PROGRAM CODE

server.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <pthread.h>

#define PORT 8000
#define SIZE 100

typedef struct packet {
	int data;
	int type; // SEQ (0) or ACK (1)
	int seq; // Sequence number (0 or 1)
} packet;

void main() {
	int server_fd, client_fd;
	struct sockaddr_in address;
	int addrlen = sizeof(address);
	int arr[SIZE], k = 0;

	for(int i = 0; i < SIZE; i++)
		arr[i] = -1;

	printf("Stop and Wait ARQ\nTCP Server\n");

	if((server_fd = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
		printf("Socket creation failed!\n");
		exit(1);
	}

	address.sin_family = AF_INET;
	address.sin_addr.s_addr = INADDR_ANY;
	address.sin_port = htons(PORT);

	if(bind(server_fd, (struct sockaddr*) &address, addrlen) < 0) {
		printf("Socket binding failed!\n");
		exit(1);
	}

	if(listen(server_fd, 5) < 0) {
		printf("Listening failed!\n");
		exit(1);
	}

	if((client_fd = accept(server_fd, (struct sockaddr*) &address,
(socklen_t*) &addrlen)) < 0) {
		printf("Connection failed!\n");
		exit(1);
	} else {
		printf("Connected to client.\n");
	}

	packet p;
	int flag = -1;

	while(1) {
```

```
            int status = recv(client_fd, &p, sizeof(packet), 0);

            if(status < 0) {
                  printf("Receive failed!\n");
            } else if (status == 0) {
                  printf("Receive completed.\nArray: ");

                  for(int i = 0; arr[i] != -1; i++) {
                        printf("%d ", arr[i]);
                  }

                  printf("\n");

                  break;
            } else {
                  if(flag != p.seq) {
                        arr[k] = p.data;
                        k++;
                  }

                  printf("Received: %d (SEQ %d)\n", p.data, p.seq);
                  flag = p.seq;

                  p.type = 1;
                  p.seq = (p.seq + 1) % 2;

                  if(rand() % 5 != 2) {
                        if(send(client_fd, &p, sizeof(packet), 0) < 0) {
                              printf("Send failed!\n");
                        } else {
                              printf("Sent: ACK %d\n", p.seq);
                        }
                  } else {
                        printf("ACK %d lost\n", p.seq);
                  }
            }
      }

      close(server_fd);
      close(client_fd);
}
```

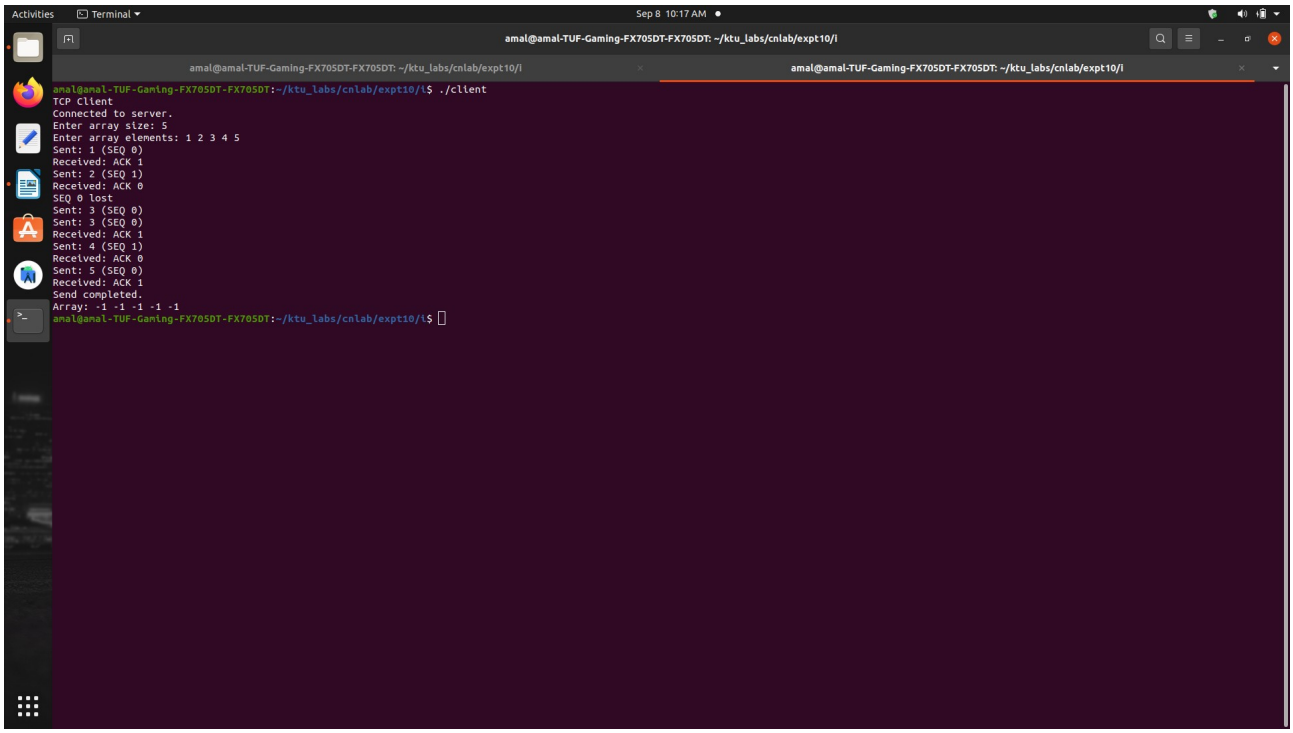
client.c

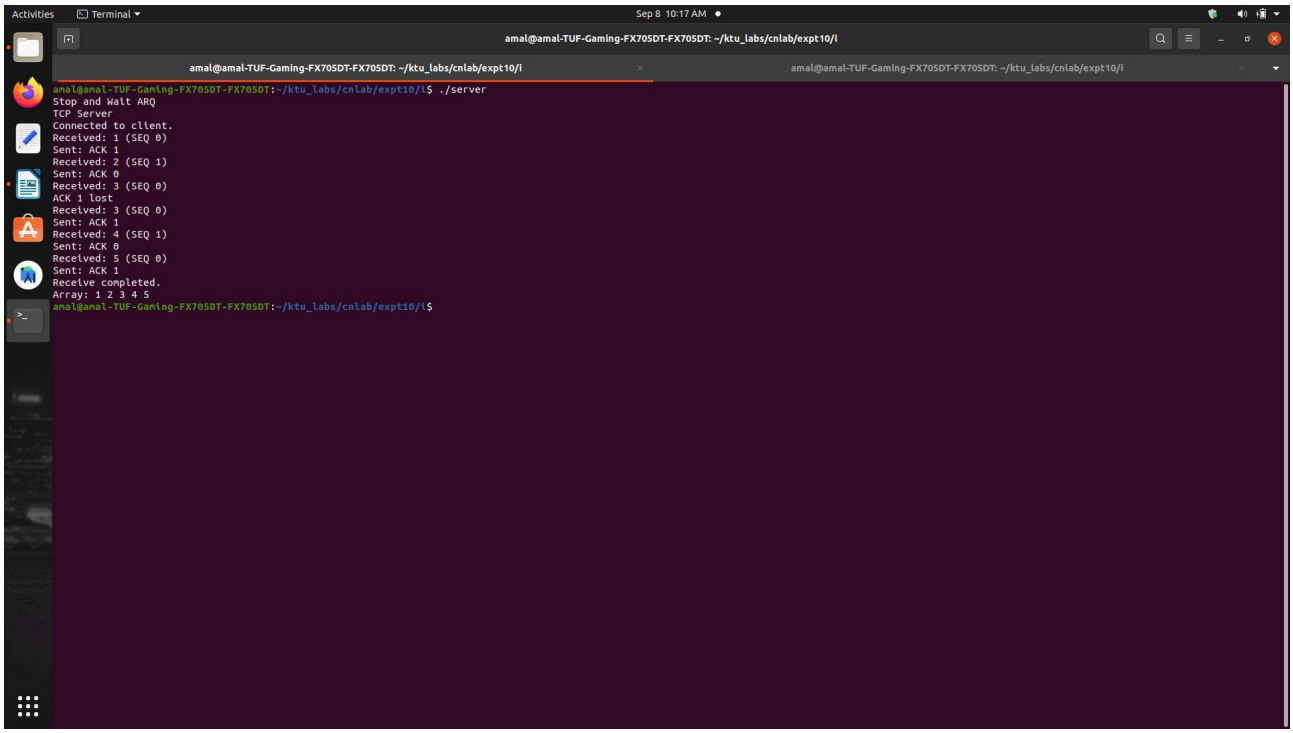
```
#include <stdio.h>
#include <stdlib.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <pthread.h>

#define PORT 8000

typedef struct packet {
      int data;
      int type; // SEQ (0) or ACK (1)
      int seq; // Sequence number (0 or 1)
} packet;

typedef struct data {
      int* arr;
      int* i;
      int client_fd;
      packet* p;
```

```
} data;

void* client(void* arg) {
	data d = *((data*) arg);

	d.p->type = 0;
	d.p->data = d.arr[*d.i];

	if(rand() % 5 != 2) {
		if(send(d.client_fd, d.p, sizeof(packet), 0) < 0) {
			printf("Send failed!\n");
		} else {
			printf("Sent: %d (SEQ %d)\n", d.p->data, d.p->seq);

			if(recv(d.client_fd, d.p, sizeof(packet), 0) < 0) {
				printf("Receive failed!\n");
			} else {
				printf("Received: ACK %d\n", d.p->seq);

				d.arr[*d.i] = -1;

				*(d.i) = *(d.i) + 1;
			}
		}
	} else {
		printf("SEQ %d lost\n", d.p->seq);
	}
}

void* timeout(void* t) {
	sleep(1);
	pthread_t tid = *((pthread_t*) t);
	pthread_cancel(tid);
}

void main() {
	int client_fd;
	struct sockaddr_in serv_addr;

	printf("TCP Client\n");

	client_fd = socket(AF_INET, SOCK_STREAM, 0);

	if(client_fd < 0) {
		printf("Socket creation failed!\n");
		exit(1);
	}

	serv_addr.sin_family = AF_INET;
	serv_addr.sin_addr.s_addr = INADDR_ANY;
	serv_addr.sin_port = htons(PORT);

	if(connect(client_fd, (struct sockaddr*) &serv_addr, sizeof(serv_addr)) <
0) {
		printf("Connection failed!\n");
		exit(1);
	} else {
		printf("Connected to server.\n");
	}

	int n;

	printf("Enter array size: ");
	scanf("%d", &n);
```

```
	int arr[n];

	printf("Enter array elements: ");
	for(int i = 0; i < n; i++) {
		scanf("%d", &arr[i]);
	}

	int i = 0;
	packet p;
	data d;
	d.client_fd = client_fd;
	d.p = &p;
	d.arr = arr;
	d.i = &i;
	p.seq = 0;
	pthread_t tid1, tid2;

	while(1) {
		if(i == n) {
			printf("Send completed.\nArray: ");

			for(int j = 0; j < n; j++) {
				printf("%d ", arr[j]);
			}

			printf("\n");
			break;
		}
		pthread_create(&tid1, NULL, client, &d);
		pthread_create(&tid2, NULL, timeout, &tid1);
		pthread_join(tid1, NULL);
		pthread_join(tid2, NULL);
	}

	close(client_fd);
}
```

OUTPUT


Activities
Terminal
Sep 8 10:17 AM
amal@amal-TUF-Gaming-FX705DT-FX705DT: ~/ktu_labs/cnlab/expt10/i
amal@amal-TUF-Gaming-FX705DT-FX705DT: ~/ktu_labs/cnlab/expt10/i
amal@amal-TUF-Gaming-FX705DT-FX705DT: ~/ktu_labs/cnlab/expt10/i
amal@amal-TUF-Gaming-FX705DT-FX705DT:~/ktu_labs/cnlab/expt10/i$ ./client
TCP Client
Connected to server.
Enter array size: 5
Enter array elements: 1 2 3 4 5
Sent: 1 (SEQ 0)
Received: ACK 1
Sent: 2 (SEQ 1)
Received: ACK 0
SEQ 0 lost
Sent: 3 (SEQ 0)
Sent: 3 (SEQ 0)
Received: ACK 1
Sent: 4 (SEQ 1)
Received: ACK 0
Sent: 5 (SEQ 0)
Received: ACK 1
Send completed.
Array: -1 -1 -1 -1 -1
amal@amal-TUF-Gaming-FX705DT-FX705DT:~/ktu_labs/cnlab/expt10/i$


Activities
Terminal
Sep 8 10:17 AM
amal@amal-TUF-Gaming-FX705DT-FX705DT: ~/ktu_labs/cnlab/expt10/i
amal@amal-TUF-Gaming-FX705DT-FX705DT: ~/ktu_labs/cnlab/expt10/i
amal@amal-TUF-Gaming-FX705DT-FX705DT: ~/ktu_labs/cnlab/expt10/i
amal@amal-TUF-Gaming-FX705DT-FX705DT:~/ktu_labs/cnlab/expt10/i$ ./server
Stop and Wait ARQ
TCP Server
Connected to client.
Received: 1 (SEQ 0)
Sent: ACK 1
Received: 2 (SEQ 1)
Sent: ACK 0
Received: 3 (SEQ 0)
ACK 1 lost
Received: 3 (SEQ 0)
Sent: ACK 1
Received: 4 (SEQ 1)
Sent: ACK 0
Received: 5 (SEQ 0)
Sent: ACK 1
Receive completed.
Array: 1 2 3 4 5
amal@amal-TUF-Gaming-FX705DT-FX705DT:~/ktu_labs/cnlab/expt10/i$