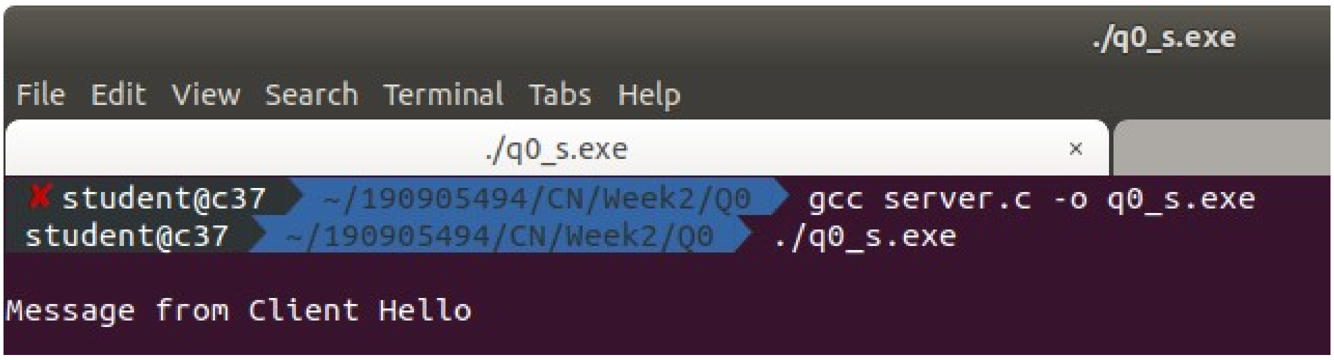
CN Lab #2

Practice Exersize

SERVER

```
//*
Concurrent Severs : Write a TCP concurrent Echo server and simple client.
*/

// server side
#include<stdio.h>
#include<string.h>
#include<stdlib.h>

#include<sys/types.h>
#include<sys/socket.h>

#include<netinet/in.h>
#include<arpa/inet.h>

#include <unistd.h>
#include <netdb.h>

#define PORTNO 10200

int main(){
    int sockfd,newsockfd,portno,clilen,n=1;
    char buf[256];

    struct sockaddr_in seraddr, cliaddr;

    int i,value;

    sockfd = socket(AF_INET,SOCK_STREAM,0);

    seraddr.sin_family = AF_INET;
    seraddr.sin_addr.s_addr = inet_addr("172.16.57.237"); // **

    seraddr.sin_port = htons(PORTNO);

    bind(sockfd,(struct sockaddr *)&seraddr,sizeof(seraddr));

    //Create a connection queue, ignore child exit details, and wait for clients

    listen(sockfd,5);

    while(1){
        //Accept the connection
        clilen = sizeof(clilen);
        newsockfd=accept(sockfd,(struct sockaddr *)&cliaddr,&clilen);

        //Fork to create a process for this client and perform a test to see whether
        //you’re the parent or the child:
```

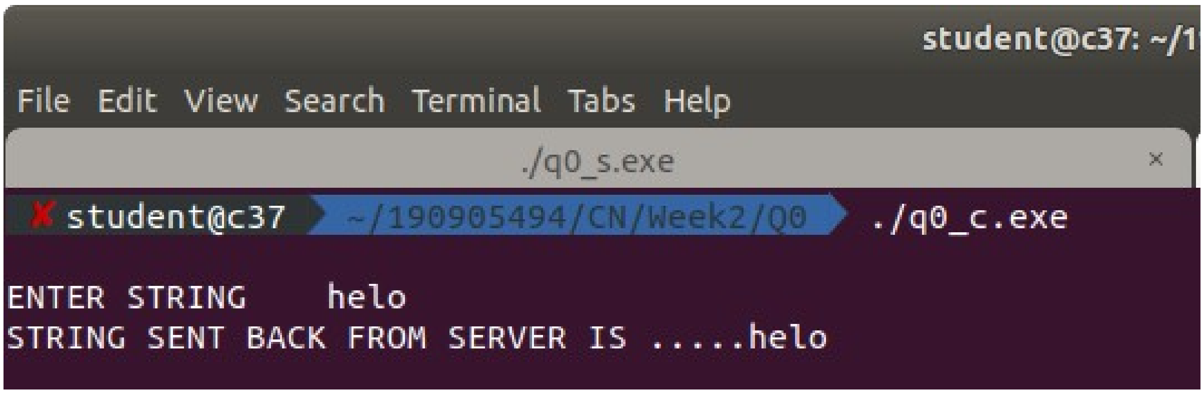
```
        if(fork()==0){      // If you're the child, you can now read/write to the client on newsockfd.
PAGE: 16

            n = read(newsockfd,buf,sizeof(buf));

            printf(" \nMessage from Client %s \n",buf);
            n = write(newsockfd,buf,sizeof(buf));

            close(newsockfd);exit(0);
        } else close(newsockfd);
        //Otherwise, you must be the parent and your work for this client is finished
    }
}
```



```
student@c37 ~/190905494/CN/Week2/Q0 gcc server.c -o q0_s.exe
student@c37 ~/190905494/CN/Week2/Q0 ./q0_s.exe

Message from Client Hello
```

CLIENT

```
/*
Concurrent Severs : Write a TCP concurrent Echo server and simple client.
*/

// client side
#include<stdio.h>
#include<string.h>
#include<stdlib.h>

#include<sys/types.h>
#include<sys/socket.h>

#include<netinet/in.h>
#include<arpa/inet.h>

#include <unistd.h>
#include <netdb.h>


int main(){
    int len,result,sockfd,n=1;
    struct sockaddr_in address;
    char ch[256],buf[256];

    //Create a socket for the client
    sockfd = socket(AF_INET, SOCK_STREAM, 0);

    //Name the socket as agreed with the server
    address.sin_family=AF_INET;
    address.sin_addr.s_addr=inet_addr("172.16.57.237"); // **
    address.sin_port=htons(10200);
    len = sizeof(address);

    //Connect your socket to the server's socket
    result=connect(sockfd,(struct sockaddr *)&address,len);

    if(result==-1){
        perror("\nCLIENT ERROR");
        exit(1);
    }

    //You can now read and write via sockfd (Logic for problem mentioned here)
    printf("\nENTER STRING\t");
    fgets(ch, 256, stdin);
    ch[strlen(ch)]='\0';
    write(sockfd,ch,strlen(ch));
    printf("STRING SENT BACK FROM SERVER IS .....");

    while(n){
        n=read(sockfd,buf,sizeof(buf));
        puts(buf);
    }
}

// **- indicates replace this address with your systems IP address
```



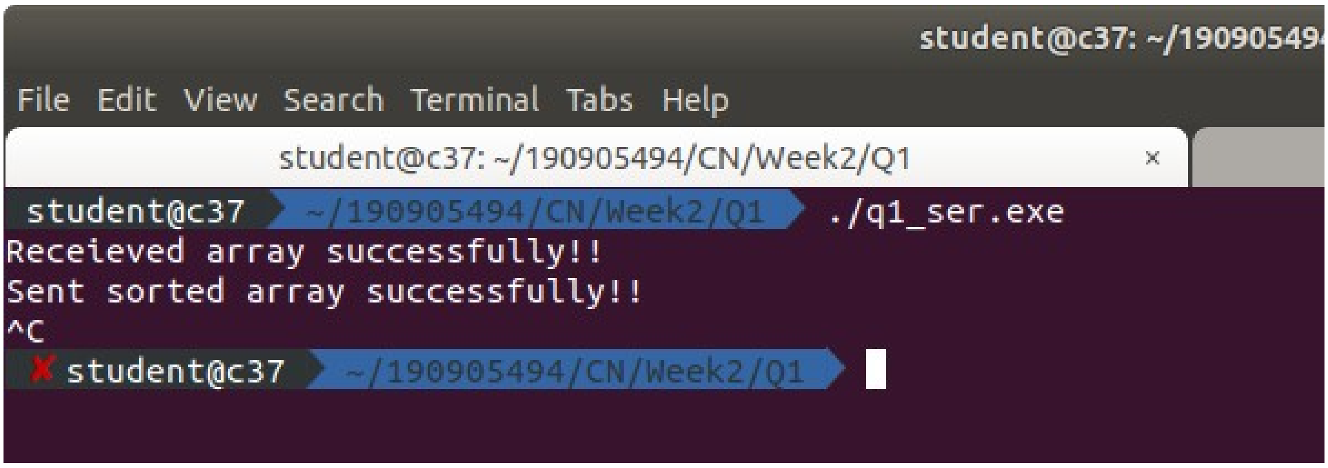
```
student@c37  ~/190905494/CN/Week2/Q0  ./q0_c.exe

ENTER STRING    helo
STRING SENT BACK FROM SERVER IS .....helo
```

Question 3

SERVER

```
/*
Write a TCP concurrent client server program where server accepts integer array
from client and sorts it and returns it to the client along with process id.
*/
// server side

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <sys/wait.h>
#include <signal.h>

void sort(int *arr) {
    for(int i=0;i<4;i++) {
        int min=i;

        for(int j=i;j<5;j++) {
            if(arr[j]<arr[min]) min=j;
        }

        if(i!=min) {
            int temp=arr[i];
            arr[i]=arr[min];
            arr[min]=temp;
        }
    }

}


void main()
```

```
{
   int sd,nd,n,len,reult;
   struct sockaddr_in seradress,cliaddr;
   int buf[5];

   sd=socket(AF_INET,SOCK_STREAM,0);
   seradress.sin_family=AF_INET;

   seradress.sin_addr.s_addr=htonl(INADDR_ANY);
   seradress.sin_port=htons(10200);

   bind(sd,(struct sockaddr*)&seradress,sizeof(seradress));

   listen(sd,5);
   len=sizeof(cliaddr);

   while(1){
      nd=accept(sd,(struct sockaddr*)&cliaddr,&len);

      if(fork()==0){
         close(sd);
         n=read(nd,buf,sizeof(buf));
         if(n==sizeof(buf))printf("Receieved array successfully!!\n");
         sort(buf);
         n=write(nd,buf,sizeof(buf));
         if(n==sizeof(buf))printf("Sent sorted array successfully!!\n");
      }
      close(nd);
   }
}
```


student@c37: ~/19090549
File Edit View Search Terminal Tabs Help
student@c37: ~/190905494/CN/Week2/Q1

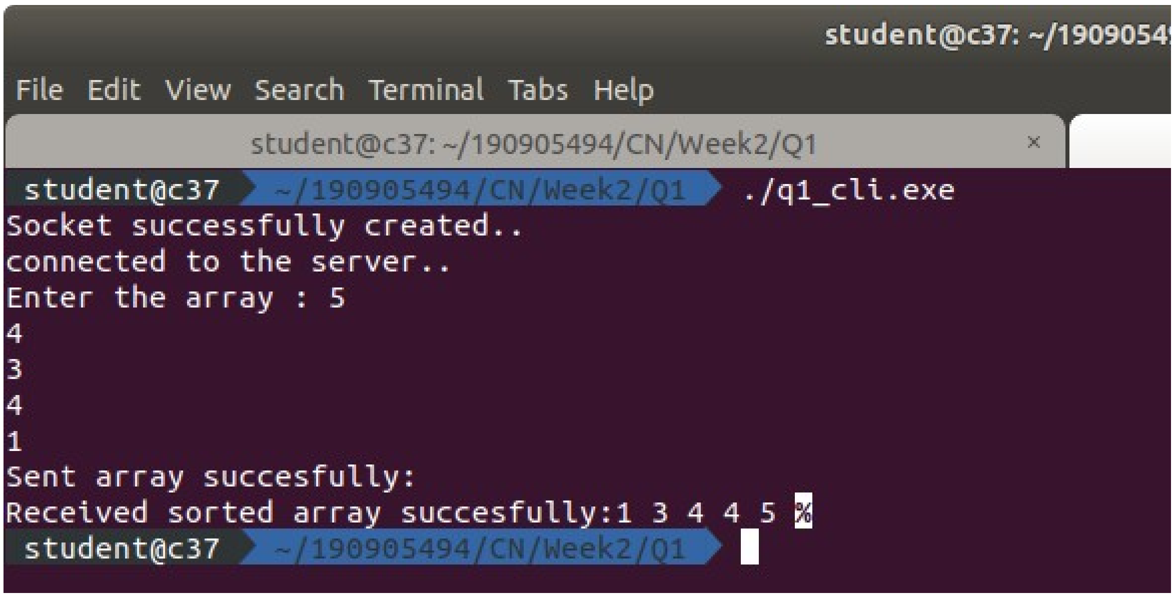
```
 student@c37  ~/190905494/CN/Week2/Q1  ./q1_ser.exe
Receieved array successfully!!
Sent sorted array successfully!!
^C
 ✘ student@c37  ~/190905494/CN/Week2/Q1 
```

CLIENT

```
/*
Write a TCP concurrent client server program where server accepts integer array
from client and sorts it and returns it to the client along with process id.
*/
// TCP client side
```

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

#define MAX 5
#define PORT 10200
#define SA struct sockaddr

void clifunc(int sockfd)
{
   int buff[MAX];
   int n;

   bzero(buff, sizeof(buff));
   printf("Enter the array : ");

   for(int i=0;i<MAX;i++) scanf("%d",&buff[i]);

   n = 0;
   n=write(sockfd, buff, sizeof(buff));
   if(n==sizeof(buff))

   {
       printf("Sent array succesfully:\n");
   }

   bzero(buff, sizeof(buff));
   n=read(sockfd, buff, sizeof(buff));
   if(n==sizeof(buff))
   {
       printf("Received sorted array succesfully:");
       for(int i=0;i<MAX;i++) printf("%d ",buff[i]);
   }

}


int main()
{
   int sockfd, connfd;
   struct sockaddr_in servaddr, cli;


   sockfd = socket(AF_INET, SOCK_STREAM, 0);
   if (sockfd == -1) {
       printf("socket creation failed...\n");
       exit(0);
   }
```

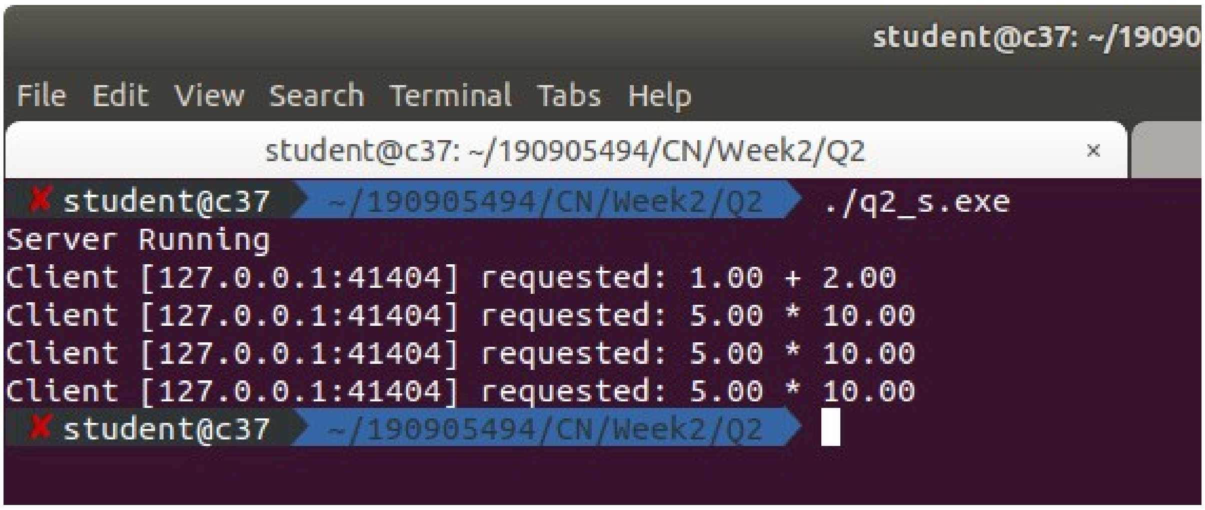
```
  else
     printf("Socket successfully created..\n");

  bzero(&servaddr, sizeof(servaddr));

  // assign IP, PORT
  servaddr.sin_family = AF_INET;
  servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
  servaddr.sin_port = htons(PORT);

  // connect the client socket to server socket
  if (connect(sockfd, (SA*)&servaddr, sizeof(servaddr)) != 0) {
     printf("connection with the server failed...\n");
     exit(0);
  }

  else
     printf("connected to the server..\n");

  // function for client
  clifunc(sockfd);

  // close the socket
  close(sockfd);

}
```


student@c37: ~/1909054
File Edit View Search Terminal Tabs Help
student@c37: ~/190905494/CN/Week2/Q1

```
 student@c37  ~/190905494/CN/Week2/Q1  ./q1_cli.exe
Socket successfully created..
connected to the server..
Enter the array : 5
4
3
4
1
Sent array succesfully:
Received sorted array succesfully:1 3 4 4 5 %
 student@c37  ~/190905494/CN/Week2/Q1 
```

Question 4
SERVER
/*
mplement concurrent Remote Math Server To perform arithmetic operations in the server and display the result at the client. The client accepts two integers and anoperator from the user and sends it to the server. The server then receives integers andoperator. The server will performs the operation on integers and sends result back tothe client which is displayed on the client screen. Then both the processes terminate.

```
*/
// server side

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <ctype.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#define MAXSIZE 150
#define PORT 5000
#define MAXLINE 1000

typedef struct obj
{
   double a,b,r;
   char op;
   char ans[10];
} obj1,*obj_ptr;

void main()
{
   int sockfd,newsockfd,retval;
   socklen_t actuallen;
   int recedbytes,sentbytes, sentans;
   struct sockaddr_in serveraddr,clientaddr;
   obj_ptr buffer = (obj_ptr)malloc(sizeof(obj1));
   sockfd=socket(AF_INET,SOCK_STREAM,0);
   if(sockfd==-1)
      printf("\nSocket creation error");
   serveraddr.sin_family=AF_INET;
   serveraddr.sin_port=htons(PORT);
   serveraddr.sin_addr.s_addr=htons(INADDR_ANY);
   bind(sockfd,(struct sockaddr*)&serveraddr,sizeof(serveraddr));
   puts("Server Running");
   listen(sockfd,1);
   actuallen=sizeof(clientaddr);
   newsockfd=accept(sockfd,(struct sockaddr*)&clientaddr,&actuallen);
   do
   {
      recv(newsockfd,buffer,sizeof(obj1),0);
      if(strcmp(buffer->ans, "stop") == 0)
      {
         puts("Stopping");
         close(sockfd);
         close(newsockfd);
      }
      else
      {
            printf("Client [%s:%d] requested: %.2lf %c %.2lf\n", inet_ntoa(clientaddr.sin_addr),
ntohs(clientaddr.sin_port), buffer->a, buffer->op, buffer->b);
            switch (buffer->op)
            {
               case '+': buffer->r = buffer->a + buffer->b;
                  break;
               case '-': buffer->r = buffer->a - buffer->b;
                  break;
               case '*': buffer->r = buffer->a * buffer->b;
                  break;
               case '/': buffer->r = buffer->a / buffer->b;
                  break;
               case '%': buffer->r = buffer->a / buffer->b;
                  break;
               default:
                  break;
            }
            sentbytes = send(newsockfd,buffer,sizeof(obj1),0);
         }
   } while(strcmp(buffer->ans, "stop") != 0);
}
```


student@c37: ~/19090
File Edit View Search Terminal Tabs Help
student@c37: ~/190905494/CN/Week2/Q2

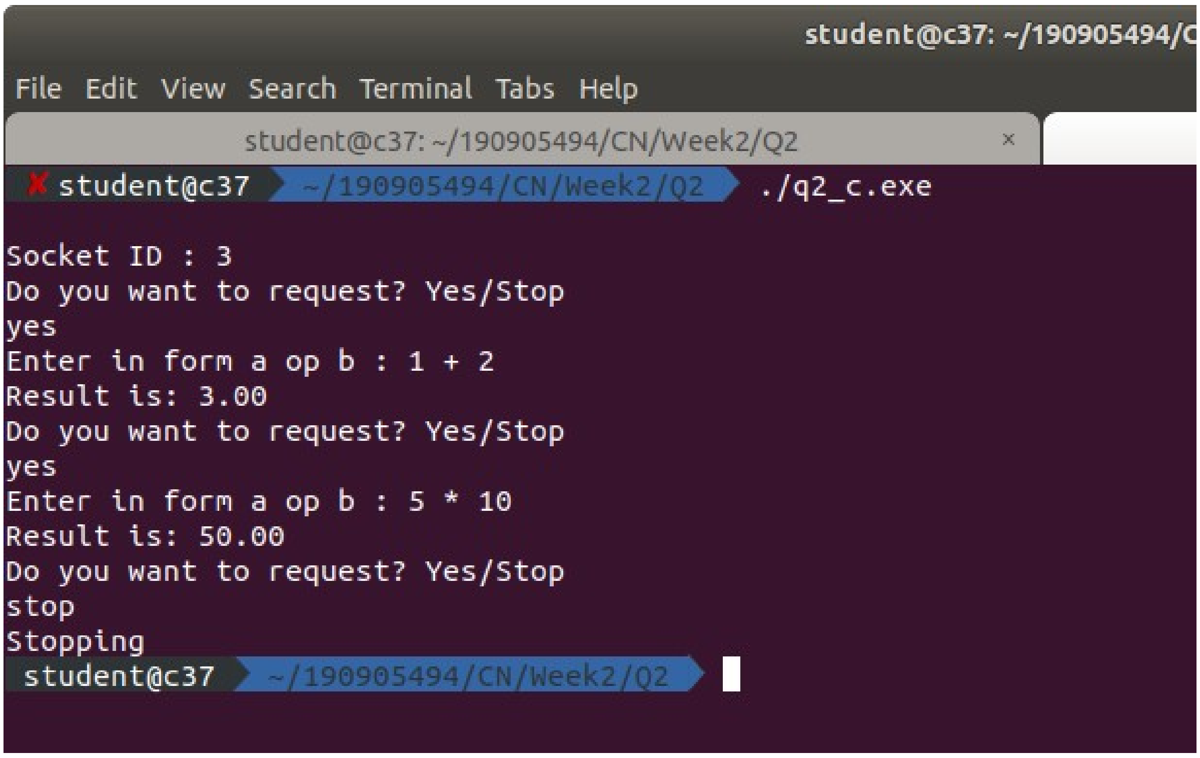
```
✗ student@c37  ~/190905494/CN/Week2/Q2  ./q2_s.exe
Server Running
Client [127.0.0.1:41404] requested: 1.00 + 2.00
Client [127.0.0.1:41404] requested: 5.00 * 10.00
Client [127.0.0.1:41404] requested: 5.00 * 10.00
Client [127.0.0.1:41404] requested: 5.00 * 10.00
✗ student@c37  ~/190905494/CN/Week2/Q2
```

CLIENT

```
/*
mplement concurrent Remote Math Server  To perform arithmetic operations in the server
and display the result at the client. The client accepts  two integers  and anoperator
from the user and sends it to the server. The server then receives integers andoperator.
The server will performs the operation on integers and sends result back tothe client which
is displayed on the client screen. Then both the processes terminate.
*/
// client side

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

```
#include <sys/stat.h>
#include <fcntl.h>
#include <arpa/inet.h>
#include <string.h>
#include <stdlib.h>
#define MAXSIZE 150
#define PORT 5000
#define MAXLINE 1000

typedef struct obj
{
   double a,b,r;
   char op;
   char ans[10];
} obj1,*obj_ptr;

void main()
{
   int sockfd,retval;char ch;
   int recedbytes,sentbytes, recans;
   struct sockaddr_in serveraddr;
   obj_ptr buffer = (obj_ptr)malloc(sizeof(obj1));
   obj_ptr buffer1 = (obj_ptr)malloc(sizeof(obj1));
   sockfd=socket(AF_INET,SOCK_STREAM,0);
   if(sockfd==-1)
      printf("\nSocket Creation Error");
   printf("\nSocket ID : %d\n",sockfd);
   serveraddr.sin_family=AF_INET;
   serveraddr.sin_port=htons(PORT);
   serveraddr.sin_addr.s_addr=htonl(INADDR_ANY);
   retval=connect(sockfd,(struct sockaddr*)&serveraddr,sizeof(serveraddr));
   if(retval==-1)
      printf("Connection error");
   do
   {
         printf("Do you want to request? Yes/Stop\n");
      scanf("%c",&ch);
      scanf("%[^\n]%*c",(buffer->ans));
      if(strcmp(buffer->ans,"stop")==0)
      {
         puts("Stopping");
         sentbytes=send(sockfd,buffer,sizeof(buffer),0);
         close(sockfd);
      }
      else
      {
         printf("Enter in form a op b : ");
         scanf("%lf %c %lf",&buffer->a, &buffer->op, &buffer->b);
         sentbytes=send(sockfd,buffer,sizeof(obj1),0);
                  recedbytes=recv(sockfd,buffer1,sizeof(obj1),0);
         printf("Result is: %.2lf \n",buffer1->r);
      }
```

```
    } while(strcmp(buffer->ans, "stop") != 0);
}
```

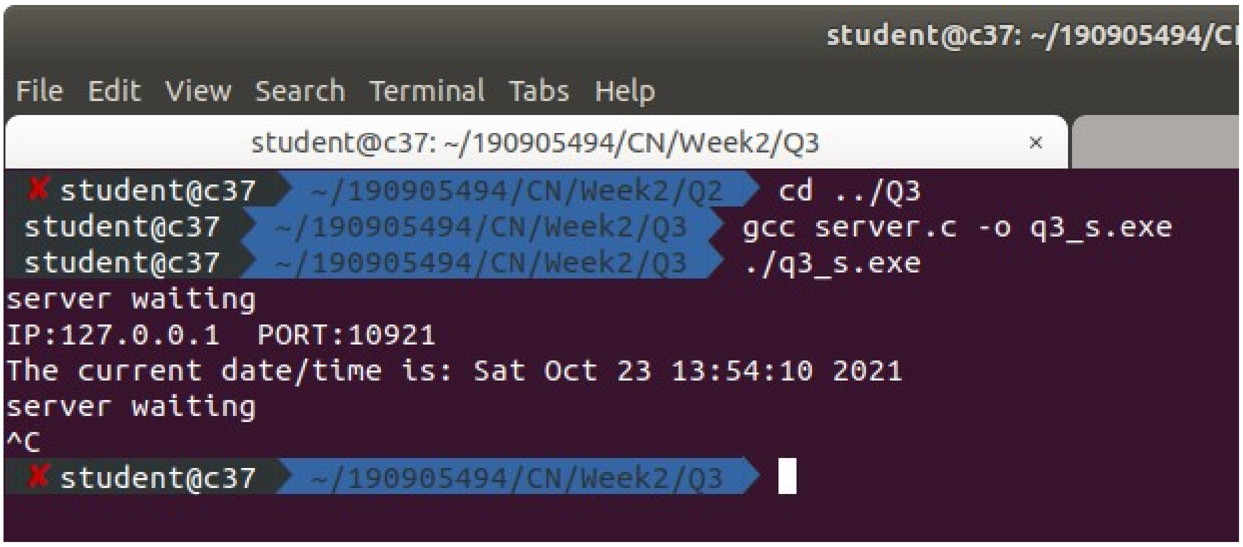

student@c37: ~/190905494/C
File Edit View Search Terminal Tabs Help
student@c37: ~/190905494/CN/Week2/Q2

```
✗ student@c37  ~/190905494/CN/Week2/Q2  ./q2_c.exe

Socket ID : 3
Do you want to request? Yes/Stop
yes
Enter in form a op b : 1 + 2
Result is: 3.00
Do you want to request? Yes/Stop
yes
Enter in form a op b : 5 * 10
Result is: 50.00
Do you want to request? Yes/Stop
stop
Stopping
 student@c37  ~/190905494/CN/Week2/Q2 
```

Question 5

SERVER

```
/*
Implement simple TCP daytime server using select().
*/
// server side

#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <stdlib.h>
#include <time.h>

void main()
{
    time_t rawtime;
    struct tm * timeinfo;
    char *reply;
    int server_sockfd, client_sockfd;
    int server_len, client_len;
    struct sockaddr_in server_address;
    struct sockaddr_in client_address;
    int hour,mins,sec,pid;
    /*  Create an unnamed socket for the server.  */
    server_sockfd = socket(AF_INET, SOCK_STREAM, 0);
    /*  Name the socket.  */
    server_address.sin_family = AF_INET;
    server_address.sin_addr.s_addr = inet_addr("127.0.0.1");
    server_address.sin_port = 9734;
    server_len = sizeof(server_address);
    bind(server_sockfd, (struct sockaddr *)&server_address, server_len);
    /*  Create a connection queue and wait for clients.  */
    listen(server_sockfd, 5);

    while(1)
    {
        char ch;
        printf("server waiting\n");
    /*  Accept a connection.  */
        client_len = sizeof(client_address);
        client_sockfd = accept(server_sockfd, (struct sockaddr *)&client_address, &client_len);
    /*  We can now read/write to client on client_sockfd.  */
    //char *inet_ntoa(client_addr.sin_addr);
        char * ip_add =inet_ntoa(client_address.sin_addr);
        int port=client_address.sin_port;
        printf("IP:%s  PORT:%d\n", ip_add,port);
        //get the time
        time ( &rawtime );
        timeinfo = localtime ( &rawtime );
        reply = asctime(timeinfo);
        printf ( "The current date/time is: %s", reply );
        hour =  timeinfo->tm_hour;
        mins = timeinfo->tm_min;
        sec = timeinfo->tm_sec;
        pid = getpid();
        write(client_sockfd, &hour, 1);
        write(client_sockfd, &mins, 1);
        write(client_sockfd, &sec, 1);
        write(client_sockfd, &pid, 1);
        //close(client_sockfd);
    }
}
```

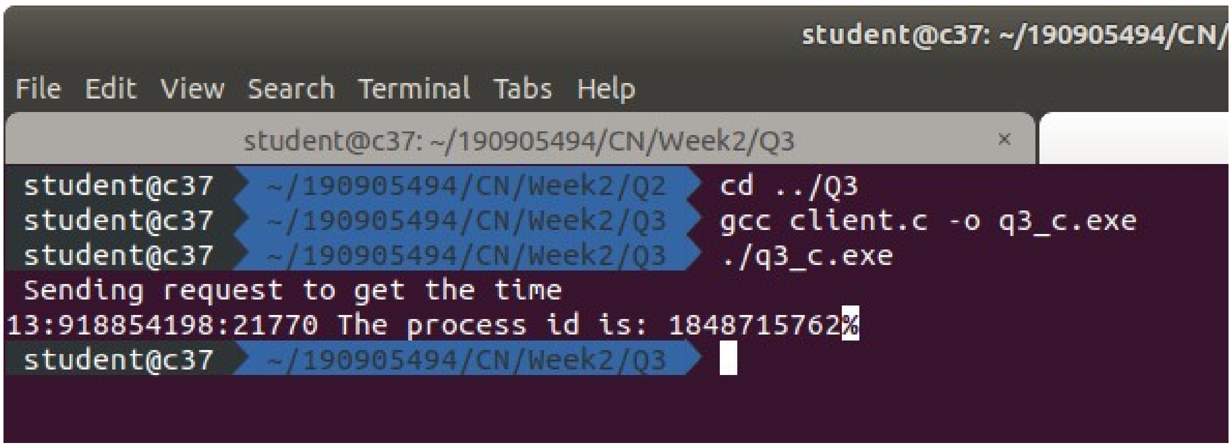

student@c37: ~/190905494/C
File Edit View Search Terminal Tabs Help
student@c37: ~/190905494/CN/Week2/Q3

```
 ✘ student@c37  ~/190905494/CN/Week2/Q2  cd ../Q3
 student@c37  ~/190905494/CN/Week2/Q3  gcc server.c -o q3_s.exe
 student@c37  ~/190905494/CN/Week2/Q3  ./q3_s.exe
server waiting
IP:127.0.0.1  PORT:10921
The current date/time is: Sat Oct 23 13:54:10 2021
server waiting
^C
 ✘ student@c37  ~/190905494/CN/Week2/Q3 
```

CLIENT

```
/*
Implement simple TCP daytime server using select().
*/
// client side

#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <stdlib.h>
#include <time.h>

void main()
{
   int sockfd;
   int len;
   struct sockaddr_in address;
   struct tm * timeinfo;
   int result;
   char *reply;
   int hour,mins,sec,pid;
   /*  Create a socket for the client.  */
   sockfd = socket(AF_INET, SOCK_STREAM, 0);
   /*  Name the socket, as agreed with the server.  */
   address.sin_family = AF_INET;
   address.sin_addr.s_addr = inet_addr("127.0.0.1");
   address.sin_port = 9734;
   len = sizeof(address);
   /*  Now connect our socket to the server socket. */
   result = connect(sockfd, (struct sockaddr *)&address, len);

   if(result == -1)
   {
      perror("oops: client2");
```

```
        exit(1);
    }

    /*  We can now read/write via sockfd.  */
    printf(" Sending request to get the time\n");
    read(sockfd, &hour , 1);
    read(sockfd, &mins , 1);
    read(sockfd, &sec , 1);
    read(sockfd, &pid , 1);
    printf("%d:%d:%d", hour, mins, sec);
    printf(" The process id is: %d",pid);
    close(sockfd);
    exit(0);
}
```


student@c37: ~/190905494/CN/
File Edit View Search Terminal Tabs Help
student@c37: ~/190905494/CN/Week2/Q3

```
 student@c37  ~/190905494/CN/Week2/Q2  cd ../Q3
 student@c37  ~/190905494/CN/Week2/Q3  gcc client.c -o q3_c.exe
 student@c37  ~/190905494/CN/Week2/Q3  ./q3_c.exe
 Sending request to get the time
13:918854198:21770 The process id is: 1848715762%
 student@c37  ~/190905494/CN/Week2/Q3 
```