Student ID	Points		
	91		
Program Feature	Max	Additional remarks and clarification	
Basic programming block		block's points	42
Working makefile (compiles with make)	5	Ignore if dependencies are not perfect.	
Makefile has targets all and clean	2		
Executables have same names as in text	2		
Client and server take given arguments (which, order)	2		
Client: takes input, communicates over UDP, terminates, does not segfault	10	This is not about leaks	
Server: communicates over UDP, does not segfault	10	This is not about leaks	
Runs with given pre-code	2	Always use an original send_packet version for testing.	
Client compiles without fixes	3		
Server compiles without fixes	3		
Checks for system call return value	3	(a) memory allocation, (b) network operations, (c) others	
Server		block's points	15
Server waits for UDP packets on the port given on the command line	1		
Server does not busy wait	1	A select that waits 10ms for packets or timeout is not busy waiting. That is OK even if it does nothing after busy-waiting.	
Server does not sleep without listening for something.	1		
Registration list for clients is implemented.	3	The actual data structure is unimportant. Full points only if it can grow dynamically.	
Registration list for clients is not leaking memory.	3	The list may still grow infinitely: if a nick's record is reused, but disabled without heartbeat, it may never shrink. That is OK. But no double nicks in the list.	
Lookup sends correct information.	1		
Lookup responds to client via address taken from recvfrom.	3		
Server identifies correct IP and port from recvfrom in REG	2	Correct behaviour although not perfect: if the client sends from localhost, the server registers 127.0.0.1 as IP.	
Client basics		block's points	6
Client registers itself at the server at start	1		
Client quits when initial registration fails	1		
Client implements an event loop centered on select	1	event loop means that there is only ONE select where everthing happens. select() is an explicit demand from the assignment. poll() and epoll() are not portable.	
Waiting time is minimum of retrans times for all clients (or heartbeat)	1		
Sending client implements a cache of IP/port addresses for each receiver nick	1		
Client has no valgrind warning when terminating with QUIT	1		
Stop-and-wait		block's points	11
Client implements stop-and-wait semantics	5	Semantics: it does not have to be actual stop-and-wait. Semantics mean max one message in flight, retrans after timeout, msg and ack loss detected. OK even if it only works between exactly two clients.	
Receiver does not print duplicates	1	needed to avoid duplicates; "sender" can be nick or IP/port - neither is 100% safe but both are accepted	
Sender has one retransmission timeout per client	1		
Sender maintains 1 sequence number per nick	1	equally ok to have one seq no per IP/port	
Receiver maintains 1 sequence number per sender	1	client does distinct stop-and-wait instances with each other client; "sender" can be identified by nick or IP/port - neither is 100% safe but both are accepted	
Receiver always answers MSG with an ACK containing same sequence number	1	We made this choice. Other choices would be possible, but this is explicitly stated in the assignment.	
Sender has only one packet in flight per peer	1	We don't want sliding window here	
Asynchronous client behaviour		block's points	7
Client can wait for stdin and for retransmission timeout	5	(at the same time)	
Client can read from stdin although a previous message is not ACKed yet	1		
After 2 (or 3) timeouts, client executes lookup the lookup operation again	1	note that we allow these particular lookup operations to be completely synchronous and block everything else	
Heartbeat		block's points	5
Client sends heartbeat is sent every 10 seconds	1		
Server does no longer return client info if no heartbeat for 40 seconds	1	It is not important how this is solved, with own timeout, by list removal, by flag ...	
Heartbeat protocol makes sense	1	Students can use REG as we intended. Heartbeat can also be a different message.	
Heartbeat does not use stop-and-wait	1		
Server implements lookup disabling when heartbeat expires	1	There a many possible solutions, the server does not need to wait for the next expiration.	
Blocking clients		block's points	5
Client implements a list of blocked clients	1		
Client receives messages from client, but does not print them	2		
Client takes messages from stdin to blocked clients, but discards them silently	1		
Client can unblock clients	1		