Assignment-8

Read the content of File

1) Server

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <signal.h>
#include <netinet/in.h>
#include <unistd.h>
#include <string.h>
#include <arpa/inet.h>
#define TRUE 1

int main()
{
  int soc, chpid,msgsock;
  socklen_t len;
  char buf[1024],buf1[1024];
  struct sockaddr_in server, client;
  socklen_t addrlen=sizeof(client);

/* First call to socket() function */
  soc = socket(AF_INET, SOCK_STREAM, 0);

  if (soc < 0) {
    perror("ERROR opening socket");
    exit(1);
  }

  server.sin_family = AF_INET;
//Set IP address to localhost

  server.sin_addr.s_addr = inet_addr("127.0.0.1");
//Set port number, using htons function to use proper byte order

  server.sin_port = 0;
```

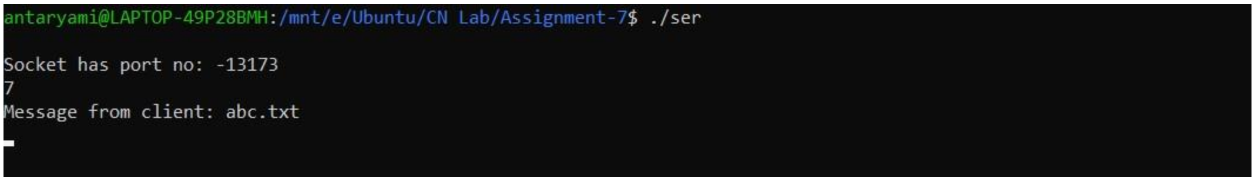
```
if (bind(soc, (struct sockaddr *) &server, sizeof(server)) < 0) {
  perror("ERROR on binding");
  exit(1);
}

len=sizeof(server);
if(getsockname(soc,(struct sockaddr *)&server,&len))
{
    perror("\nError in getting port..");
    exit(3);
}
printf("\nSocket has port no: %hd\n",htons(server.sin_port));

listen(soc,5); // Listen on the socket, with 5 max connection requests queued
signal(SIGCHLD,SIG_IGN);
do
{
    /* Accept actual connection from the client */
    msgsock = accept(soc, (struct sockaddr *)&client,(socklen_t*) &addrlen);
    if(msgsock == -1){
        perror("\nError in accept..");
        exit(0);
    }else{
        if((chpid=fork())==0){
            close(soc);
            do{
                read(msgsock,buf,1024);
                printf("%d",strlen(buf));
                printf("\nMessage from client: %s\n",buf);
                FILE* fp=fopen(buf,"r");
                if (NULL == fp) {
                      printf("file can't be opened \n");
                }
                char ch;
                int ctr=0;
                do {
                    ch = fgetc(fp);
                    buf1[ctr]=ch;
                    ctr++;
                } while (ch != EOF);
                buf1[ctr]='\0';
                write(msgsock,buf1,1024);
             }while(strcmp(buf1,"bye")!=0);
```

```
                        close(msgsock);
                        exit(0);
                }
                else
                        close(msgsock);
        }
  }while(TRUE);
  close(soc);
  return 0;
}
```

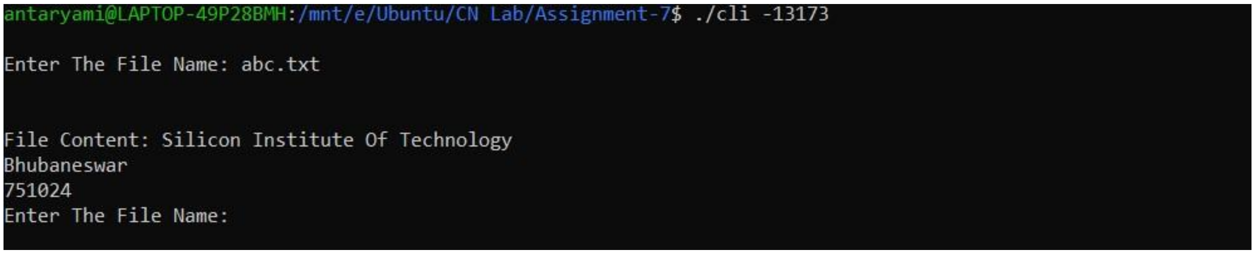
2) Client

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <signal.h>
#include <netinet/in.h>
#include <unistd.h>
#include <string.h>
#include <arpa/inet.h>
#define TRUE 1

int main(int argc, char *argv[])
{
  int soc, chpid,msgsock;
  socklen_t len;
  char buf[1024],buf1[1024];
  struct sockaddr_in server, client;
  socklen_t addrlen=sizeof(client);

/* First call to socket() function */
  soc = socket(AF_INET, SOCK_STREAM, 0);

  if (soc < 0) {
    perror("ERROR opening socket");
    exit(1);
  }
  server.sin_family = AF_INET;
  //Set IP address to localhost
```

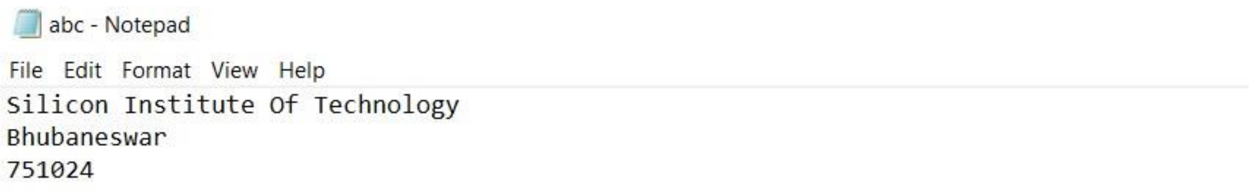
```
    server.sin_addr.s_addr = inet_addr("127.0.0.1");
    //Set port number, using htons function to use proper byte order

    server.sin_port = htons(atoi(argv[1]));

    if(connect(soc,(struct sockaddr *)&server,sizeof(server)) < 0) {
            perror("\nError in connection...");
            exit(2);
    }
    do{
            printf("\nEnter The File Name: ");
            scanf(" %[^\n]",buf);
            write(soc,buf,1024);
            printf("\n");
            read(soc,buf1,1024);
            printf("\nFile Content: %s",buf1);
    }while(strcmp(buf,"bye")!=0);
    return 0;
  }
```



```
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-7$ ./ser

Socket has port no: -13173
7
Message from client: abc.txt
```



```
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-7$ ./cli -13173

Enter The File Name: abc.txt


File Content: Silicon Institute Of Technology
Bhubaneswar
751024
Enter The File Name:
```



```
Silicon Institute Of Technology
Bhubaneswar
751024
```

Assignment-7

File Creation with content in client site

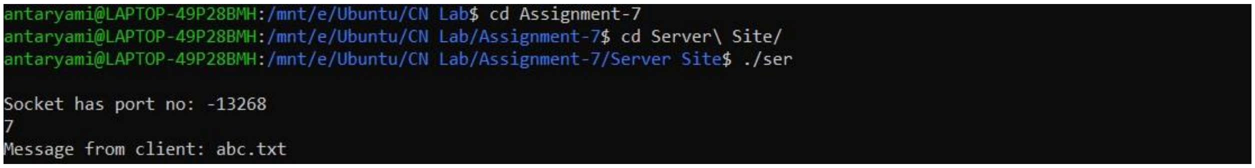
1) Server

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <signal.h>
#include <netinet/in.h>
#include <unistd.h>
#include <arpa/inet.h>
#define TRUE 1

int main()
{
  int soc, chpid,msgsock;
  socklen_t len;
  char buf[1024],buf1[1024];
  struct sockaddr_in server, client;
  socklen_t addrlen=sizeof(client);

  soc = socket(AF_INET, SOCK_STREAM, 0);
  if (soc < 0) {
    perror("ERROR opening socket");
    exit(1);
  }
  server.sin_family = AF_INET;
  server.sin_addr.s_addr = inet_addr("127.0.0.1");
  server.sin_port = 0;

  if (bind(soc, (struct sockaddr *) &server, sizeof(server)) < 0) {
    perror("ERROR on binding");
    exit(1);
  }
  len=sizeof(server);
  if(getsockname(soc,(struct sockaddr *)&server,&len))
  {
          perror("\nError in getting port..");
          exit(3);
  }
  printf("\nSocket has port no: %hd\n",htons(server.sin_port));

  listen(soc,5); // Listen on the socket, with 5 max connection requests queued
  signal(SIGCHLD,SIG_IGN);
  do
  {
     msgsock = accept(soc, (struct sockaddr *)&client,(socklen_t*) &addrlen);
                if(msgsock == -1){
                        perror("\nError in accept..");
                        exit(0);
                }
                else{
                        if((chpid=fork())==0){
                                close(soc);
                                do{
                                        read(msgsock,buf,1024);
                                        printf("%d",strlen(buf));
                                        printf("\nMessage from client: %s\n",buf);
                                        FILE* fp=fopen(buf,"r");
                                        if (NULL == fp) {
                                                  printf("file can't be opened \n");
                                        }
                                        char ch;
                                        int ctr=0;
                                        do{
                                                ch = fgetc(fp);
                                                buf1[ctr]=ch;
                                                ctr++;
                                        }while(ch != EOF);

                                        buf1[ctr]='\0';
                                        write(msgsock,buf1,1024);

                                }while(strcmp(buf1,"bye")!=0);

                                close(msgsock);
                                exit(0);
                        }else
                                close(msgsock);
                }
  }while(TRUE);
  close(soc);
  return 0;
}
```

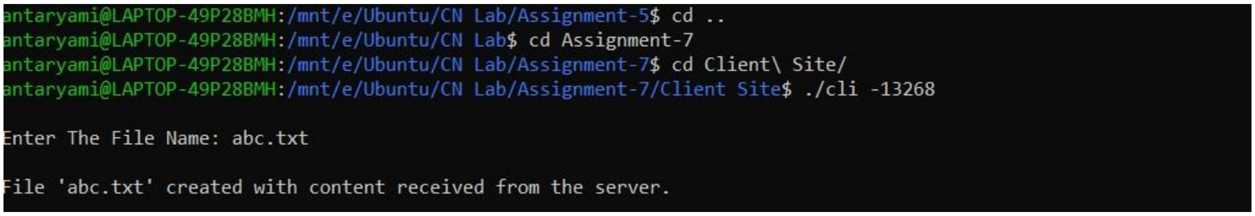
2) Client

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <signal.h>
#include <netinet/in.h>
#include <unistd.h>
#include <arpa/inet.h>
#define TRUE 1

int main(int argc, char *argv[])
{
  int soc, chpid,msgsock;
  socklen_t len;
  char buf[1024],buf1[1024];
  struct sockaddr_in server, client;
  socklen_t addrlen=sizeof(client);

  soc = socket(AF_INET, SOCK_STREAM, 0);

  if (soc < 0) {
    perror("ERROR opening socket");
    exit(1);
  }

  server.sin_family = AF_INET;
  server.sin_addr.s_addr = inet_addr("127.0.0.1");
  server.sin_port = htons(atoi(argv[1]));

  if(connect(soc,(struct sockaddr *)&server,sizeof(server)) < 0){
          perror("\nError in connection...");
          exit(2);
  }
  do {
          printf("\nEnter The File Name: ");
          scanf(" %[^\n]",buf);
          write(soc,buf,1024);
          printf("\n");

          FILE *file = fopen(buf, "w"); // Open a new file in write mode
```

```
                if(file == NULL) {
                    perror("Error creating file");
                    exit(3);
                }
                read(soc,buf1,1024);
                fprintf(file, "%s", buf1); // Write the content to the file
                fclose(file); // Close the file

                printf("File '%s' created with content received from the server.\n", buf);
        }while(strcmp(buf,"bye")!=0);

        return 0;
    }
```



```
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab$ cd Assignment-7
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-7$ cd Server\ Site/
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-7/Server Site$ ./ser

Socket has port no: -13268
7
Message from client: abc.txt
```



```
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-5$ cd ..
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab$ cd Assignment-7
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-7$ cd Client\ Site/
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-7/Client Site$ ./cli -13268

Enter The File Name: abc.txt

File 'abc.txt' created with content received from the server.
```

Assignment-6

1) Concurrent UDP Server

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

int main(void) {
  int socket_desc;
  struct sockaddr_in server_addr, client_addr;
  char client_message[2000], server_message[2000];
  int client_struct_length = sizeof(client_addr);

  memset(client_message, '\0', sizeof(client_message));
  memset(server_message, '\0', sizeof(server_message));

  // Create UDP socket:
  socket_desc = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);
  if (socket_desc < 0) {
    printf("Error while creating socket\n");
    return -1;
  }
  printf("Socket created successfully\n");

  server_addr.sin_family = AF_INET;
  server_addr.sin_port = htons(3000);
  server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");

  if (bind(socket_desc, (struct sockaddr*)&server_addr, sizeof(server_addr)) < 0) {
    printf("Couldn't bind to the port\n");
    return -1;
  }
  printf("Done with binding\n");
  printf("Listening for incoming messages...\n\n");

while (1) {
    // Receive client's message:
    if (recvfrom(socket_desc, client_message, sizeof(client_message), 0,
      (struct sockaddr*)&client_addr, &client_struct_length) < 0) {
      printf("Couldn't receive\n");
```

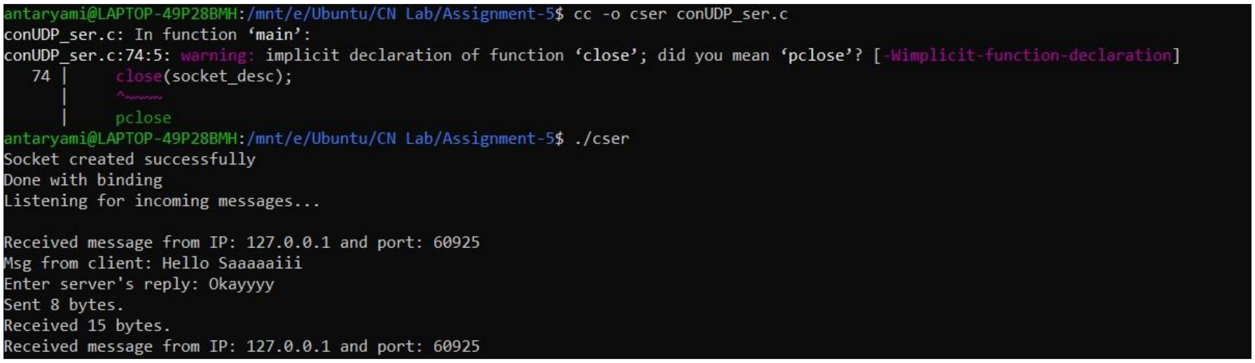
```
            return -1;
        }

        printf("Received message from IP: %s and port: %i\n",
              inet_ntoa(client_addr.sin_addr), ntohs(client_addr.sin_port));
        printf("Msg from client: %s\n", client_message);

        // Prompt user for server's reply:
        printf("Enter server's reply: ");
        fgets(server_message, sizeof(server_message), stdin);

        // Respond to client with the user-provided reply:
        if (sendto(socket_desc, server_message, strlen(server_message), 0,
            (struct sockaddr*)&client_addr, client_struct_length) < 0) {
            printf("Can't send\n");
            return -1;
        }
        printf("Sent %ld bytes.\n", strlen(server_message));
        printf("Received %ld bytes.\n", strlen(client_message));

        // Check if the client wants to exit
        if (strcmp(client_message, "exit") == 0) {
            printf("Exiting server...\n");
            break;
        }
    }

    // Close the socket:
    close(socket_desc);
    return 0;
}
```

2) Concurrent UDP Client

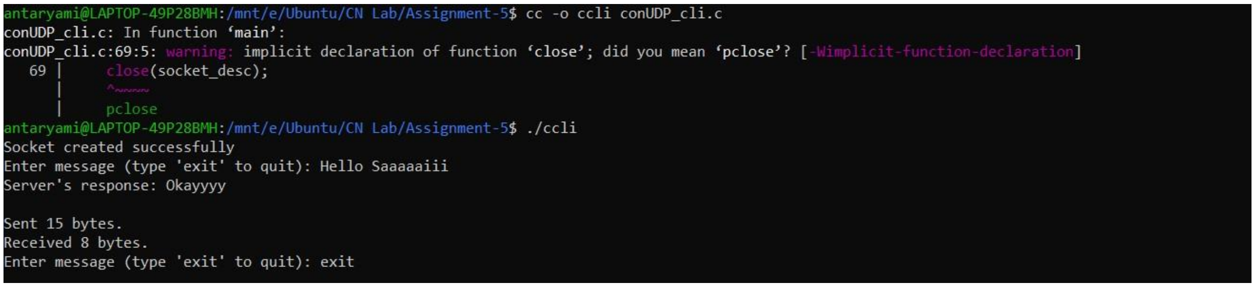
```
#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <arpa/inet.h>

int main(void) {
    int socket_desc;
    struct sockaddr_in server_addr;
    char server_message[2000], client_message[2000];
```

```
int server_struct_length = sizeof(server_addr);

// Clean buffers:
memset(server_message, '\0', sizeof(server_message));
memset(client_message, '\0', sizeof(client_message));

// Create socket:
socket_desc = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);
if (socket_desc < 0) {
    printf("Error while creating socket\n");
    return -1;
}
printf("Socket created successfully\n");

// Set port and IP:
server_addr.sin_family = AF_INET;
server_addr.sin_port = htons(3000);
server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");

while (1) {
    // Get input from the user:
    printf("Enter message (type 'exit' to quit): ");
    fgets(client_message, sizeof(client_message), stdin);

    // Remove the newline character from the input
    size_t len = strlen(client_message);
    if (len > 0 && client_message[len - 1] == '\n') {
        client_message[len - 1] = '\0';
    }

    // Send the message to the server:
    if (sendto(socket_desc, client_message, strlen(client_message), 0,
            (struct sockaddr*)&server_addr, server_struct_length) < 0) {
        printf("Unable to send message\n");
        return -1;
    }

    // Receive the server's response:
    if (recvfrom(socket_desc, server_message, sizeof(server_message), 0,
             (struct sockaddr*)&server_addr, &server_struct_length) < 0) {
        printf("Error while receiving server's msg\n");
        return -1;
    }
```

```
        printf("Server's response: %s\n", server_message);

        // Display the number of bytes/characters sent and received
        printf("Sent %ld bytes.\n", strlen(client_message));
        printf("Received %ld bytes.\n", strlen(server_message));

        // Check if the user wants to exit
        if (strcmp(client_message, "exit") == 0) {
            printf("Exiting client...\n");
            break;
        }
    }

    // Close the socket:
    close(socket_desc);

    return 0;
}
```



```
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-5$ cc -o cser conUDP_ser.c
conUDP_ser.c: In function ‘main’:
conUDP_ser.c:74:5: warning: implicit declaration of function ‘close’; did you mean ‘pclose’? [-Wimplicit-function-declaration]
   74 |     close(socket_desc);
      |     ^~~~~
      |     pclose
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-5$ ./cser
Socket created successfully
Done with binding
Listening for incoming messages...

Received message from IP: 127.0.0.1 and port: 60925
Msg from client: Hello Saaaaaiii
Enter server's reply: Okayyyy
Sent 8 bytes.
Received 15 bytes.
Received message from IP: 127.0.0.1 and port: 60925
```



```
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-5$ cc -o ccli conUDP_cli.c
conUDP_cli.c: In function ‘main’:
conUDP_cli.c:69:5: warning: implicit declaration of function ‘close’; did you mean ‘pclose’? [-Wimplicit-function-declaration]
   69 |     close(socket_desc);
      |     ^~~~~
      |     pclose
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-5$ ./ccli
Socket created successfully
Enter message (type 'exit' to quit): Hello Saaaaaiii
Server's response: Okayyyy

Sent 15 bytes.
Received 8 bytes.
Enter message (type 'exit' to quit): exit
```

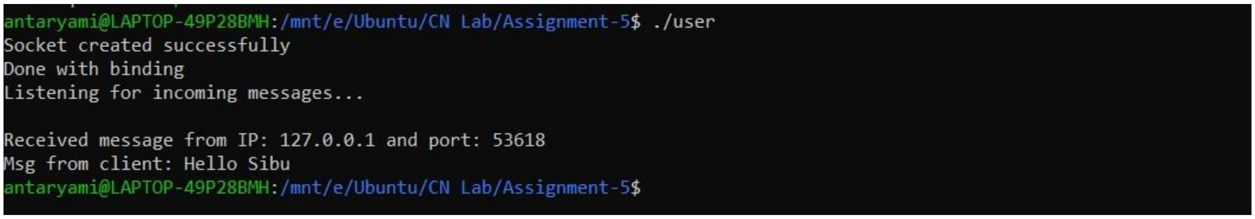
Assignment-5

1) UDP Server

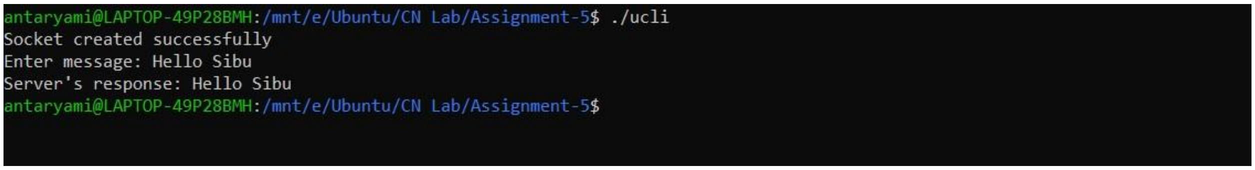
```
#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <arpa/inet.h>
int main(void){
        int socket_desc;
        struct sockaddr_in server_addr, client_addr;
        char server_message[2000], client_message[2000];
        int client_struct_length = sizeof(client_addr);

        // Create UDP socket:
        socket_desc = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);

        // Set IPPROTO_UDP to 0
        // if there is error msg.

        if(socket_desc < 0){
        printf("Error while creating socket\n");
           return -1;
        }
        printf("Socket created successfully\n");

        // Set port and IP:
        server_addr.sin_family = AF_INET;
        server_addr.sin_port = htons(3000);
        server_addr.sin_addr.s_addr = inet_addr("127.0.0.1"); // You can use
(INADDR_ANY)

        // Bind to the set port and IP:
        if(bind(socket_desc, (struct sockaddr*)&server_addr, sizeof(server_addr)) < 0){
        printf("Couldn't bind to the port\n");
 return -1;
 }
        printf("Done with binding\n");

        printf("Listening for incoming messages...\n\n");

        // Receive client's message:
        if (recvfrom(socket_desc, client_message, sizeof(client_message), 0,
        (struct sockaddr*)&client_addr, &client_struct_length) < 0){
            printf("Couldn't receive\n");
            return -1;
        }
        printf("Received message from IP: %s and port: %i\n",
        inet_ntoa(client_addr.sin_addr), ntohs(client_addr.sin_port));
        printf("Msg from client: %s\n", client_message);

        strcpy(server_message, client_message);

        if (sendto(socket_desc, server_message, strlen(server_message), 0,
                                  (struct sockaddr*)&client_addr, client_struct_length) < 0){
                printf("Can't send\n");
                return -1;
        }
        close(socket_desc);
        return 0;
}
```

2) UDP Client

```
#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <arpa/inet.h>
int main(void){
        int socket_desc;
        struct sockaddr_in server_addr;
        char server_message[2000], client_message[2000];
        int server_struct_length = sizeof(server_addr);

        // Clean buffers:
        memset(server_message, '\0', sizeof(server_message));
        memset(client_message, '\0', sizeof(client_message));

         // Create socket:
         socket_desc = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);

         if(socket_desc < 0){
              printf("Error while creating socket\n");
              return -1;
          }
          printf("Socket created successfully\n");
        // Set port and IP:
        server_addr.sin_family = AF_INET;
        server_addr.sin_port = htons(3000);
        server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");

        // Get input from the user:
        printf("Enter message: ");
        gets(client_message);

        // Send the message to server:
        if(sendto(socket_desc, client_message, strlen(client_message), 0,
                (struct sockaddr*)&server_addr, server_struct_length) < 0){
        printf("Unable to send message\n");
            return -1;
        }

        // Receive the server's response:
        if(recvfrom(socket_desc, server_message, sizeof(server_message), 0,
                    (struct sockaddr*)&server_addr, &server_struct_length) < 0){
                printf("Error while receiving server's msg\n");
                return -1;
        }

        printf("Server's response: %s\n", server_message);
        close(socket_desc);

        return 0;
}
```



```
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-5$ ./user
Socket created successfully
Done with binding
Listening for incoming messages...

Received message from IP: 127.0.0.1 and port: 53618
Msg from client: Hello Sibu
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-5$
```



```
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-5$ ./ucli
Socket created successfully
Enter message: Hello Sibu
Server's response: Hello Sibu
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-5$
```

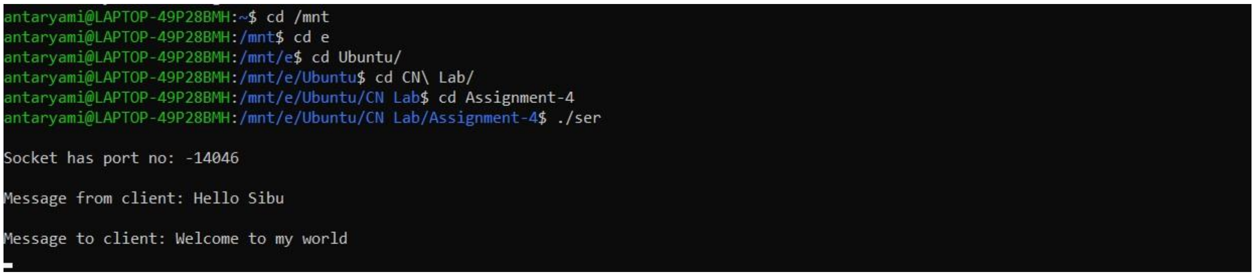
Assignment-4

1) Concurrent Server

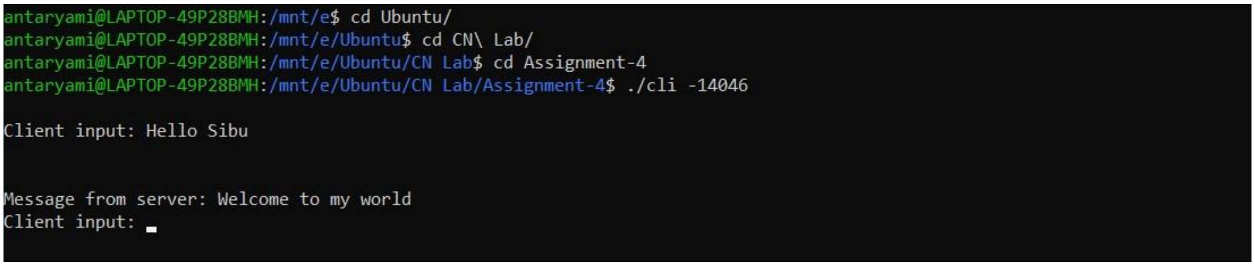
```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <signal.h>
#include <netinet/in.h>
#include <unistd.h>
#include <string.h>
#include <arpa/inet.h>
#define TRUE 1

int main()
{
  int soc, chpid,msgsock;
  socklen_t len;
  char buf[1024],buf1[1024];
  struct sockaddr_in server, client;
  socklen_t addrlen=sizeof(client);

/* First call to socket() function */
  soc = socket(AF_INET, SOCK_STREAM, 0);

  if (soc < 0) {
    perror("ERROR opening socket");
    exit(1);
  }

  server.sin_family = AF_INET;
//Set IP address to localhost
  server.sin_addr.s_addr = inet_addr("127.0.0.1");

//Set port number, using htons function to use proper byte order
  server.sin_port = 0;

/* Now bind the host address using bind() call.*/
  if (bind(soc, (struct sockaddr *) &server, sizeof(server)) < 0) {
    perror("ERROR on binding");
    exit(1);
  }

  len=sizeof(server);
  if(getsockname(soc,(struct sockaddr *)&server,&len))
  {
            perror("\nError in getting port..");
            exit(3);
  }
  printf("\nSocket has port no: %hd\n",htons(server.sin_port));

  listen(soc,5); // Listen on the socket, with 5 max connection requests queued
  signal(SIGCHLD,SIG_IGN);
  do
  {
          /* Accept actual connection from the client */
          msgsock = accept(soc, (struct sockaddr *)&client,(socklen_t*) &addrlen);
          if(msgsock == -1)
          {
                    perror("\nError in accept..");
                    exit(0);
          }
          else
          {
                    if((chpid=fork())==0)
                    {
                              close(soc);
                              do
                              {
                                        read(msgsock,buf,1024);
                                        printf("\nMessage from client: %s\n",buf);
                                        printf("\nMessage to client: ");
                                        scanf(" %[^\n]",buf1);
                                        write(msgsock,buf1,1024);
                              }while(strcmp(buf1,"bye")!=0);
                              close(msgsock);
                              exit(0);
                    }
                    else
                              close(msgsock);
          }
  }while(TRUE);
  close(soc);
  return 0;
}
```

2) Concurrent Client

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <signal.h>
#include <netinet/in.h>
#include <unistd.h>
#include <string.h>
#include <arpa/inet.h>
#define TRUE 1

int main(int argc, char *argv[])
{
  int soc, chpid,msgsock;
  socklen_t len;
  char buf[1024],buf1[1024];
  struct sockaddr_in server, client;
  socklen_t addrlen=sizeof(client);

/* First call to socket() function */
  soc = socket(AF_INET, SOCK_STREAM, 0);

  if (soc < 0) {
    perror("ERROR opening socket");
    exit(1);
  }


  server.sin_family = AF_INET;
//Set IP address to localhost
  server.sin_addr.s_addr = inet_addr("127.0.0.1");
//Set port number, using htons function to use proper byte order
  server.sin_port = htons(atoi(argv[1]));
if(connect(soc,(struct sockaddr *)&server,sizeof(server)) < 0)
 {
        perror("\nError in connection...");
        exit(2);
 }
 do
 {
        printf("\nClient input: ");
        scanf(" %[^\n]",buf);
        write(soc,buf,1024);
        printf("\n");
        read(soc,buf1,1024);
        printf("\nMessage from server: %s",buf1);
 }while(strcmp(buf,"bye")!=0);
 return 0;
}
```



```
antaryami@LAPTOP-49P28BMH:~$ cd /mnt
antaryami@LAPTOP-49P28BMH:/mnt$ cd e
antaryami@LAPTOP-49P28BMH:/mnt/e$ cd Ubuntu/
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu$ cd CN\ Lab/
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab$ cd Assignment-4
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-4$ ./ser

Socket has port no: -14046

Message from client: Hello Sibu

Message to client: Welcome to my world
```



```
antaryami@LAPTOP-49P28BMH:/mnt/e$ cd Ubuntu/
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu$ cd CN\ Lab/
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab$ cd Assignment-4
antaryami@LAPTOP-49P28BMH:/mnt/e/Ubuntu/CN Lab/Assignment-4$ ./cli -14046

Client input: Hello Sibu


Message from server: Welcome to my world
Client input:
```