DCCN LAB 5

NAME: **MALOTH ADITYA**
ROLL NO.: **120CS0124**

Q.1

Client Code:

```
// Client side implementation of UDP client-server model
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#define SA struct sockaddr

int main(){
    int sid;
    char c='Y';
    struct sockaddr_in server_address;
    int ser_len;
    server_address.sin_family = AF_INET;
    server_address.sin_addr.s_addr = inet_addr("127.0.0.1");
    server_address.sin_port = 6969;
    ser_len = sizeof(server_address);
    printf("Character sent: %c\nSeeking a signal from server\n\n",c);
    sid=socket(AF_INET,SOCK_DGRAM,0);
    sendto(sid,&c,1,0,(SA *)&server_address,ser_len);
    recvfrom(sid,&c,1,0,(SA *)&server_address,&ser_len);
    printf("Character received from server: %c\n\n",c);
    close(sid);
    return 0;
}
```

Server Code:

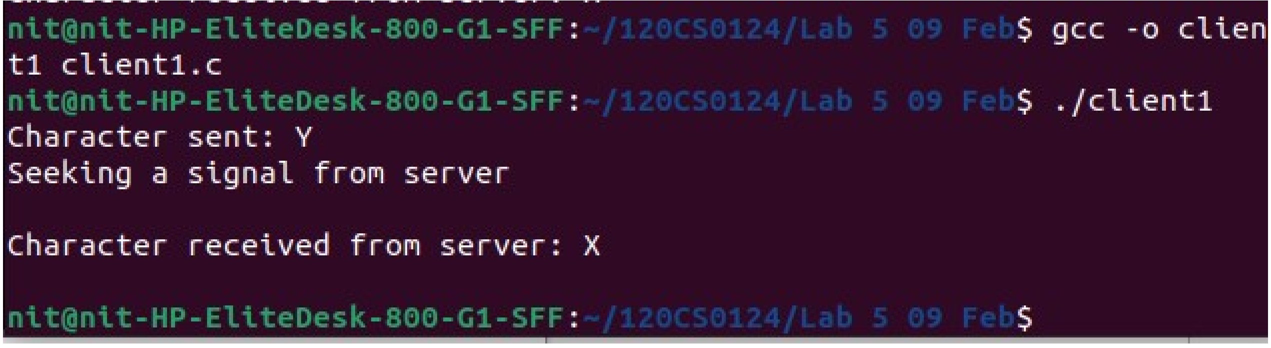
```
// Server side implementation of UDP client-server model
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#define SA struct sockaddr

int main(){
    int sockfd;
```

DCCN LAB 5

```
    char c;
    struct sockaddr_in servaddr, cliaddr;

    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    servaddr.sin_port = 6969;
    int ser_len = sizeof(servaddr);
    int cli_len = sizeof(cliaddr);
    sockfd = socket(AF_INET,SOCK_DGRAM,0);
    bind(sockfd,(SA *)&servaddr,ser_len);

    while(1){
        printf("----------------------------\n");
        printf("Ready to receive datagram :)\n");
        recvfrom(sockfd,&c,1,0,(SA *)&cliaddr,&cli_len);
        printf("Received %c\n",c);
        c='X';
        sendto(sockfd,&c,1,0,(SA *)&cliaddr,cli_len);
        printf("____________________________\n\n");
    }
    close(sockfd);
    return 0;
}
```

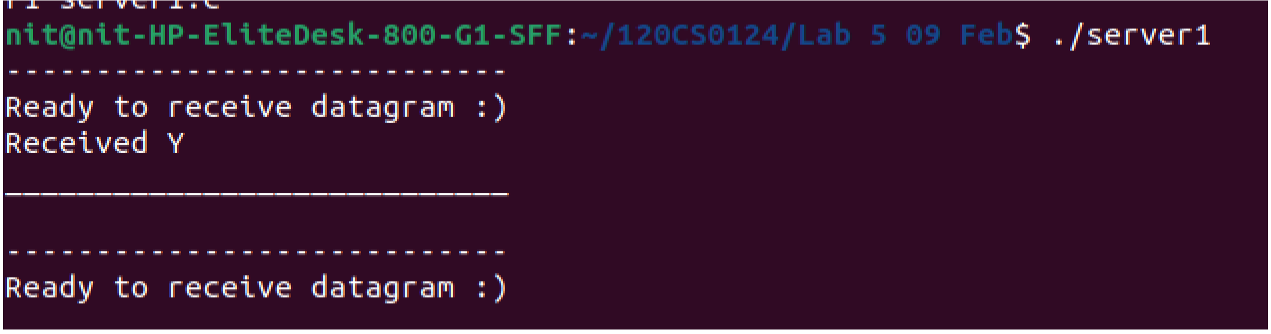
Output:

Client Output:



```
nit@nit-HP-EliteDesk-800-G1-SFF:~/120CS0124/Lab 5 09 Feb$ gcc -o clien
t1 client1.c
nit@nit-HP-EliteDesk-800-G1-SFF:~/120CS0124/Lab 5 09 Feb$ ./client1
Character sent: Y
Seeking a signal from server

Character received from server: X

nit@nit-HP-EliteDesk-800-G1-SFF:~/120CS0124/Lab 5 09 Feb$
```

Server Output:



```
nit@nit-HP-EliteDesk-800-G1-SFF:~/120CS0124/Lab 5 09 Feb$ ./server1
----------------------------
Ready to receive datagram :)
Received Y
____________________________

----------------------------
Ready to receive datagram :)
```

DCCN LAB 5

Q.2

Client Code:

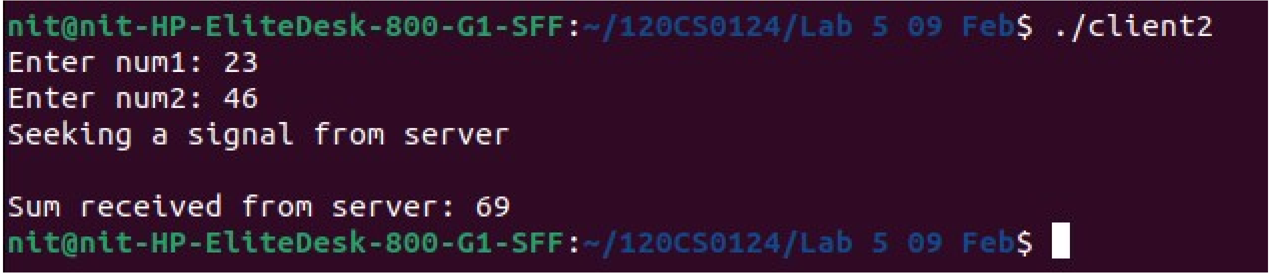
```
// Client side implementation of UDP client-server model
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#define SA struct sockaddr

int main(){
	int sid,num1,num2,res;
	printf("Enter num1: ");
	scanf("%d",&num1);
	printf("Enter num2: ");
	scanf("%d",&num2);
	struct sockaddr_in server_address;
	int ser_len;
	server_address.sin_family = AF_INET;
	server_address.sin_addr.s_addr = inet_addr("127.0.0.1");
	server_address.sin_port = 6969;
	ser_len = sizeof(server_address);
	printf("Seeking a signal from server\n\n");
	sid=socket(AF_INET,SOCK_DGRAM,0);
	sendto(sid,&num1,sizeof(num1),0,(SA *)&server_address,ser_len);
	sendto(sid,&num2,sizeof(num2),0,(SA *)&server_address,ser_len);
	recvfrom(sid,&res,sizeof(res),0,(SA *)&server_address,&ser_len);
	printf("Sum received from server: %d\n",res);
	close(sid);
	return 0;
}
```

Server Code:

```
// Server side implementation of UDP client-server model
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#define SA struct sockaddr

int main(){
	int sockfd,res,num;
	//const char hello = "Hello from server";
	struct sockaddr_in servaddr, cliaddr;

	servaddr.sin_family = AF_INET;
	servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
	servaddr.sin_port = 6969;
	int ser_len = sizeof(servaddr);
	int cli_len = sizeof(cliaddr);
	sockfd = socket(AF_INET,SOCK_DGRAM,0);
	bind(sockfd,(SA *)&servaddr,ser_len);

	while(1){
		printf("____________________________\n");
		printf("Ready to receive datagram :)\n");
		recvfrom(sockfd,&res,sizeof(res),0,(SA *)&cliaddr,&cli_len);
		recvfrom(sockfd,&num,sizeof(num),0,(SA *)&cliaddr,&cli_len);
		printf("Received num1: %d\tnum2: %d\n",res,num);
		res=res+num;
		sendto(sockfd,&res,sizeof(res),0,(SA *)&cliaddr,cli_len);
		printf("----------------------------\n\n");
	}
	close(sockfd);
	return 0;
}
```

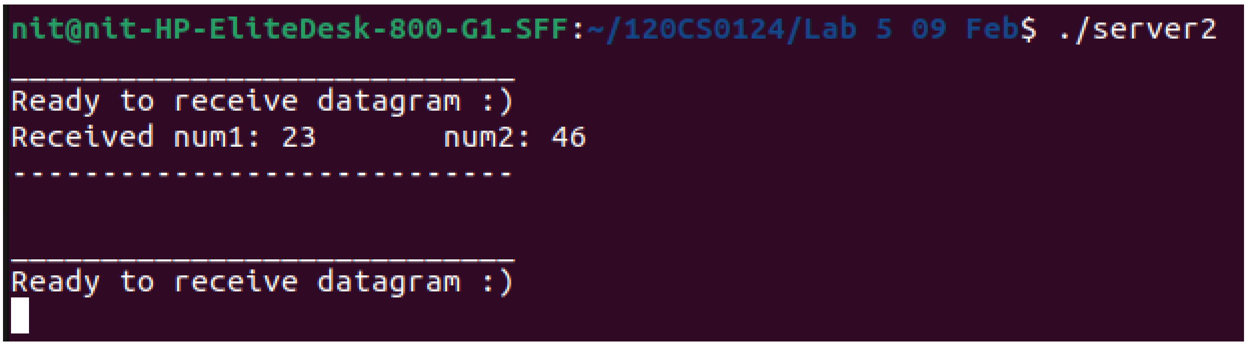
Output:

Client Output:



```
nit@nit-HP-EliteDesk-800-G1-SFF:~/120CS0124/Lab 5 09 Feb$ ./client2
Enter num1: 23
Enter num2: 46
Seeking a signal from server

Sum received from server: 69
nit@nit-HP-EliteDesk-800-G1-SFF:~/120CS0124/Lab 5 09 Feb$ 
```

Server Output:



```
nit@nit-HP-EliteDesk-800-G1-SFF:~/120CS0124/Lab 5 09 Feb$ ./server2
____________________________
Ready to receive datagram :)
Received num1: 23       num2: 46
----------------------------


____________________________
Ready to receive datagram :)
```

DCCN LAB 5

Q.3

Client Code in UDP:

```
// Client side implementation of UDP client-server model
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#define SA struct sockaddr

void prime(int num){
        if(num) printf("prime number\n");
        else printf("not a prime number\n");
}

int main(){
        int sid,num1,num2,res;
        printf("Enter num1: ");
        scanf("%d",&num1);
        printf("Enter num2: ");
        scanf("%d",&num2);
        struct sockaddr_in server_address;
        int ser_len;
        server_address.sin_family = AF_INET;
        server_address.sin_addr.s_addr = inet_addr("127.0.0.1");
        server_address.sin_port = 6969;
        ser_len = sizeof(server_address);
        printf("\n\nSeeking a signal from server\n");
        sid=socket(AF_INET,SOCK_DGRAM,0);
        sendto(sid,&num1,sizeof(num1),0,(SA *)&server_address,ser_len);
        sendto(sid,&num2,sizeof(num2),0,(SA *)&server_address,ser_len);
        int diff,prod,quo;
        recvfrom(sid,&diff,4,0,(SA *)&server_address,&ser_len);
        recvfrom(sid,&prod,4,0,(SA *)&server_address,&ser_len);
        recvfrom(sid,&quo,4,0,(SA *)&server_address,&ser_len);
        //for(int i=0;i<3;i++) rans[i]=ntohl(ans[i]);
        printf("Heard something from server\n");
        printf("\nDifference: %d\nProduct: %d\nQuotient: %d\n",diff,prod,quo);
        if(quo==-1) printf("Since num2 is 0, couldn't perform division\n");
        printf("-----Checking if a number is prime-----\n");
        printf("%d: ",num1);
        recvfrom(sid,&num1,4,0,(SA *)&server_address,&ser_len);
        prime(num1);
        printf("%d: ",num2);
        recvfrom(sid,&num2,4,0,(SA *)&server_address,&ser_len);
```

```
    prime(num2);
    close(sid);
    return 0;
}
```

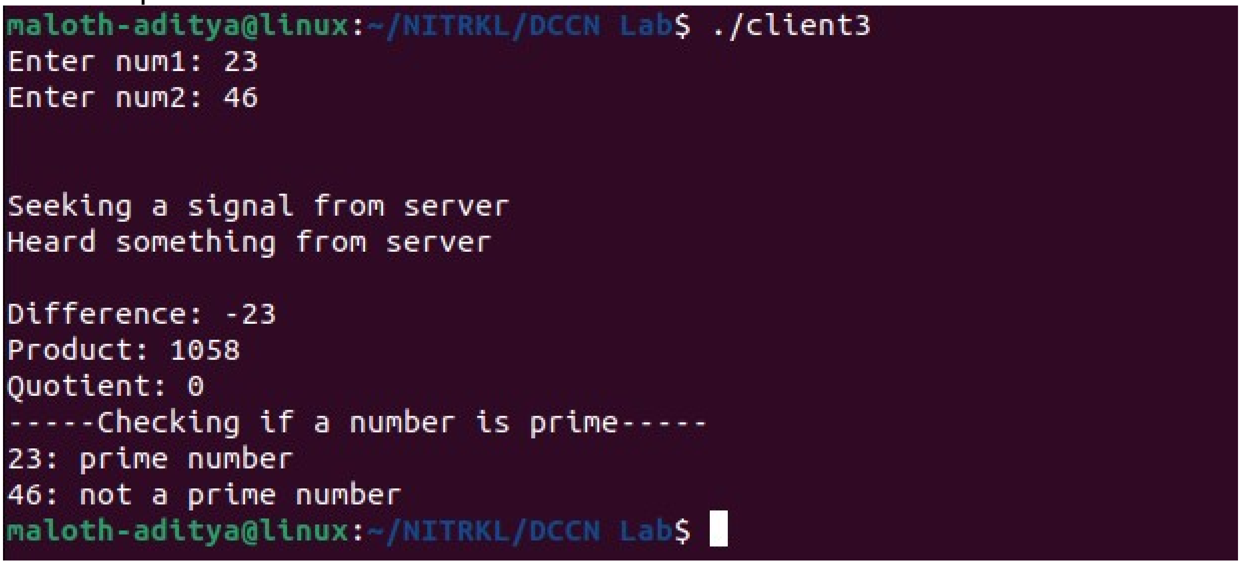
Server Code in UDP:

```
// Server side implementation of UDP client-server model
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
//#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#define SA struct sockaddr

#define PORT 6969
#define MAXLINE 1024

int prime(int num){
    for(int i=2;i*i<=num;i++){
        if(num%i==0){
            printf("Not a prime number\n");
            return 0;
        }
    }
    printf("Prime number\n");
    return 1;
}

int main(){
    int sockfd,res,num;
    struct sockaddr_in servaddr, cliaddr;

    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    servaddr.sin_port = 6969;
    int ser_len = sizeof(servaddr);
    int cli_len = sizeof(cliaddr);
    sockfd = socket(AF_INET,SOCK_DGRAM,0);
    bind(sockfd,(SA *)&servaddr,ser_len);
    int diff,prod,quo;
    while(1){
        printf("______________________________\n");
        printf("Ready to receive datagram :)\n");
        recvfrom(sockfd,&res,sizeof(res),0,(SA *)&cliaddr,&cli_len);
        recvfrom(sockfd,&num,sizeof(num),0,(SA *)&cliaddr,&cli_len);
```

```
		printf("Received num1: %d\tnum2: %d\n",res,num);
		diff = res-num;
		prod = res*num;
		//for(int i=0;i<3;i++) sans[i]=htonl(ans[i]);
		if(num==0) quo=-1;
		else quo = res/num;
		printf("\nDifference: %d\nProduct: %d\nQuotient: %d\
n",diff,prod,quo);
		printf("%d: ",res);
		res=prime(res);
		printf("%d: ",num);
		num=prime(num);
		sendto(sockfd,&diff,4,0,(SA *)&cliaddr,cli_len);
		sendto(sockfd,&prod,4,0,(SA *)&cliaddr,cli_len);
		sendto(sockfd,&quo,4,0,(SA *)&cliaddr,cli_len);
		sendto(sockfd,&res,4,0,(SA *)&cliaddr,cli_len);
		sendto(sockfd,&num,4,0,(SA *)&cliaddr,cli_len);
		printf("----------------------------\n\n");
	}
	close(sockfd);
	return 0;
}
```

Output:

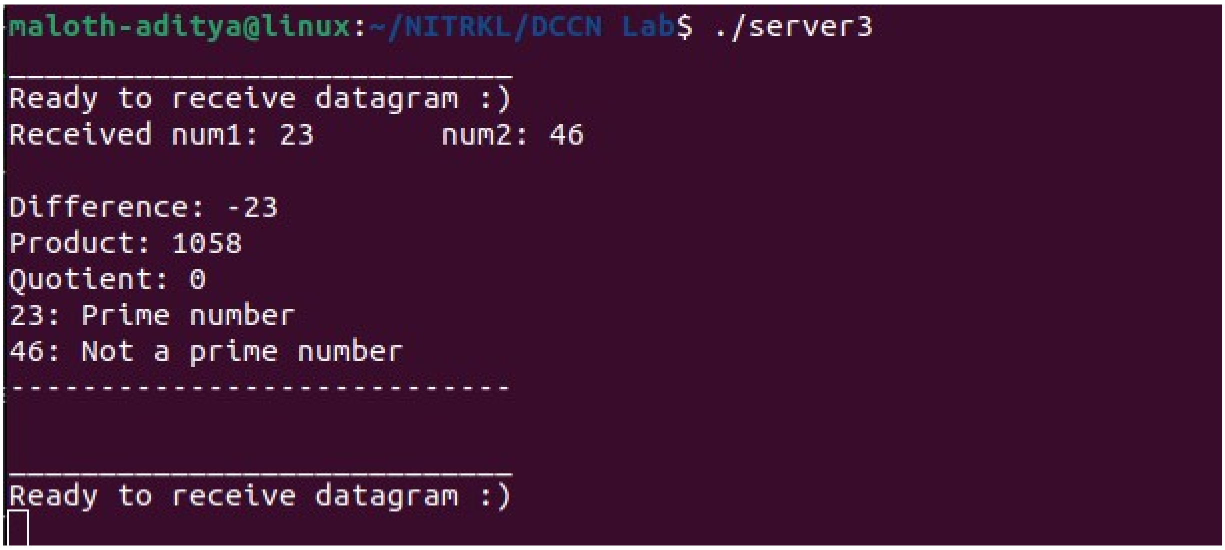
Client Output:



```
maloth-aditya@linux:~/NITRKL/DCCN Lab$ ./client3
Enter num1: 23
Enter num2: 46


Seeking a signal from server
Heard something from server

Difference: -23
Product: 1058
Quotient: 0
-----Checking if a number is prime-----
23: prime number
46: not a prime number
maloth-aditya@linux:~/NITRKL/DCCN Lab$ 
```

Server Output:

DCCN LAB 5



```
maloth-aditya@linux:~/NITRKL/DCCN Lab$ ./server3
____________________________
Ready to receive datagram :)
Received num1: 23       num2: 46

Difference: -23
Product: 1058
Quotient: 0
23: Prime number
46: Not a prime number
----------------------------


____________________________
Ready to receive datagram :)
```

Q.3

Client Code in TCP:

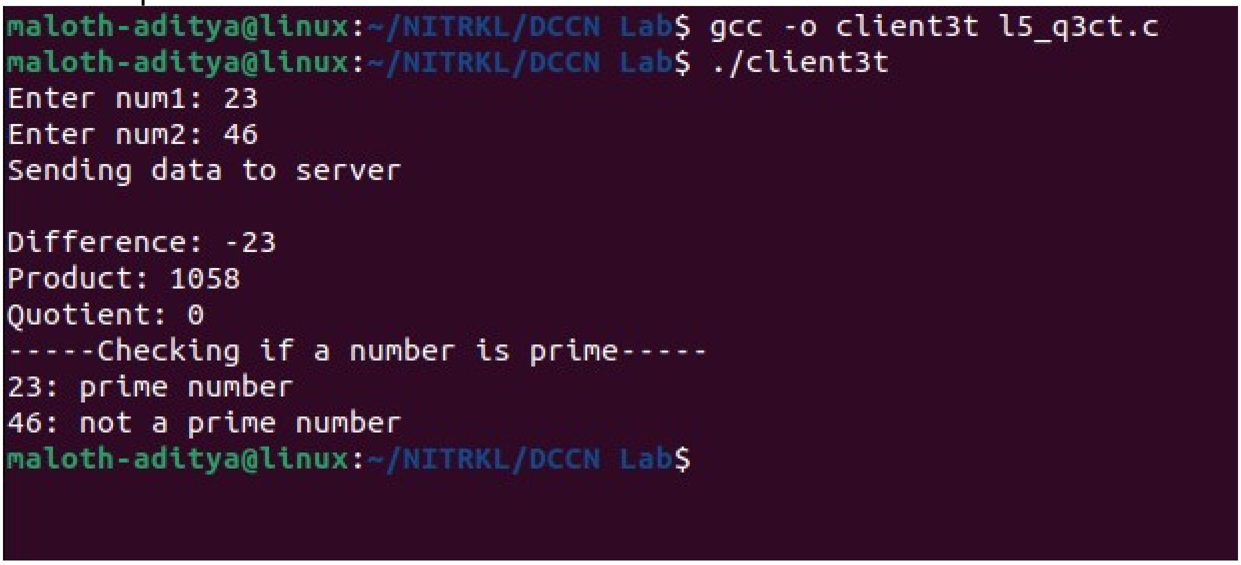
```
// Client side implementation of TCP client-server model
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <arpa/inet.h>

void prime(int num){
        if(num) printf("prime number\n");
        else printf("not a prime number\n");
}

int main(){
    int sid,num1,num2;
    printf("Enter num1: ");
    scanf("%d",&num1);
    printf("Enter num2: ");
    scanf("%d",&num2);
    struct sockaddr_in server_address;
    int server_addlen;
    server_address.sin_family=AF_INET;
    server_address.sin_addr.s_addr=inet_addr("127.0.0.1");
    server_address.sin_port=5080;
    server_addlen=sizeof(server_address);
    sid=socket(AF_INET,SOCK_STREAM,0);
    connect(sid,(struct sockaddr *)&server_address,server_addlen);
    printf("Sending data to server\n");
    write(sid,&num1,sizeof(int));
    write(sid,&num2,sizeof(int));

    int diff,prod,quo,ans[3];
    read(sid,&diff,sizeof(int));
    read(sid,&prod,sizeof(int));
    read(sid,&quo,sizeof(int));
    printf("\nDifference: %d\nProduct: %d\nQuotient: %d\n",diff,prod,quo);
    if(quo==-1) printf("Since num2 is 0, couldn't perform division\n");
    printf("-----Checking if a number is prime-----\n");
        printf("%d: ",num1);
        read(sid,&num1,sizeof(int));
        prime(num1);
        printf("%d: ",num2);
        read(sid,&num2,sizeof(int));
        prime(num2);
    close(sid);
    return(0);
}
```

Server Code in TCP:

```
// Server side implementation of TCP client-server model
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <arpa/inet.h>

int prime(int num){
        for(int i=2;i*i<=num;i++){
                if(num%i==0){
                        printf("Not a prime number\n");
                        return 0;
                }
        }
        printf("Prime number\n");
        return 1;
}

int main(){
    int serid,sessid;
    int res,num;
    struct sockaddr_in server_address,client_address;
    int server_addlen,client_addlen;
    server_address.sin_family=AF_INET;
    server_address.sin_addr.s_addr=inet_addr("127.0.0.1");
    server_address.sin_port=5080;
    server_addlen=sizeof(server_address);
    client_addlen=sizeof(client_addlen);
    serid=socket(AF_INET,SOCK_STREAM,0);
    bind(serid,(struct sockaddr*)&server_address,server_addlen);
    listen(serid,10);
    while(1){
        printf("________________________________\n");
       printf("Server is ready to accept ......\n");
       sessid=accept(serid,(struct sockaddr *)&client_address,&client_addlen);
       read(sessid,&res,sizeof(int));
       read(sessid,&num,sizeof(int));
       printf("Received num1: %d\tnum2: %d\n",res,num);
       int prod,diff,quo;
       diff = res-num;
        prod = res*num;
        if(num==0) quo=-1;
        else quo = res/num;
        printf("\nDifference: %d\nProduct: %d\nQuotient: %d\n",diff,prod,quo);
        printf("%d: ",res);
        res=prime(res);
        printf("%d: ",num);
        num=prime(num);
        write(sessid,&diff,sizeof(int));
        write(sessid,&prod,sizeof(int));
        write(sessid,&quo,sizeof(int));
        write(sessid,&res,sizeof(int));
        write(sessid,&num,sizeof(int));
        printf("----------------------------------\n\n");
       close(sessid);
    }
    return(0);
}
```

Output:

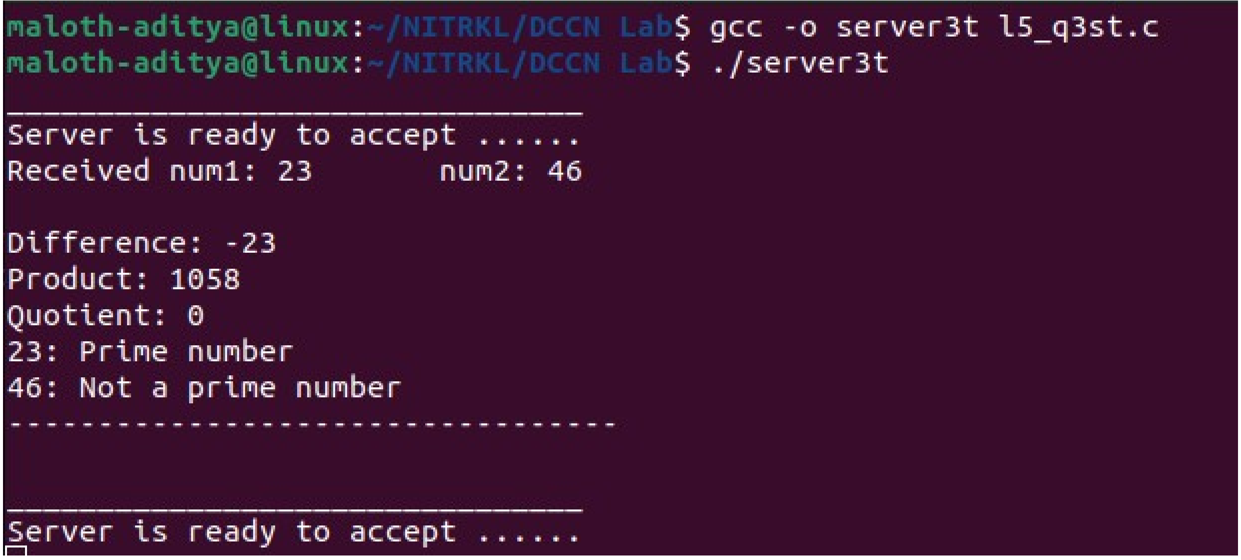
Client Output:



```
maloth-aditya@linux:~/NITRKL/DCCN Lab$ gcc -o client3t l5_q3ct.c
maloth-aditya@linux:~/NITRKL/DCCN Lab$ ./client3t
Enter num1: 23
Enter num2: 46
Sending data to server

Difference: -23
Product: 1058
Quotient: 0
-----Checking if a number is prime-----
23: prime number
46: not a prime number
maloth-aditya@linux:~/NITRKL/DCCN Lab$
```

DCCN LAB 5

Server Output:



```
maloth-aditya@linux:~/NITRKL/DCCN Lab$ gcc -o server3t l5_q3st.c
maloth-aditya@linux:~/NITRKL/DCCN Lab$ ./server3t

________________________________
Server is ready to accept ......
Received num1: 23       num2: 46

Difference: -23
Product: 1058
Quotient: 0
23: Prime number
46: Not a prime number
----------------------------------


________________________________
Server is ready to accept ......
```