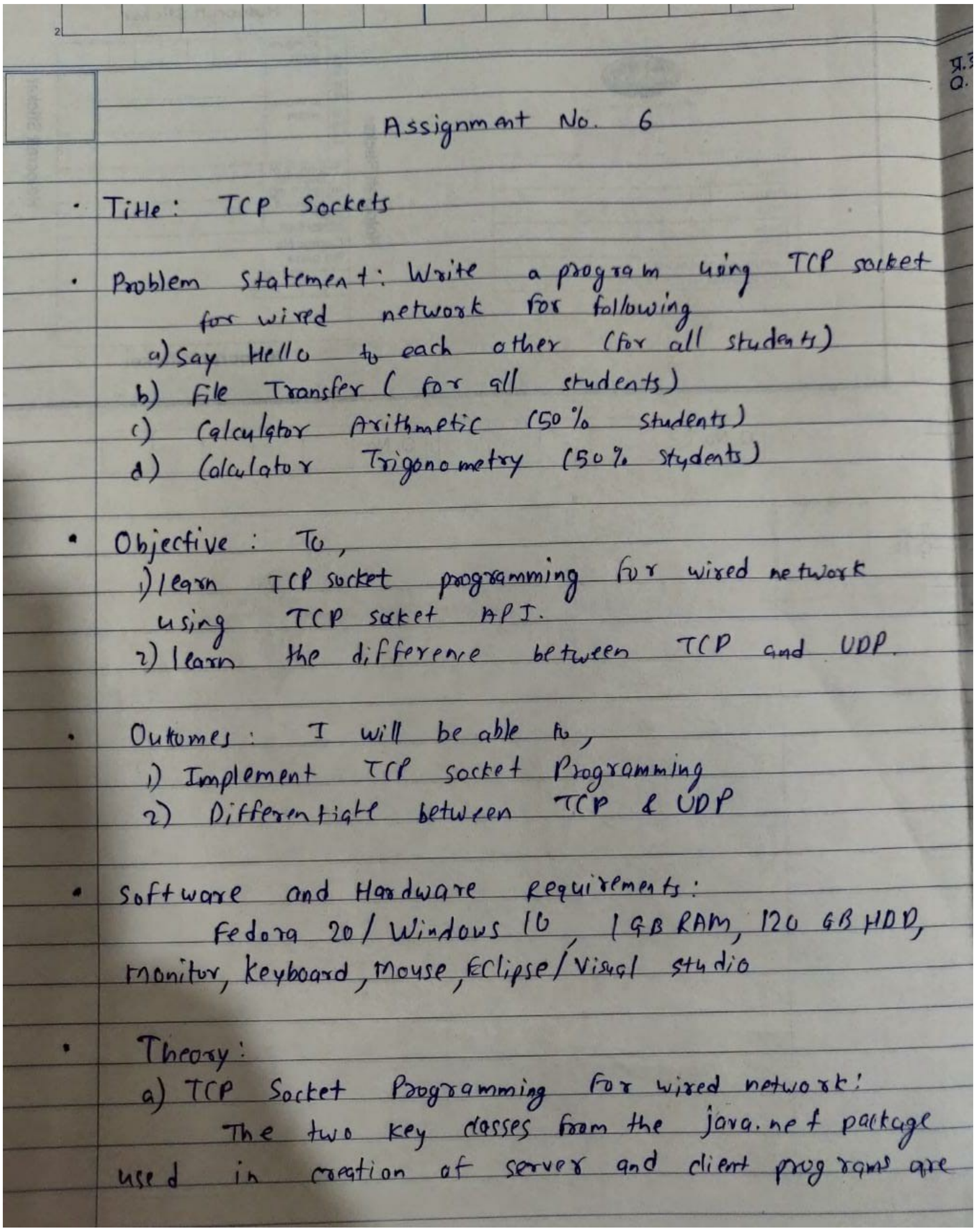
CNL Assignment 6 (TCP Sockets) **(31139-Durvesh)**

// Write-ups



Assignment No. 6

- Title: TCP Sockets

- Problem Statement: Write a program using TCP socket for wired network for following
  - a) Say Hello to each other (for all students)
  - b) File Transfer (for all students)
  - c) Calculator Arithmetic (50% students)
  - d) Calculator Trigonometry (50% students)

- Objective: To,
  1) learn TCP socket programming for wired network using TCP socket API.
  2) learn the difference between TCP and UDP.

- Outcomes: I will be able to,
  1) Implement TCP socket Programming
  2) Differentiate between TCP & UDP

- Software and Hardware requirements:
  Fedora 20 / Windows 10, 1 GB RAM, 120 GB HDD, monitor, keyboard, mouse, Eclipse / Visual studio

- Theory:
  a) TCP Socket Programming for wired network:
  The two key classes from the java.net package used in creation of server and client programs are

ServerSocket & Socket. A server program creates a specific type of socket that is used to listen for client requests. In the case of connection request, the programs creates a new socket through which it will exchange data with the client using input and output streams. The socket abstraction is very similar to the file concept: developers have to open a socket, perform I/O and close it

b) File Transfer:

A TCP client initiates the communication with a server which is waiting for the connection. TCP is connection oriented and UDP is connectionless, which means that UDP Sockets do not need to be connected before being used.

A TCP listener is created and starts listening to the specified port. Again the buffer size is set to 1024 bytes. A TCP listener can ~~preeke~~ precheck to see if there are any connections pending before calling the AcceptTcpClient method. It returns true if there are any pending connections.

- A simple Server Program in Java

The steps for creating a simple server program are:

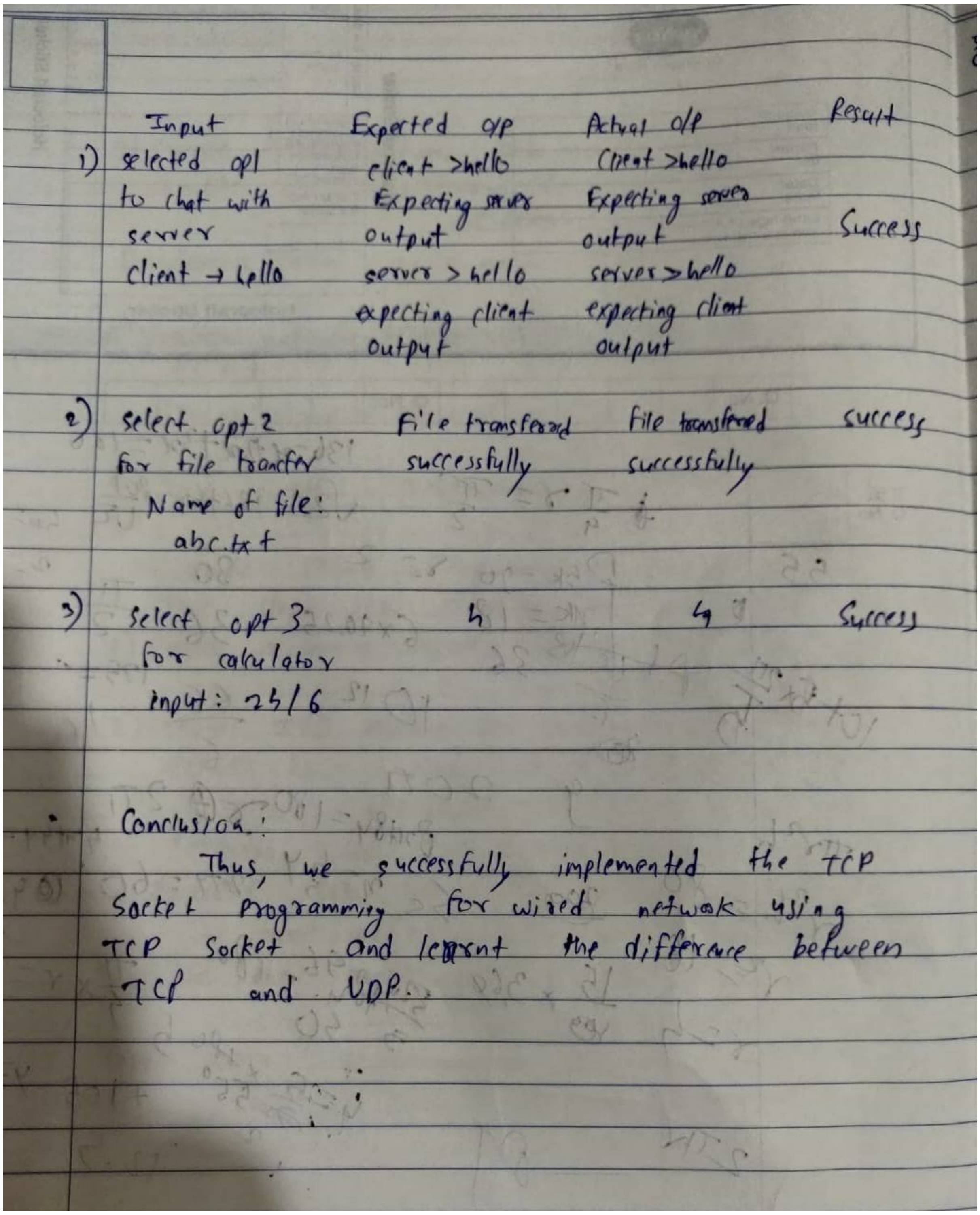
1) Open the Server Socket: ServerSocket server = new ServerSocket(PORT);

2) Wait for the Client Request: Socket client = server.accept();

3) Create I/O streams for communicating to the client:- DataInputStream is = new DataInputStream (client.getInputStream());
DataOutputStream os = new DataOutputStream (client.getOutputStream());
4) Perform communication with client receive from client:
String line = is.readLine(); Send to client : os.writeBytes("Hello\n");
5) Close socket: client.close();

• A simple Client Program in Java
The steps for creating a simple client program are:
1) Create a socket object: Socket client = new Socket(server, port_id);
2) Create I/O streams for communicating with the server is = new DataInputStream (client.getInputStream());
3) Perform I/O or communication with the server. Receive data from the server: String line = is.readLine();
4) Close the socket when done: client.close();

• Test cases:



Input	Expected o/p	Actual o/p	Result
1) selected op1 to chat with server client → hello	client >hello Expecting server output server > hello expecting client output	Client >hello Expecting server output server > hello expecting client output	Success
2) select opt 2 for file transfer Name of file: abc.txt	File transfered successfully	File transfered successfully	success
3) Select opt 3 for calculator input: 25/6	4	4	Success

Conclusion:

Thus, we successfully implemented the TCP Socket Programming for wired netwok using TCP Socket and learnt the differance between TCP and UDP.

// Sample Code

-------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>




void error(const char *msg) {

   perror(msg);

   exit(1);

}



int main(int argc, char const *argv[]) {

   if(argc<2){

      fprintf(stderr,"Port number not provided.\n");

      exit(1);

   }

   int sockfd, newsockfd, portno, n;

   char buffer[255]; // The data to be sent to the server and received from it


   struct sockaddr_in serv_addr, cli_addr;

   socklen_t clilen;


   sockfd = socket(AF_INET, SOCK_STREAM, 0);      //SOCK_STREAM for TCP

   if(sockfd<0){

      error("Error opening the server socket");

   }
```

```
bzero((char *)&serv_addr,sizeof(serv_addr));
portno = atoi(argv[1]);

serv_addr.sin_family = AF_INET;
serv_addr.sin_addr.s_addr = INADDR_ANY;
serv_addr.sin_port = htons(portno);     // Host to network short

if(bind(sockfd,(struct sockaddr *) &serv_addr,sizeof(serv_addr))<0){
   error("Binding Failed.");
}

listen(sockfd, 5);  // Waiting for client to connect
clilen = sizeof(cli_addr);

newsockfd = accept(sockfd,(struct sockaddr *)&cli_addr,&clilen);    // Client accepted

if(newsockfd < 0)
error("Error accepting");
if(portno==9890){       // Chat App from server to client and vice versa
   while(1){
      bzero(buffer,255);
      n = read(newsockfd,buffer,255);     // Reading from client
      if(n<0)
      error("Error on reading");
      printf("Client: %s\n",buffer);
      bzero(buffer,255);
      fgets(buffer,255,stdin);  // Reading from the server terminal
      n = write(newsockfd,buffer,strlen(buffer));     // Sending to CLient
      if(n<0)
      error("Error on writing\n");
      int i = strncmp("bye",buffer,3);
```

```
            if (i==0)

            break;

        }

    }

    if(portno==9891){       // File Transfer

        FILE *fp;

        int ch =0;

        fp=  fopen("receivedtextfile.txt","a");     // Append if the file already exists or else create a new
one

        int words;

        read(newsockfd,&words,sizeof(int));        // Read the word count

        while (ch!=words) {

            read(newsockfd,buffer,255);            // Read a word

            fprintf(fp,"%s ",buffer);              // Write that word to the file

            ch++;

        }

        printf("The file was received");

    }

    if(portno==9892){

        int num1,num2,answer,choice;

        char choices[5][15]={"Addition","Subtraction","Multiplication","Division","Exit"};


S:      n = write(newsockfd,"Enter number 1: ",strlen("Enter number 1: "));       // Sending a message
to ask for num1

        if(n<0)

            error("Error on writing\n");

        read(newsockfd,&num1,sizeof(int));      // Reading num1 sent from client

        printf("Client number 1 is: %d\n",num1);

        n = write(newsockfd,"Enter number 2: ",strlen("Enter number 2: "));        // Sending a message
to ask for num2

        if(n<0)

            error("Error on writing\n");
        read(newsockfd,&num2,sizeof(int));     // Reading num2 sent from client
        printf("Client number 2 is: %d\n",num2);


        n = write(newsockfd,"1. Addition\n2. Subtraction\n3. Multiplication\n4. Division\n5. Exit\n",
                strlen("1. Addition\n2. Subtraction\n3. Multiplication\n4. Division\n5. Exit\n"));//Sending
request for choice
        if(n<0)
            error("Error on writing\n");
        read(newsockfd,&choice,sizeof(int));   // Reading choice for operation
        printf("Client operation is: %s\n",choices[choice-1]);  // Fetching the opearation name from
string array
        switch (choice) {
            case 1:
              answer = num1+num2;    //addition
              break;
            case 2:
              answer = num1-num2;    //Subtraction
              break;
            case 3:
              answer = num1*num2;    //Multiplication
              break;
            case 4:
              answer = num1/num2;    //Division
              break;
            case 5:
              goto Q;              //Exit
              break;


        }
        write(newsockfd,&answer,sizeof(int));
        if(choice!=5){     //Exit Case
            goto S;
```

```
        }



    }
Q:    close(newsockfd);
    close(sockfd);
    return 0;
}
```

```
--------client.c
/*
filename server_ipaddress portno

*/
```

```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>

#include <netdb.h>

#include <ctype.h>



void error(const char *msg) {

   perror(msg);

   exit(1);

}


int main(int argc, char const *argv[]) {

   int sockfd,portno,n;

   struct sockaddr_in serv_addr;

   struct hostent *server;

   char buffer[255];

   if(argc<3){

      fprintf(stderr, "usage %s hostname port\nport 9890 for chat\nport 9891 for file transfer\nport
9892 for calculator\n",argv[0]);

      exit(1);

   }

   portno = atoi(argv[2]);     // String to integer

   sockfd = socket(AF_INET,SOCK_STREAM,0);

   if(sockfd < 0)

   error("Error opening Socket");
```

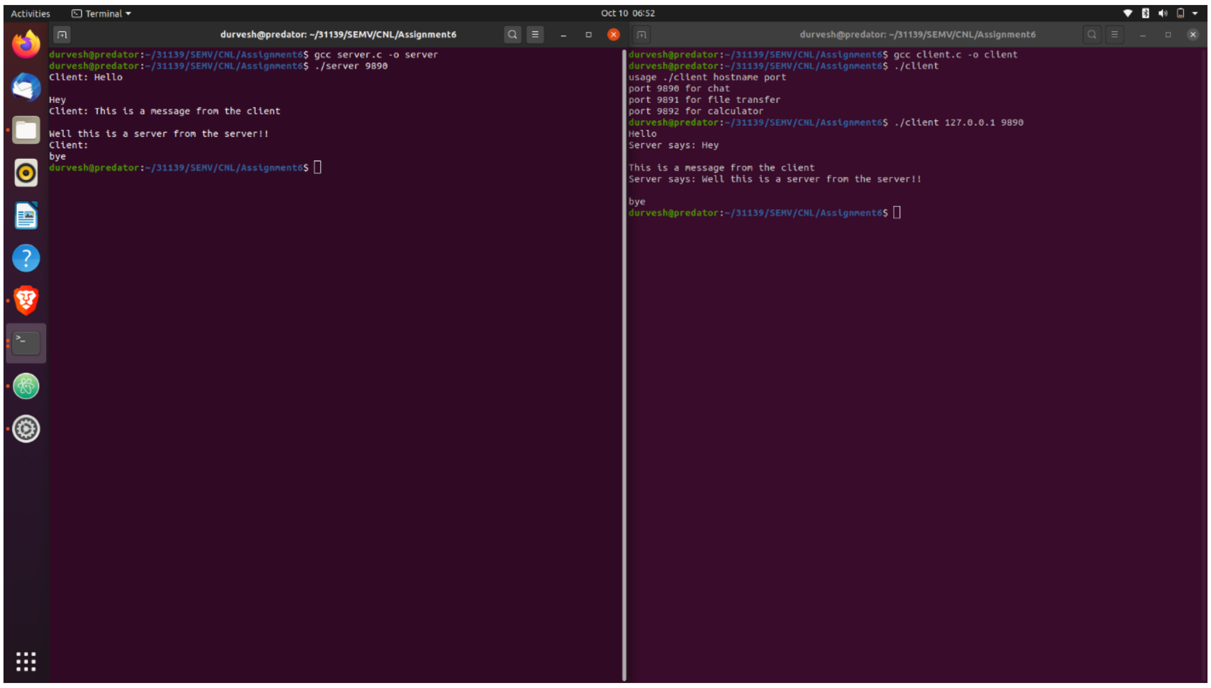
```
server = gethostbyname(argv[1]);
if(server == NULL)
fprintf(stderr,"Error, no such host");
bzero((char *)&serv_addr,sizeof(serv_addr));
serv_addr.sin_family = AF_INET;
bcopy((char*)server->h_addr,(char *)&serv_addr.sin_addr.s_addr,server->h_length);
serv_addr.sin_port = htons(portno);   // host to network short
if(connect(sockfd,(struct sockaddr *)&serv_addr,sizeof(serv_addr))<0){      // Connecting to Server
   error("Connection failed");
}
if(portno==9890){          // Chat App
   while(1){
      bzero(buffer,255);
      fgets(buffer,255,stdin);    // Reading from client terminal
      int j = strncmp("bye",buffer,3);
      if (j==0)
      break;
      n = write(sockfd,buffer,strlen(buffer));       // Sending to server
      if(n<0)
      error("Error on writing");
      bzero(buffer,255);
      n = read(sockfd,buffer,255);                  // Reading from server
      if(n<0)
      error("Error on reading");
      printf("Server says: %s\n",buffer);            // Printing server's sent message
      int i = strncmp("bye",buffer,3);
      if (i==0)
      break;
   }
}
if(portno==9891){      // File Transfer
```

```
    FILE *f;
    int words = 0;
    char c;
    f = fopen("textfile.txt","r");     // Opening an already existing file to send its data to server
    while((c= getc(f))!= EOF){          // Counting the number of words
        fscanf(f,"%s",buffer);
        if(isspace(c)||c=='\t')
        words++;
    }
    write(sockfd,&words,sizeof(int));      // Sending the number of words
    rewind(f);          // Setting the file pointer at initial word


    char ch;
    while(ch!=EOF){
        fscanf(f,"%s",buffer);
        write(sockfd,buffer,255);      // Sending words one by one
        ch = fgetc(f);
    }
    printf("The file has been sent.\n");


}
if(portno==9892){
    while(1){
        int num1,num2,choice,answer;
        bzero(buffer,255);
        n = read(sockfd,buffer,255);           // Reading first request from server
        if(n<0)
            error("Error reading");
        printf("Server- %s",buffer);           // printing first request
        scanf("%d", &num1);                    // getting num1 from client terminal
        n = write(sockfd,&num1,sizeof(int));       // sending num1 to server
```

```
if (n<0) {
   error("Error writing\n");
}
bzero(buffer,255);
n = read(sockfd,buffer,255);           // Reading first request from server
if(n<0)
error("Error reading");
printf("Server- %s",buffer);           // printing first request
scanf("%d", &num2);                     // getting num1 from client terminal
n = write(sockfd,&num2,sizeof(int));       // sending num2 to server
if (n<0) {
   error("Error writing\n");
}
bzero(buffer,255);
n = read(sockfd,buffer,255);
if(n<0)
error("Error reading");
printf("Server- %s\n",buffer);
scanf("%d", &choice);              // Reading operation choice from client

n = write(sockfd,&choice,sizeof(int));
if (n<0) {
   error("Error writing\n");
}
if(choice==5){      // Exit case
   goto E;
   break;
}
printf("Answer: ");
n = read(sockfd,&answer,sizeof(int));       // Getting the answer.
printf("%d\n",answer );            // printing the answer
```

```
        }
    }

    E:  close(sockfd);
    return 0;
}
```

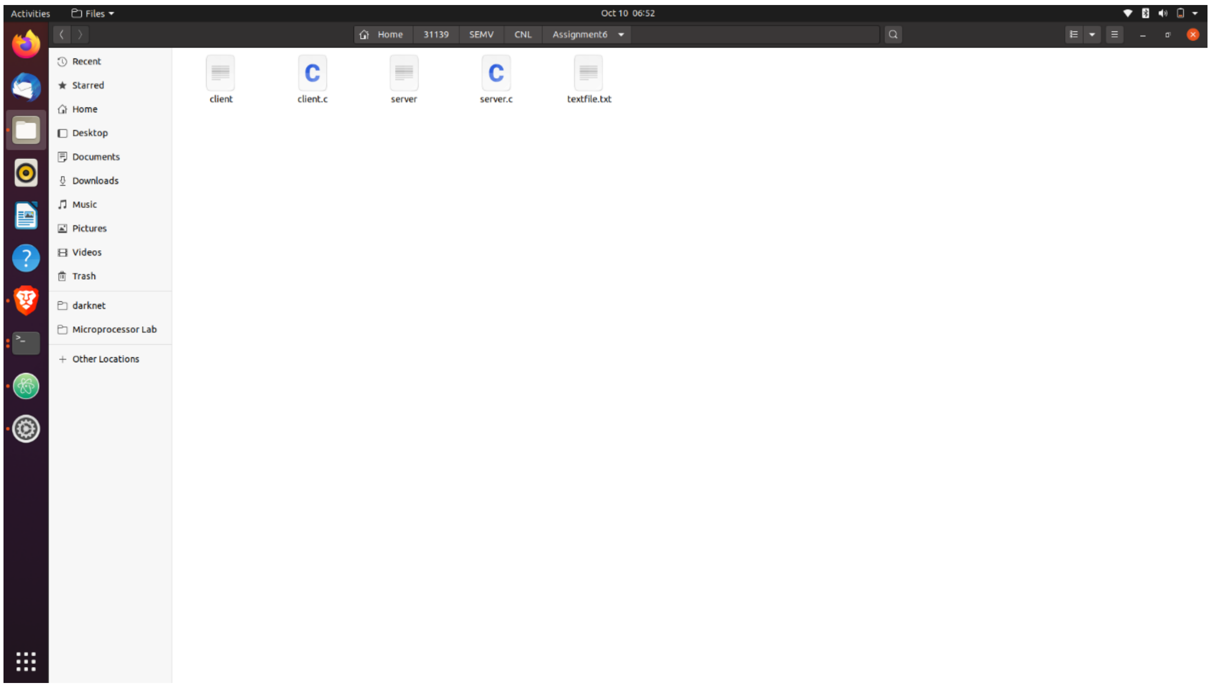
// Outputs

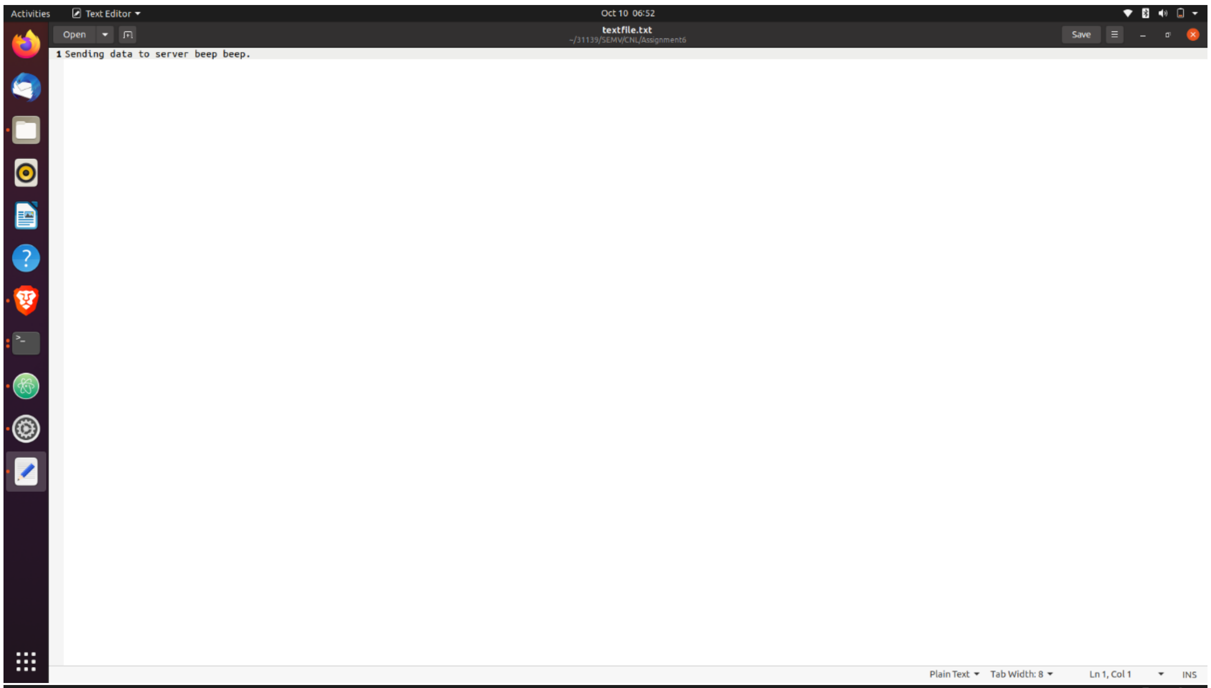


Activities
Terminal
Oct 10 06:52
durvesh@predator: ~/31139/SEMV/CNL/Assignment6
durvesh@predator: ~/31139/SEMV/CNL/Assignment6

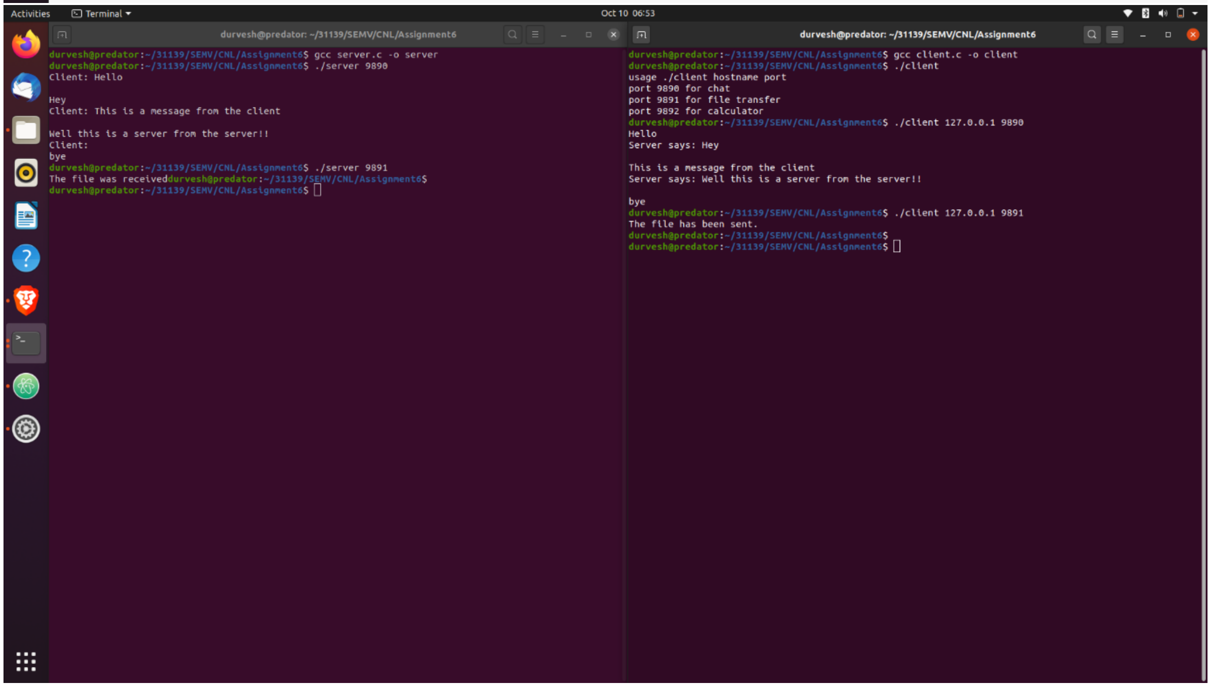
```
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ gcc server.c -o server
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./server 9890
Client: Hello

Hey
Client: This is a message from the client

Well this is a server from the server!!
Client:
bye
durvesh@predator:~/31139/SEMV/CNL/Assignment6$
```

```
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ gcc client.c -o client
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./client
usage ./client hostname port
port 9890 for chat
port 9891 for file transfer
port 9892 for calculator
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./client 127.0.0.1 9890
Hello
Server says: Hey

This is a message from the client
Server says: Well this is a server from the server!!

bye
durvesh@predator:~/31139/SEMV/CNL/Assignment6$
```


Activities
Files
Oct 10 06:52
Home
31139
SEMV
CNL
Assignment6
Recent
Starred
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
darknet
Microprocessor Lab
Other Locations
client
client.c
server
server.c
textfile.txt

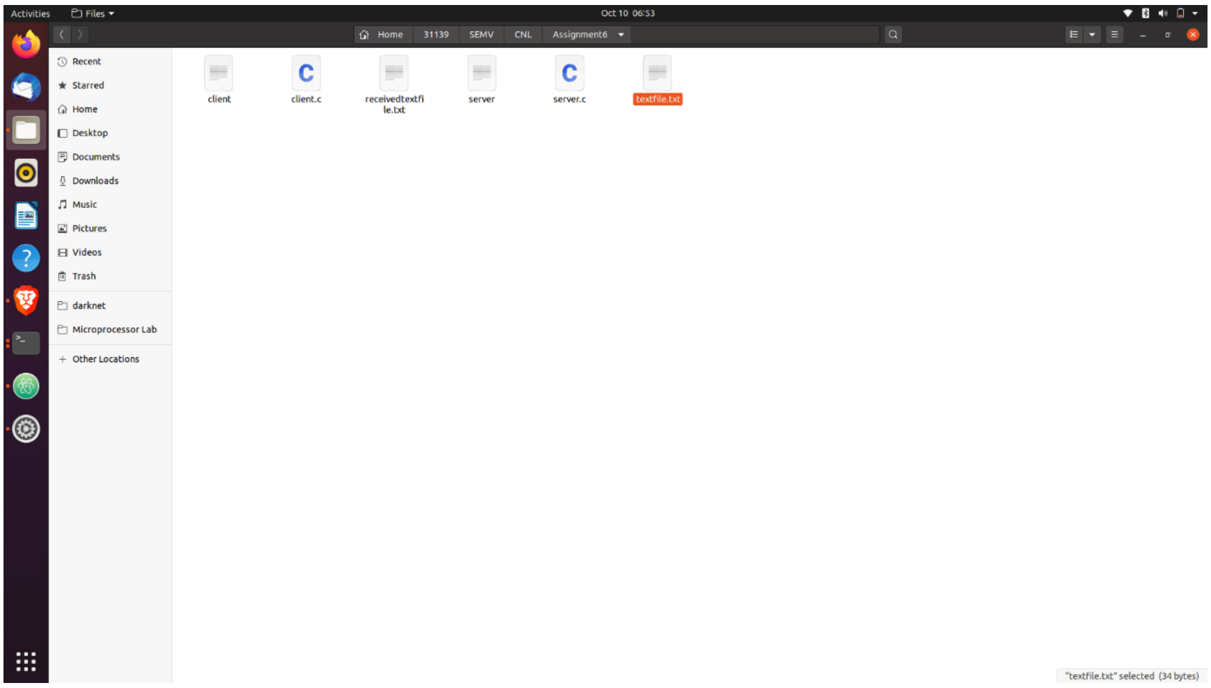


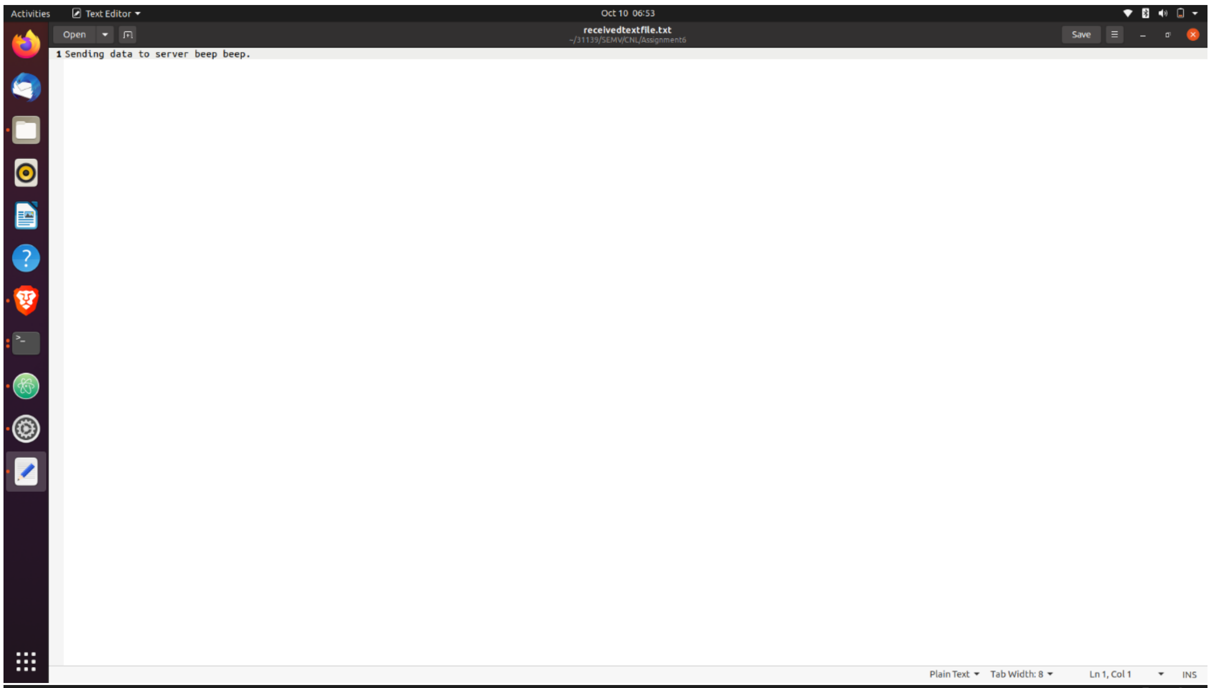
```
Sending data to server beep beep.
```


Activities
Terminal
Oct 10 06:53
durvesh@predator: ~/31139/SEMV/CNL/Assignment6
durvesh@predator: ~/31139/SEMV/CNL/Assignment6

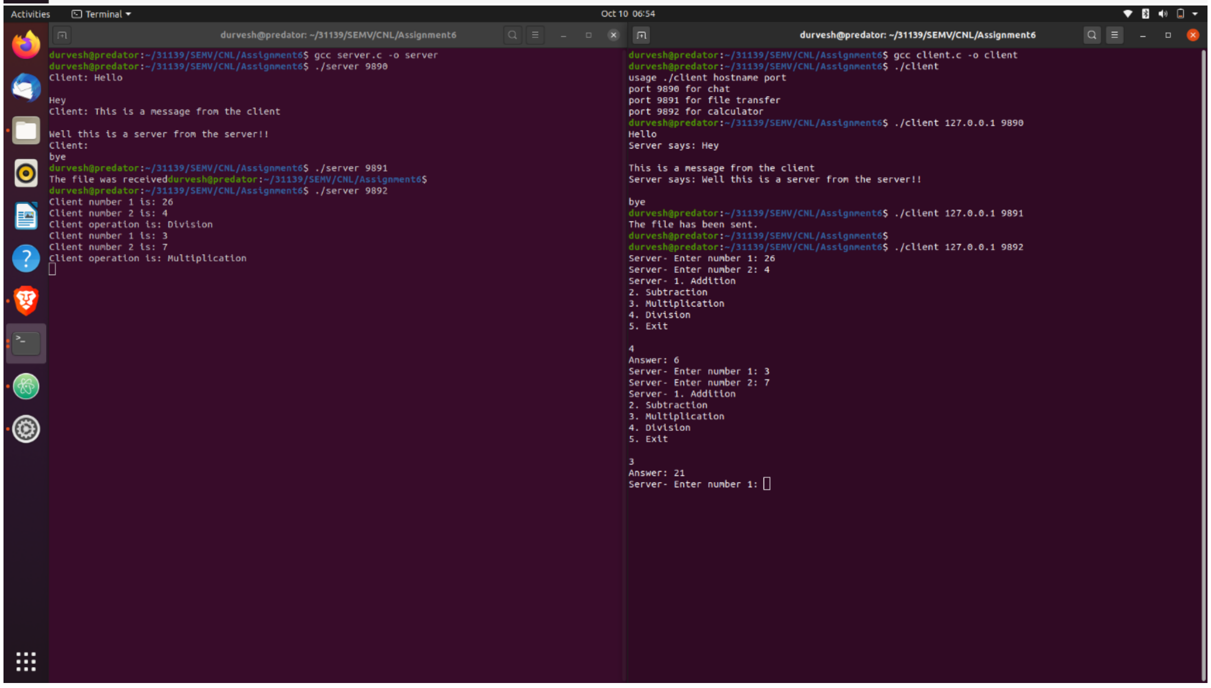
```
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ gcc server.c -o server
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./server 9890
Client: Hello

Hey
Client: This is a message from the client

Well this is a server from the server!!
Client:
bye
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./server 9891
The file was receiveddurvesh@predator:~/31139/SEMV/CNL/Assignment6$
durvesh@predator:~/31139/SEMV/CNL/Assignment6$
```

```
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ gcc client.c -o client
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./client
usage ./client hostname port
port 9890 for chat
port 9891 for file transfer
port 9892 for calculator
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./client 127.0.0.1 9890
Hello
Server says: Hey

This is a message from the client
Server says: Well this is a server from the server!!

bye
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./client 127.0.0.1 9891
The file has been sent.
durvesh@predator:~/31139/SEMV/CNL/Assignment6$
durvesh@predator:~/31139/SEMV/CNL/Assignment6$
```


Activities
Files
Oct 10 06:53
Home
31139
SEMV
CNL
Assignment6
Recent
Starred
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
darknet
Microprocessor Lab
Other Locations
client
client.c
receivedtextfile.txt
server
server.c
textfile.txt
"textfile.txt" selected (34 bytes)



```
Sending data to server beep beep.
```


Activities
Terminal
Oct 10 06:54
durvesh@predator: ~/31139/SEMV/CNL/Assignment6
durvesh@predator: ~/31139/SEMV/CNL/Assignment6

```
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ gcc server.c -o server
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./server 9890
Client: Hello

Hey
Client: This is a message from the client

Well this is a server from the server!!
Client:
bye
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./server 9891
The file was receiveddurvesh@predator:~/31139/SEMV/CNL/Assignment6$
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./server 9892
Client number 1 is: 26
Client number 2 is: 4
Client operation is: Division
Client number 1 is: 3
Client number 2 is: 7
Client operation is: Multiplication
```

```
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ gcc client.c -o client
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./client
usage ./client hostname port
port 9890 for chat
port 9891 for file transfer
port 9892 for calculator
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./client 127.0.0.1 9890
Hello
Server says: Hey

This is a message from the client
Server says: Well this is a server from the server!!

bye
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./client 127.0.0.1 9891
The file has been sent.
durvesh@predator:~/31139/SEMV/CNL/Assignment6$
durvesh@predator:~/31139/SEMV/CNL/Assignment6$ ./client 127.0.0.1 9892
Server- Enter number 1: 26
Server- Enter number 2: 4
Server- 1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit

4
Answer: 6
Server- Enter number 1: 3
Server- Enter number 2: 7
Server- 1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit

3
Answer: 21
Server- Enter number 1: 
```