DCCN LABORATORY
ASSIGNMENT NO: 5
NAME: ABINASH GUPTA
ROLL: 120CS0157

Q1 .Run the program given in UDP_Socket document

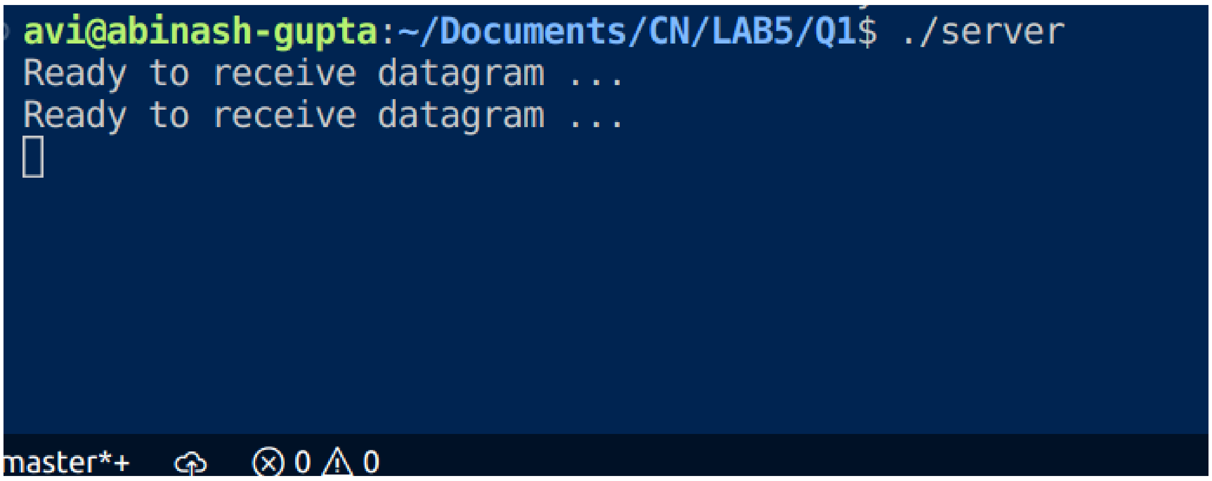
Server.c

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>


int main(){
   int sid;char c;
   struct sockaddr_in server_address;
   struct sockaddr_in client_address;
   int server_addlen,cli_addlen;
   server_address.sin_family=AF_INET;
   server_address.sin_addr.s_addr=inet_addr("127.0.0.1");
   server_address.sin_port=7890;

   server_addlen=sizeof(server_address);
   cli_addlen=sizeof(client_address);

   sid=socket(AF_INET,SOCK_DGRAM,0);
   bind(sid,(struct sockaddr *)&server_address,server_addlen);

   while(1){
       printf("Ready to receive datagram ...\n");
       recvfrom(sid,&c,1,0,(struct sockaddr
*)&client_address,&cli_addlen);
      sendto(sid,"A",1,0,(struct sockaddr
*)&client_address,cli_addlen);


   }

   return 0;
}
```

Client.c

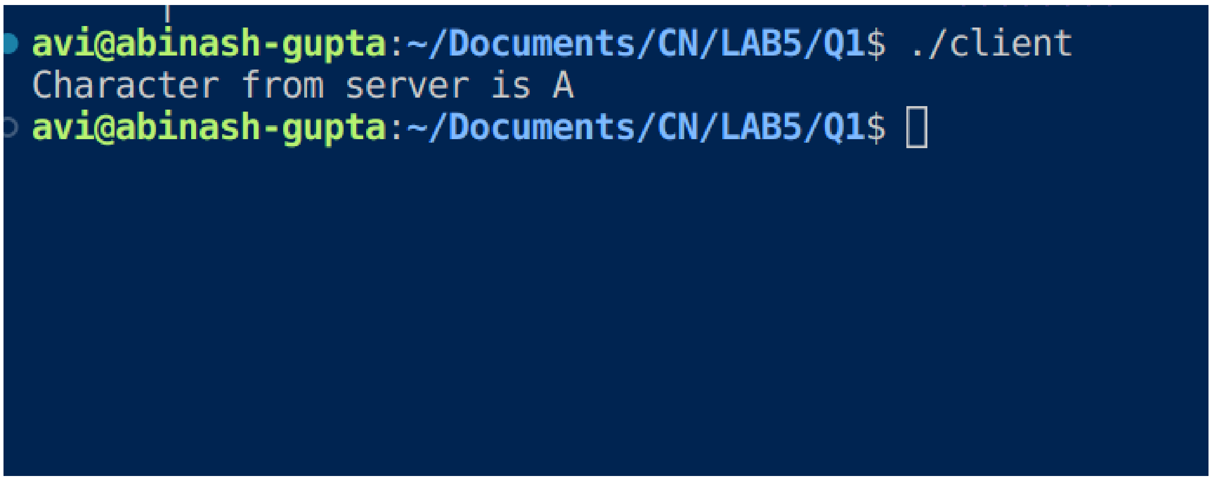
```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>


int main(){
   int sid;char c;
   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=7890;

   server_addlen=sizeof(server_address);
   sid=socket(AF_INET,SOCK_DGRAM,0);
   sendto(sid,"A",1,0,(struct sockaddr
*)&server_address,server_addlen);
   recvfrom(sid,&c,1,0,(struct sockaddr
*)&server_address,&server_addlen);
   printf("Character from server is %c\n",c);
   return 0;



}
```

Output :-



```
avi@abinash-gupta:~/Documents/CN/LAB5/Q1$ ./server
Ready to receive datagram ...
Ready to receive datagram ...
```

master*+ ⓧ 0 ⚠ 0



```
avi@abinash-gupta:~/Documents/CN/LAB5/Q1$ ./client
Character from server is A
avi@abinash-gupta:~/Documents/CN/LAB5/Q1$
```

Q2 .Execute a client/server program using UDP service for adding a two integer numbers requested by the client and evaluated at server and get back result at the client .

Server.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>


#define PORT 12345
#define MAX_LEN 1024


int main(int argc, char *argv[]) {
  int sockfd;
```

```
struct sockaddr_in servaddr, cliaddr;
int n, len;
int num1, num2, result;


// Creating socket file descriptor
sockfd = socket(AF_INET, SOCK_DGRAM, 0);
if (sockfd < 0) {
    perror("socket creation failed");
    exit(EXIT_FAILURE);
}


memset(&servaddr, 0, sizeof(servaddr));
memset(&cliaddr, 0, sizeof(cliaddr));


// Filling server information
servaddr.sin_family = AF_INET;
servaddr.sin_port = htons(PORT);
servaddr.sin_addr.s_addr = htonl(INADDR_ANY);


// Bind the socket with the server address
if (bind(sockfd, (const struct sockaddr *)&servaddr,
         sizeof(servaddr)) < 0) {
    perror("bind failed");
    exit(EXIT_FAILURE);
}


len = sizeof(cliaddr);


while (1) {
 printf("Server is waiting to add ...\n");
    n = recvfrom(sockfd, (int *)&num1, sizeof(int),
                 0, (struct sockaddr *)&cliaddr,
                 &len);
    if (n < 0) {
        perror("Error receiving num1");
        continue;
    }
```

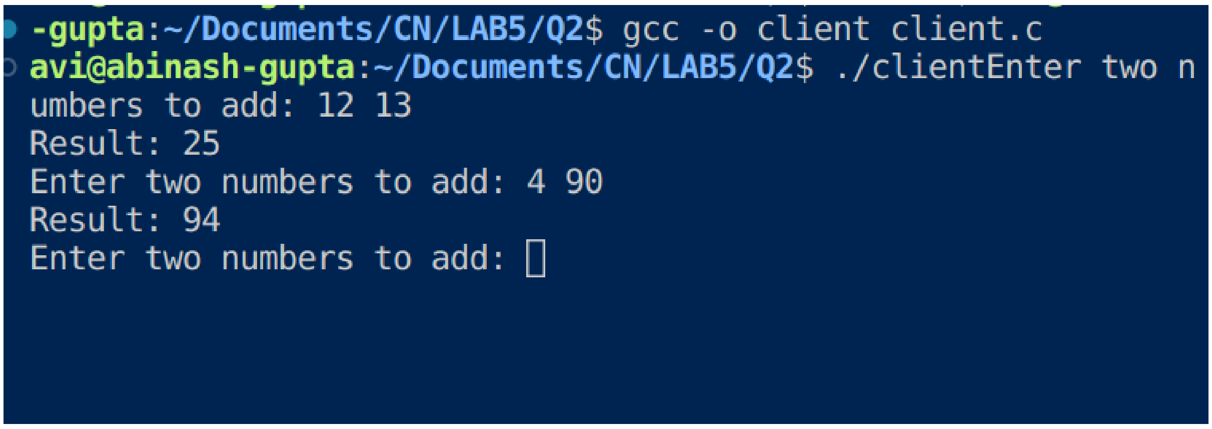
```
      n = recvfrom(sockfd, (int *)&num2, sizeof(int),
                   0, (struct sockaddr *)&cliaddr,
                   &len);
      if (n < 0) {
          perror("Error receiving num2");
          continue;
      }


      result = num1 + num2;


      sendto(sockfd, (const int *)&result, sizeof(int),
             0, (const struct sockaddr *)&cliaddr, len);
  }


  return 0;
}
```

Client.c

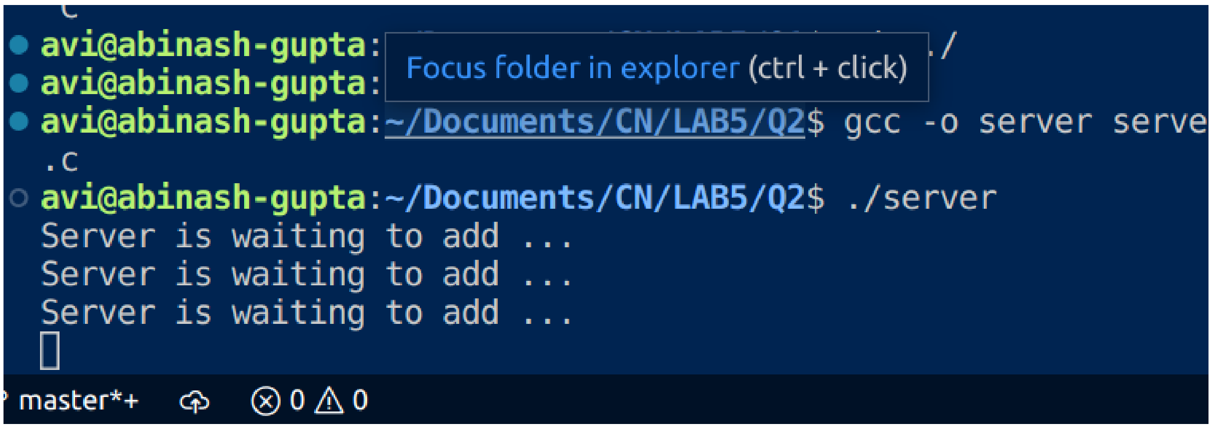
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>


#define PORT 12345
#define MAX_LEN 1024


int main(int argc, char *argv[]) {
  int sockfd;
  struct sockaddr_in servaddr;
  int n, len;
  int num1, num2, result;
```

```c
    // Creating socket file descriptor
    sockfd = socket(AF_INET, SOCK_DGRAM, 0);
    if (sockfd < 0) {
        perror("socket creation failed");
        exit(EXIT_FAILURE);
    }


    memset(&servaddr, 0, sizeof(servaddr));


    // Filling server information
    servaddr.sin_family = AF_INET;
    servaddr.sin_port = htons(PORT);
    servaddr.sin_addr.s_addr = INADDR_ANY;


    while (1) {
        printf("Enter two numbers to add: ");
        scanf("%d%d", &num1, &num2);


        sendto(sockfd, (const int *)&num1, sizeof(int),
               0, (const struct sockaddr *)&servaddr,
               sizeof(servaddr));
        sendto(sockfd, (const int *)&num2, sizeof(int),
               0, (const struct sockaddr *)&servaddr,
               sizeof(servaddr));


        n = recvfrom(sockfd, (int *)&result, sizeof(int),
                     0, NULL, NULL);

        if (n == sizeof(int)) {
            printf("Result: %d\n", result);
        } else {
            printf("Error receiving result\n");
        }
    }


    return 0;
}
```

Output :-



```
-gupta:~/Documents/CN/LAB5/Q2$ gcc -o client client.c
avi@abinash-gupta:~/Documents/CN/LAB5/Q2$ ./clientEnter two n
umbers to add: 12 13
Result: 25
Enter two numbers to add: 4 90
Result: 94
Enter two numbers to add: 
```



```
avi@abinash-gupta:
avi@abinash-gupta:
avi@abinash-gupta:~/Documents/CN/LAB5/Q2$ gcc -o server serve
.c
avi@abinash-gupta:~/Documents/CN/LAB5/Q2$ ./server
Server is waiting to add ...
Server is waiting to add ...
Server is waiting to add ...
```

Q3 .Execute a client/server program for simple calculator having following operations addition, subtraction, multiplication, division, and detecting prime numbers. Input entered at the client side and evaluated at server machine and get back result at the client machine.Try this question for both TCP and UDP socket programming.

UDP Connection

Server.c

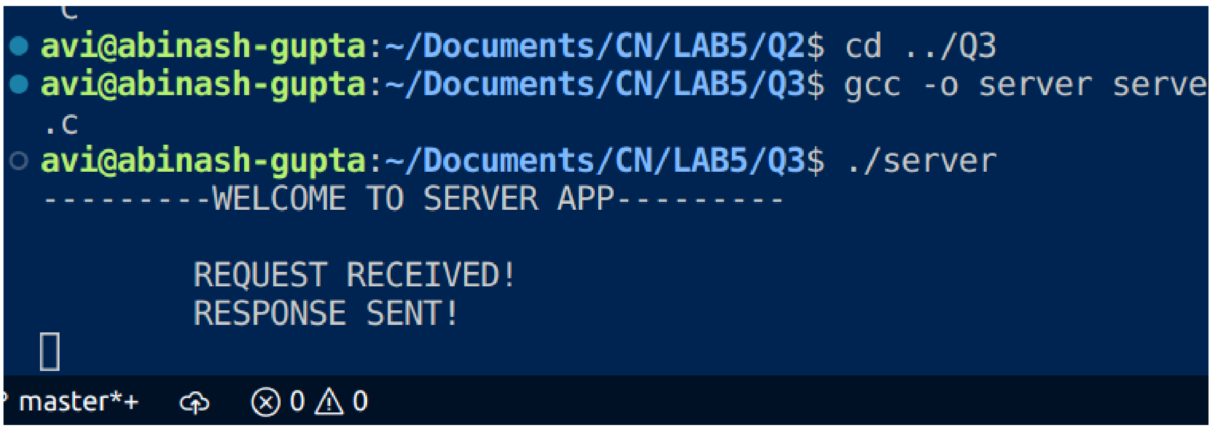
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
int main()
{
int server_fd, client_fd;
struct sockaddr_in server_addr, client_addr;
int buffer[3], result;
int i, tmp, len;
```

```
// socket creation
server_fd = socket(AF_INET, SOCK_DGRAM, 0);
if(server_fd < 0) return 1;

// binding
bzero(&server_addr, sizeof(server_addr));
bzero(&client_addr, sizeof(client_addr));

server_addr.sin_family = AF_INET;
server_addr.sin_port = htons(8080); // convert int to network bits
server_addr.sin_addr.s_addr = inet_addr("127.0.0.1"); // local ip

tmp = bind(server_fd, (struct sockaddr*) &server_addr,
sizeof(server_addr));
if(tmp < 0) return 2;

printf("---------WELCOME TO SERVER APP---------\n\n");

while(1)
{
len = sizeof(client_addr);

// receive an array of two numbers
recvfrom(server_fd, buffer, sizeof(buffer), 0, (struct sockaddr*)
&client_addr, &len);
printf("\tREQUEST RECEIVED!\n");

// carry out operation
switch(buffer[0]) {
case 0:
result = buffer[1] + buffer[2];
break;
case 1:
result = buffer[1] - buffer[2];
break;
case 2:
result = buffer[1] * buffer[2];
break;
case 3:
result = buffer[1] / buffer[2];
break;
case 4:
result = 1;
for(i = 2; i < buffer[1]; i++) {
if(buffer[1] % i == 0) {
result = 0;
}
}
break;
default:
printf("Invalid choice!\n");
exit(1);
}
// send the result
sendto(server_fd, &result, sizeof(&result), 0, (struct sockaddr*)
&client_addr, sizeof(client_addr));
printf("\tRESPONSE SENT!\n");
}
return 0;
}
```

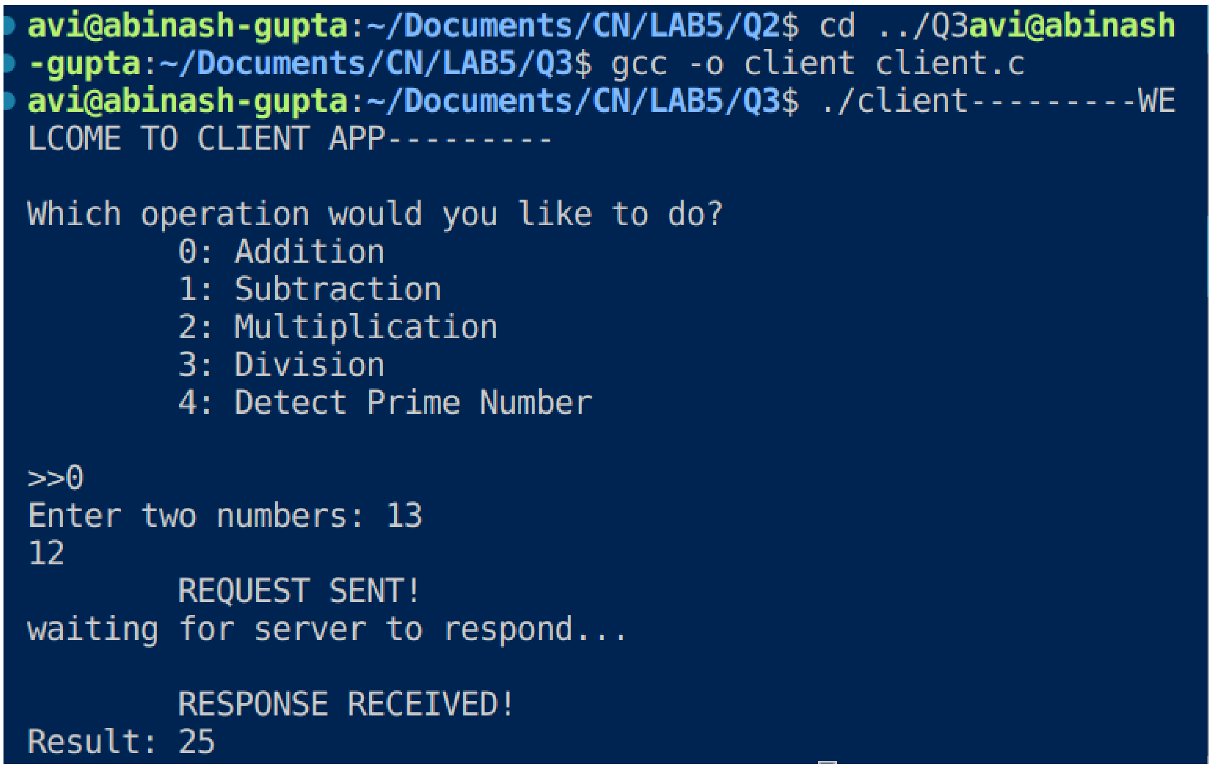
Client.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
int main()
{
int client_fd;
struct sockaddr_in server_addr;
int res = 0;
int len, *result = &res, first, second, choice;
int buffer[3]; // buffer[0] = operation

// socket creation
client_fd = socket(AF_INET, SOCK_DGRAM, 0);

bzero(&server_addr, sizeof(server_addr));
server_addr.sin_family = AF_INET;
server_addr.sin_port = htons(8080); // convert int to network bits
server_addr.sin_addr.s_addr = inet_addr("127.0.0.1"); // local ip
//server_addr.sin_addr.s_addr = inet_addr("192.168.124.61");

len = sizeof(server_addr);
printf("---------WELCOME TO CLIENT APP---------\n\n");

// choice of operation
printf("Which operation would you like to do?\n\t0: Addition\n\t1:
Subtraction\n\t2: Multiplication\n\t3: Division\n\t4: Detect Prime
Number\n\n>>");
scanf("%d", &choice);
buffer[0] = choice;

if(choice == 4) {
printf("Enter a number: ");
scanf("%d", &first);
second = 0;
} else if (choice >= 0 || choice <= 3) {
printf("Enter two numbers: ");
scanf("%d %d", &first, &second);
} else {
printf("INVALID CHOICE!\n");
exit(1);
}
// the numbers
buffer[1] = first;
buffer[2] = second;

// send numbers to server
sendto(client_fd, buffer, sizeof(buffer), 0, (struct sockaddr*)
&server_addr, sizeof(server_addr));

printf("\tREQUEST SENT!\n");
printf("waiting for server to respond...\n\n");

recvfrom(client_fd, result, sizeof(result), 0, (struct sockaddr*)
&server_addr, &len);
printf("\tRESPONSE RECEIVED!\n");

printf("Result: %d\n", *result);



return 0;
}
```

OUTPUT :-



```
 avi@abinash-gupta:~/Documents/CN/LAB5/Q2$ cd ../Q3
 avi@abinash-gupta:~/Documents/CN/LAB5/Q3$ gcc -o server serve
.c
 avi@abinash-gupta:~/Documents/CN/LAB5/Q3$ ./server
---------WELCOME TO SERVER APP---------

        REQUEST RECEIVED!
        RESPONSE SENT!

 master*+     ⓧ 0 ⚠ 0
```



```
avi@abinash-gupta:~/Documents/CN/LAB5/Q2$ cd ../Q3avi@abinash
-gupta:~/Documents/CN/LAB5/Q3$ gcc -o client client.c
avi@abinash-gupta:~/Documents/CN/LAB5/Q3$ ./client---------WE
LCOME TO CLIENT APP---------

Which operation would you like to do?
        0: Addition
        1: Subtraction
        2: Multiplication
        3: Division
        4: Detect Prime Number

>>0
Enter two numbers: 13
12
        REQUEST SENT!
waiting for server to respond...

        RESPONSE RECEIVED!
Result: 25
```

(b) TCP Connection

Server.c

```c
#include<sys/types.h>
#include<sys/socket.h>
#include<stdio.h>
#include<netinet/in.h>
#include <unistd.h>
#include<string.h>
#include <arpa/inet.h>
#include<stdbool.h>
bool ifPrime(int a)
```

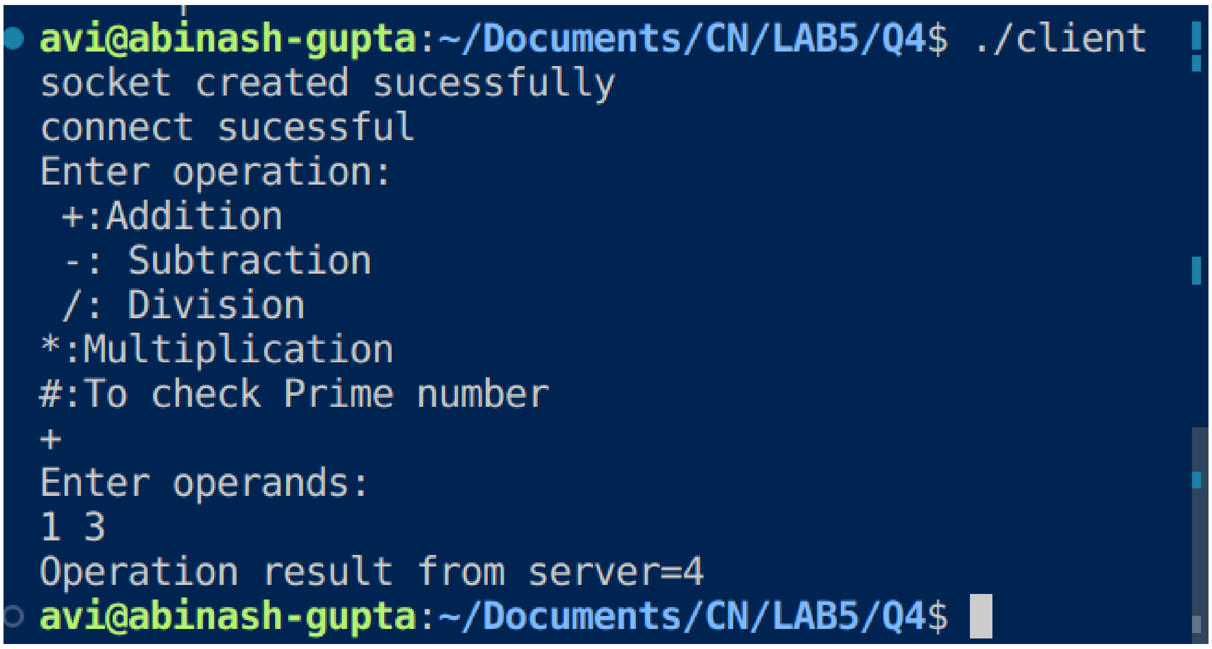
```
{
  int c;
  for ( c = 2 ; c <= a - 1 ; c++ )
  {
     if ( a%c == 0 )
    return false;
  }
  return true;
}

void main()
{
int b,sockfd,connfd,sin_size,l,n,len;
char operator;
int op1,op2,result;
if((sockfd=socket(AF_INET,SOCK_STREAM,0))>0)
printf("socket created sucessfully\n");  //socket creation
//printf("%d\n", sockfd);                 //on success 0 otherwise -1

struct sockaddr_in servaddr;
struct sockaddr_in clientaddr;

servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
servaddr.sin_port=6006;

if((bind(sockfd, (struct sockaddr *)&servaddr,sizeof(servaddr)))==0)
printf("bind sucessful\n");

if((listen(sockfd,5))==0) //listen for connections on a socket
printf("listen sucessful\n");
//printf("%d\n",l);

sin_size = sizeof(struct sockaddr_in);
if((connfd=accept(sockfd,(struct sockaddr *)&clientaddr,&sin_size))>0);
printf("accept sucessful\n");
//printf("%d\n",connfd);
read(connfd, &operator,10);
if(operator=='#') read(connfd,&op1,sizeof(op1));
else{
read(connfd,&op1,sizeof(op1));
read(connfd,&op2,sizeof(op2));
}
switch(operator) {
       case '+': result=op1 + op2;
        printf("Result is: %d + %d = %d\n",op1, op2, result);
        break;
       case '-':result=op1 - op2;
               printf("Result is: %d - %d = %d\n",op1, op2, result);
               break;
       case '*':result=op1 * op2;
               printf("Result is: %d * %d = %d\n",op1, op2, result);
               break;
       case '/':result=op1 / op2;
               printf("Result is: %d / %d = %d\n",op1, op2, result);
               break;
       case '#':result= ifPrime(op1);
           printf("Result is: #%d = %d\n",op1, result);
           break;
       default:
               printf("ERROR: Unsupported Operation");
   }
 write(connfd,&result,sizeof(result));
close(sockfd);
}
```

Client.c

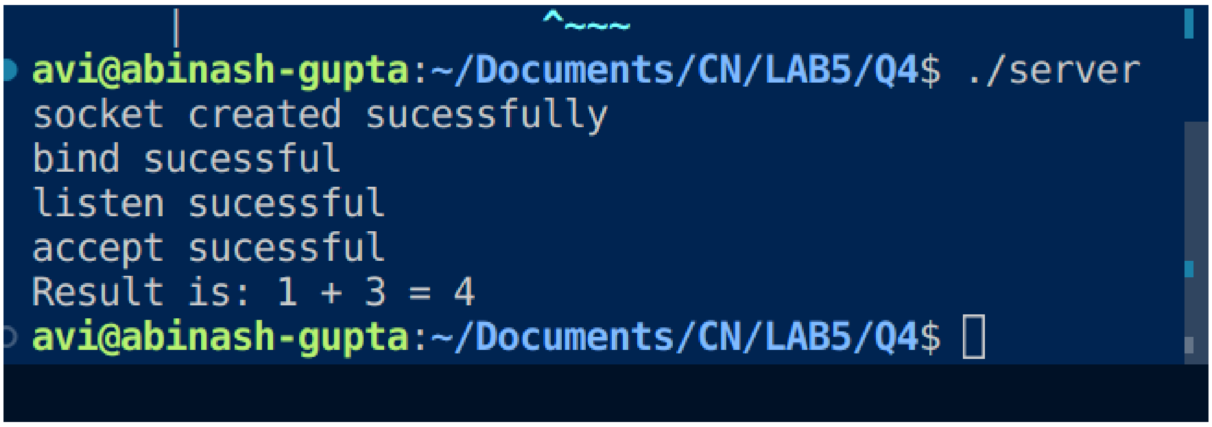
```
#include<sys/types.h>
#include<sys/socket.h>
#include<stdio.h>
#include<netinet/in.h>
#include <unistd.h>
#include<string.h>
#include<strings.h>
#include <arpa/inet.h>

//#define buffsize  150
void main()
{
   int b,sockfd,sin_size,con,n,len;
   //char buff[256];
   char operator;
   int op1,op2,result;
   if((sockfd=socket(AF_INET,SOCK_STREAM,0))>0)
   printf("socket created sucessfully\n");
```

```c
   //printf("%d\n", sockfd);
   struct sockaddr_in servaddr;

   servaddr.sin_family=AF_INET;
   servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
   servaddr.sin_port=6006;

   sin_size = sizeof(struct sockaddr_in);
   if((con=connect(sockfd,(struct sockaddr *) &servaddr,
sin_size))==0); //initiate a connection on a socket
   printf("connect sucessful\n");
   printf("Enter operation:\n +:Addition \n -: Subtraction \n /:
Division \n*:Multiplication \n#:To check Prime number \n");
   scanf("%c",&operator);
   if(operator=='#'){
   printf("Enter an operand:\n");
   scanf("%d", &op1);
   }else{
   printf("Enter operands:\n");
   scanf("%d %d", &op1, &op2);
   }
   write(sockfd,&operator,10);
   write(sockfd,&op1,sizeof(op1));
   write(sockfd,&op2,sizeof(op2));
   read(sockfd,&result,sizeof(result));
   printf("Operation result from server=%d\n",result);
   close(sockfd);
}
```

OUTPUT -



```
avi@abinash-gupta:~/Documents/CN/LAB5/Q4$ ./client
socket created sucessfully
connect sucessful
Enter operation:
 +:Addition
 -: Subtraction
 /: Division
*:Multiplication
#:To check Prime number
+
Enter operands:
1 3
Operation result from server=4
avi@abinash-gupta:~/Documents/CN/LAB5/Q4$
```



```
avi@abinash-gupta:~/Documents/CN/LAB5/Q4$ ./server
socket created sucessfully
bind sucessful
listen sucessful
accept sucessful
Result is: 1 + 3 = 4
avi@abinash-gupta:~/Documents/CN/LAB5/Q4$
```