0x6	"Client" (no null)	0x1	0x7	"Client2" (no null)	"hello \0"
2 bytes network order	1 byte	1 byte	6 bytes	1 byte	1 byte	7 bytes	6 bytes

- Two destination handles

0x2300 (35)	0x5	0x6	"Client" (no null)	0x2	0x7	"Client2" (no null)	0x7 ...
→ 2 bytes (network order)	1 byte	1 byte	6 byte	1 byte	1 byte	7 bytes	

0x7	"Client3" (no null)	"message \0"
1 byte	7 bytes	8 bytes

- %oB : Slag 4

Packet length	0x4 flag	Length of Sending Client's handle	Sending handle name	Text message
2 bytes (network) order	1 byte	1 byte	variable size	Variable size
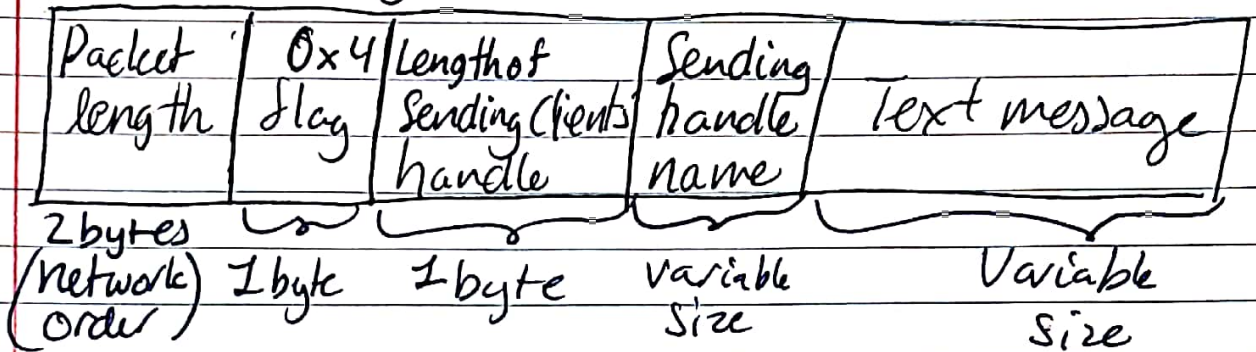
- %oL : flags 10

client to server

Packet Len (3)	0xA flag
2 bytes (networkorder)	1 byte

— Slag 11

serve to client

Packet Len	0xB Slag	Number of handles known by the serve
2 bytes (network order)	1 byte	4 bytes (network order)
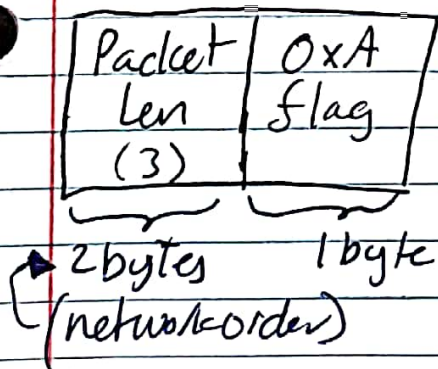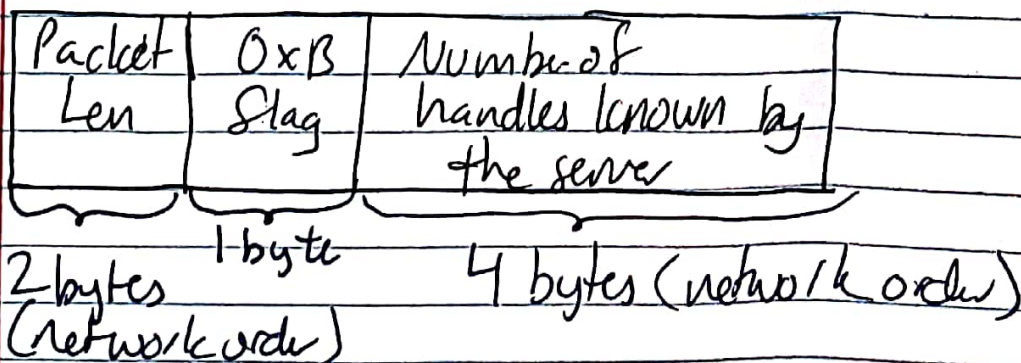
- flag 12   Server to client

Packet Len	0xC Flag	handle Len	handle

→ 2 bytes   1 byte   1 byte   Variable size
 (network order)

- flag 13   Server to client

Packet Len (3)	0xD Flag

→ 2 bytes   1 byte
  (network order)

. 0x0 E : flag 8

Packet Len (3)	0x8 Flag

→ 1 byte

2 bytes network order

- flag 9

Packet Len (3)	0x9 Flag

2 bytes   1 byte
 (network order)

- Flag 7   error packet, handle does not exist

Packet Len	0x7 flag	handle len	handle name (the one not found)
2 bytes (network order)	1 byte	1 byte	Variable length

- Flag 1   initial packet to server

Packet Len	0x1 flag	handle length	requested handle name
2 bytes (network order)	1 byte	1 byte	variable length

- Flag 2   Confirm handle name

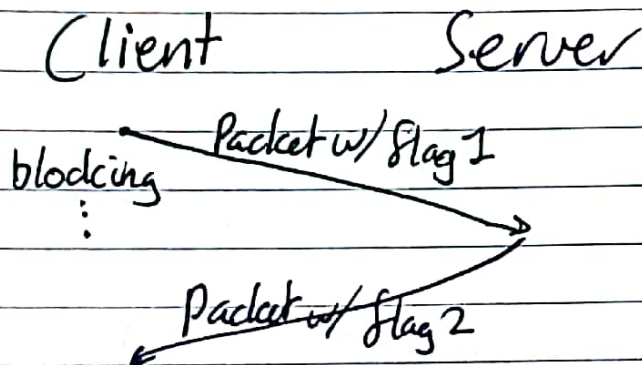| Packet Len	0x2 flag
2 bytes (network order)	1 byte

- Flag 3   error bad handle requested

Packet Len	0x3 flag
2 bytes Network order	1 byte

2) ⓐ Connection setup

- Case 1: handle is valid, no errors

Client   Server

blocking ——— Packet w/ flag 1
  ⋮
          Packet w/ flag 2

- Case 2: handle already exists, server throws error

Client   Server

blocking ——— Packet w/ flag 1
  ⋮   Packet w/ flag 3
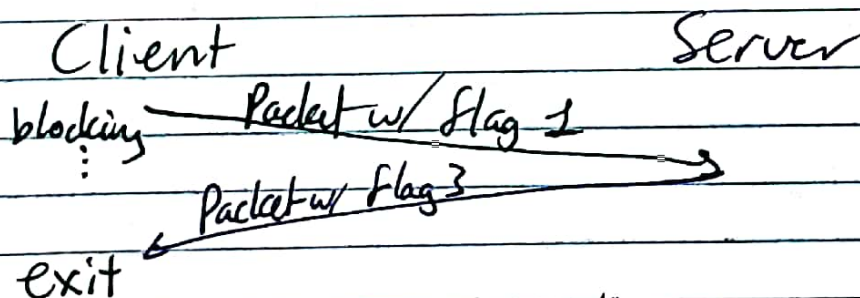
exit
~ Case 3: handle is formatted badly (too long, includes whitespace, etc). Client processes handle and does not send it to server, client exits

ⓑ Sending a message to a single client

- Case 1: no errors, destination clients exists

Client 1   Server   Client 2
     Packet w/ flag 5
noblocking ———   Packet w/ flag 5
            checks ———   received
            handle
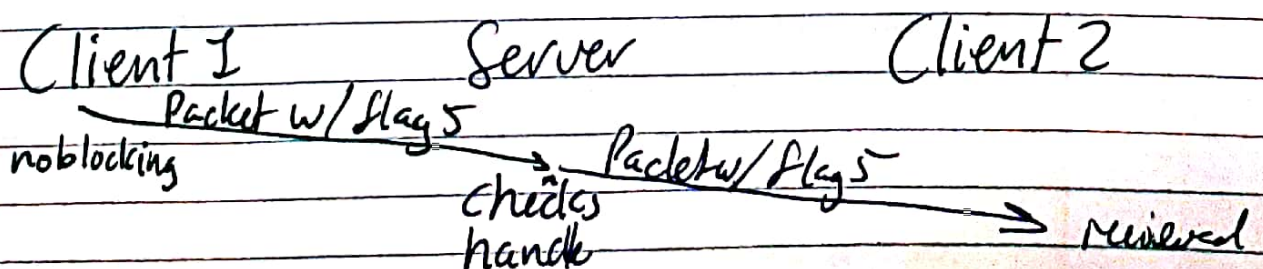
– Case 2: destination handle doesn't exist

Client 1   Server   Client 2
    Packet w/flag 5
    ─────────────────────►
                          looks up destination
                          handle → fails
    Packet w/flag 7
    ◄─────────────────────
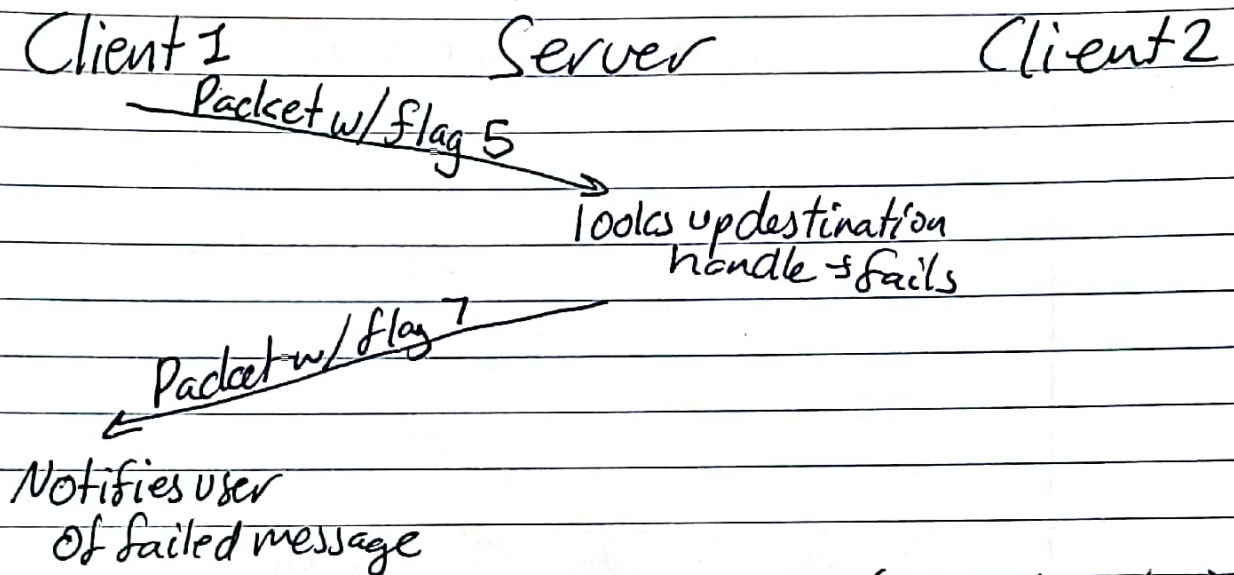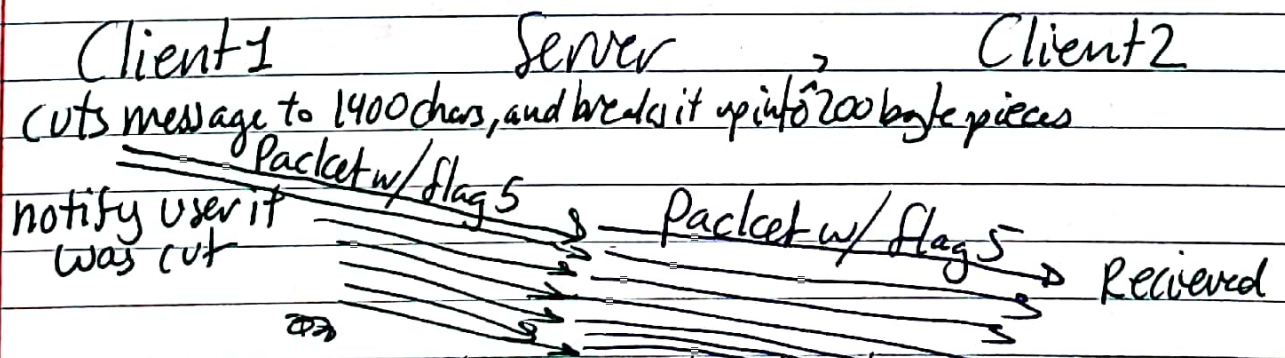Notifies user
of failed message

                                              (1400 characters)
– Case 3: message exceeds max length↑, client
       cuts it off

Client 1   Server   Client 2
Cuts message to 1400 chars, and breaks it up into 200 byte pieces
notify user if   Packet w/flag 5   Packet w/flag 5
was cut ───────────────────────── ──────────────────► Recieved

Ⓒ Send a message to multiple clients

– Case 1: no errors, handles are all valid

Client 1   Server handles   Client 2
    Packet w/flag 5   lookup packet w/5 → recieved
    to 2,3,4   packet w/5 Client 3
                                      → recieved
                          Packet w/5   Client 4
                                      → recieved
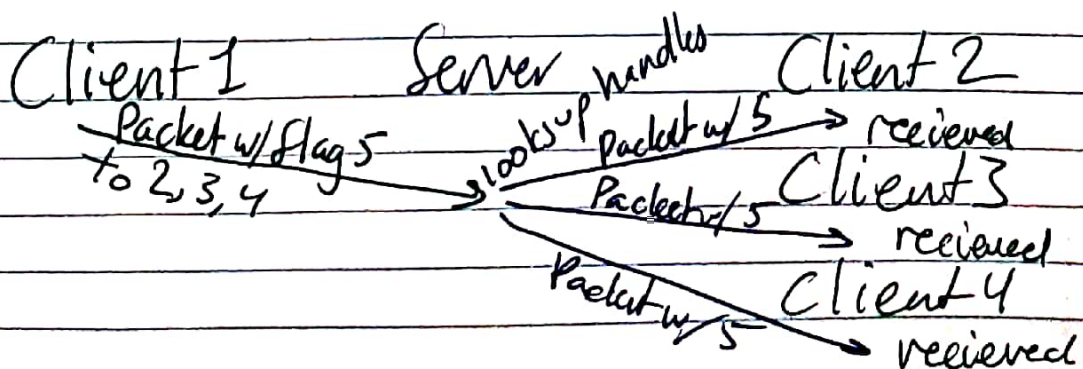
- Case 2: One bad handle, server sends to valid, and errors to client 1

Client 1   Server   Client 2

→ Packet to clients 2,3,4 →   message
                                              received

                    look up handles,   Client 3
                    "Client 4" is bad*
                                          message
                                              recieved

← Packet w/ flag 7
    and "client 4"

notify user of error
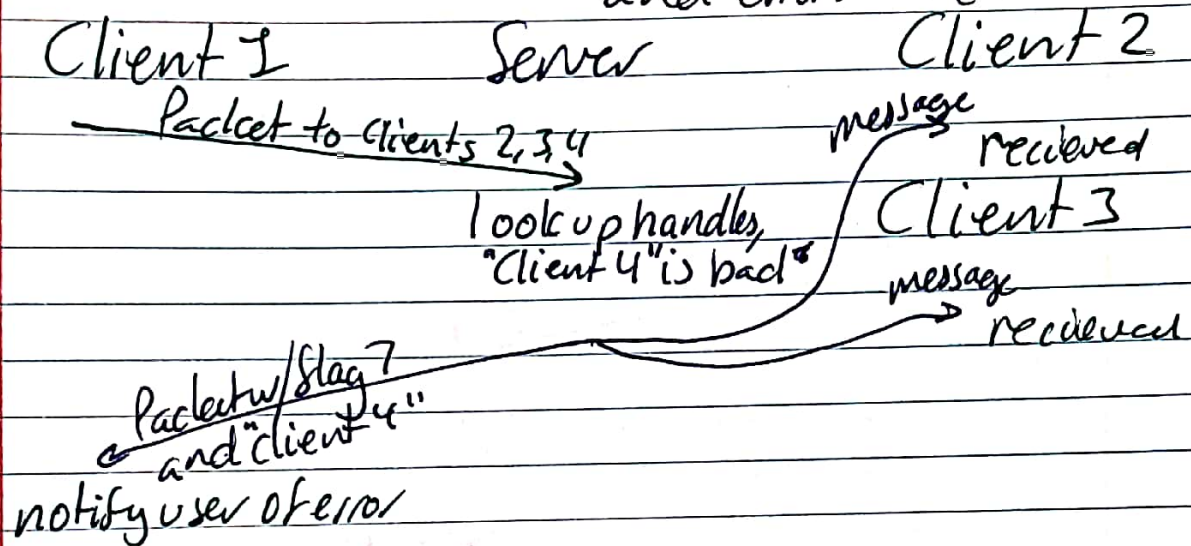
- Case 2: two bad handles, no good handles,
  server sends two error packets (flag 7)

Client 1   Server

→ Packet comm to clients 2 and 3 →
                                          look up handles

← Packet flag 7 "client 2"
notify user ← Packet flag 7 "client 3"

notify user
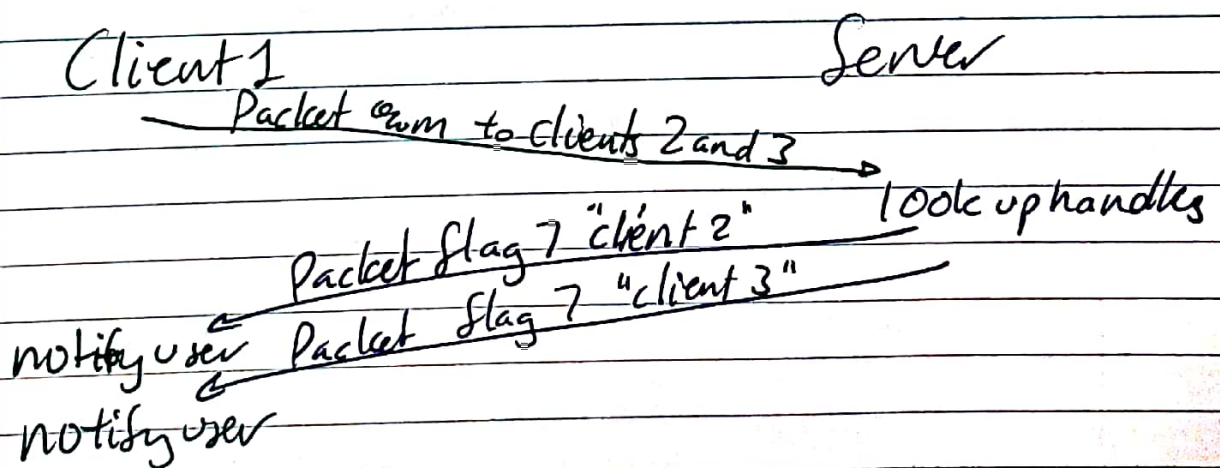
① Broadcast a message

~ Case 1: broadcast message 200 bytes or smaller

Client 1   Server

    200b packet ⟶

                                    Client 2

look up all clients 200 packet ⟶

                       200 packet⟶ Client 3
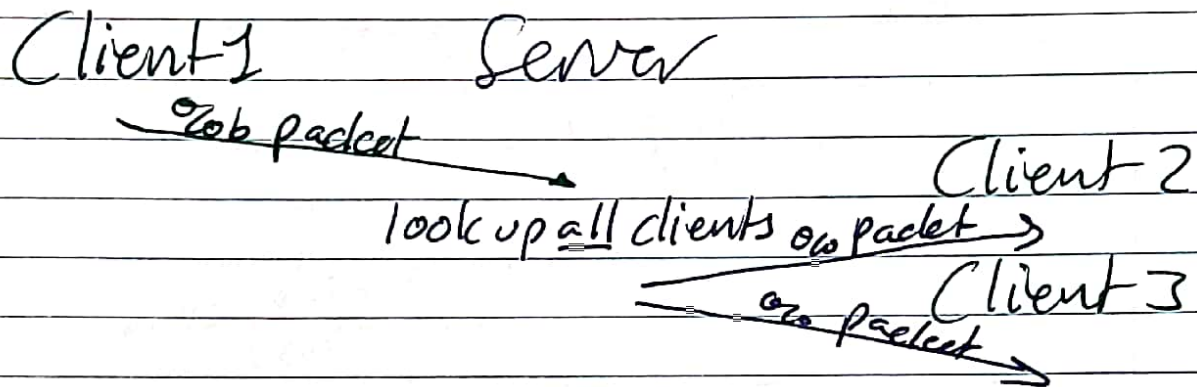
~ Case 2: broadcast message is greater than 200
bytes, client breaks message into 200 byte
pieces and sends separately

Client 1   Server

user types long message
        200b packet ⟶

                                   Client 2
        200b packet ⟶ look up all
                  clients
        200b packet ⟶ look up all
                                     Client 3
                   look up all
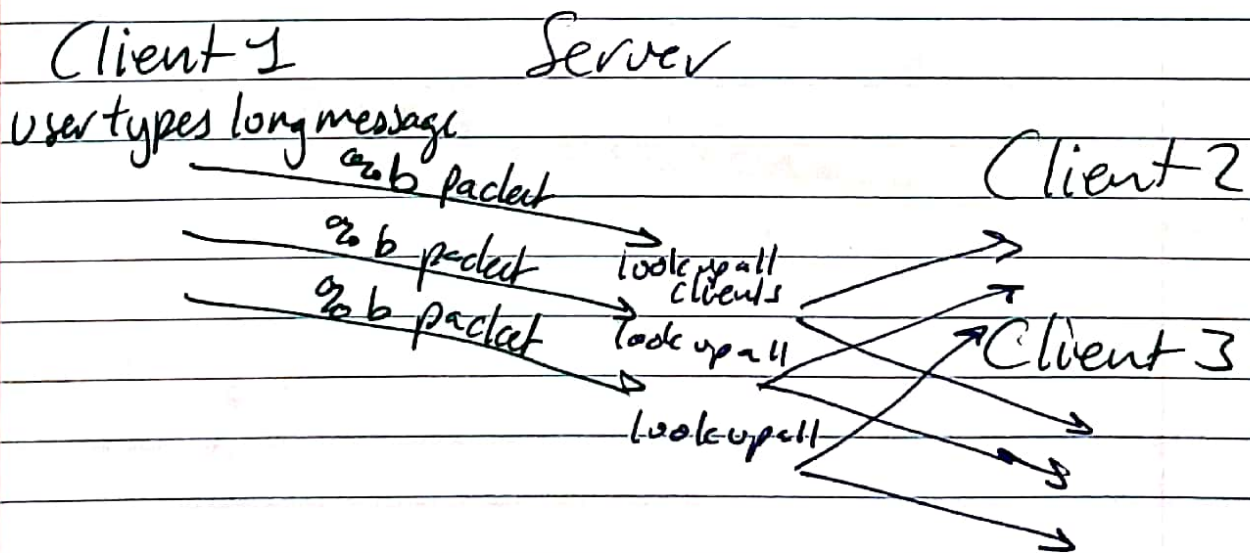
(e) Sending a message of 450 charaters

Client will break it into 3 seperate messages
and the server will just see it as 3 messages

Client 1   Server   Client 2

User types long message
client breaks it up

$\longleftarrow$ flag = 5 200byte message
$\longleftarrow$ flag = 5 200byte message   flag = 5 $\longrightarrow$
flag = 5 50 byte message $\longrightarrow$ flag = 5 $\longrightarrow$
                              flag = 5 $\longrightarrow$
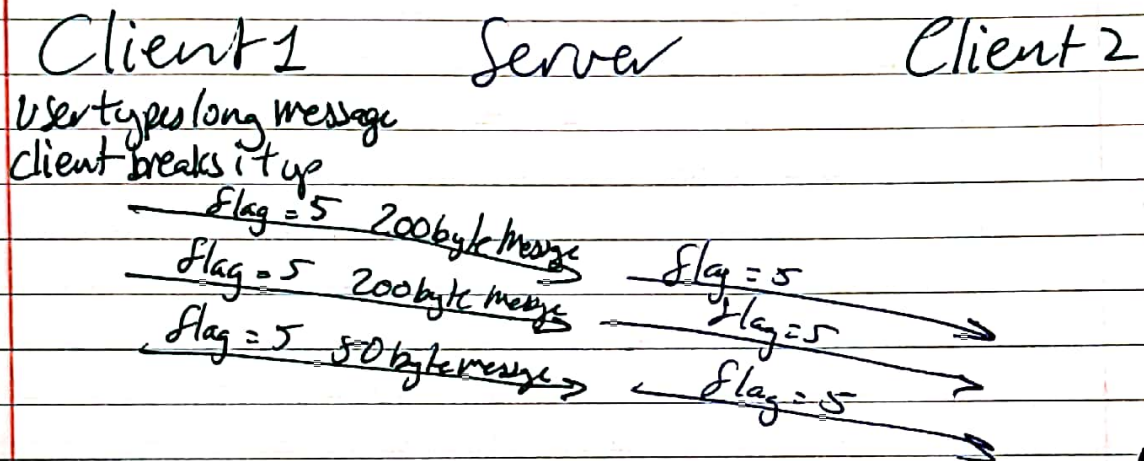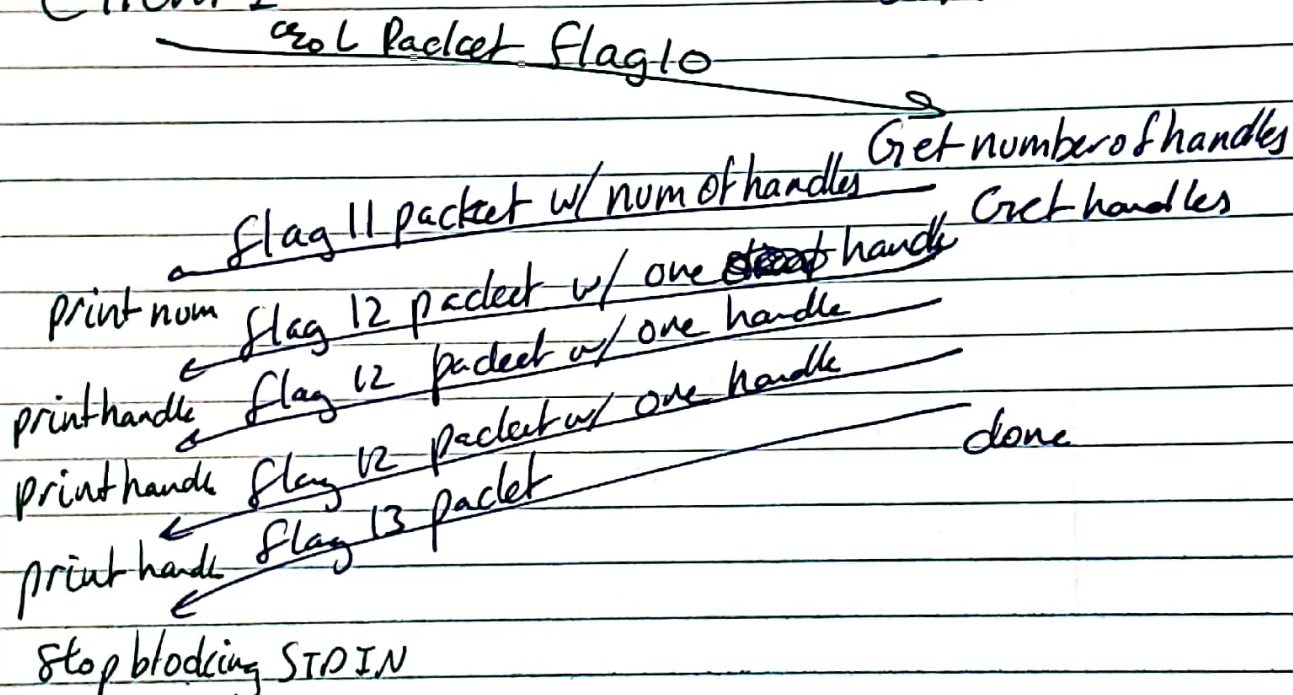
(f) Listing the handles

- Case 1: no errors, assume there are 3 clients

Client 1   Server

           flag 10 packet flag 10

                                         Get number of handles

         flag 11 packet w/ num of handles   Get handles

print num   flag 12 packet w/ one sent handle

print handle flag 12 packet w/ one handle

print handle flag 12 packet w/ one handle

print handle flag 13 packet   done

Stop blocking STDIN

(g) ending a connection

- Case 1: no errors, client sends flag 8 message,
        server ACKS then removes handle from system

Client   Server

        flag 8 packet

blocks

      :

      flag 9   remove client socket from
                             poll, remove handle
                             from table

exits