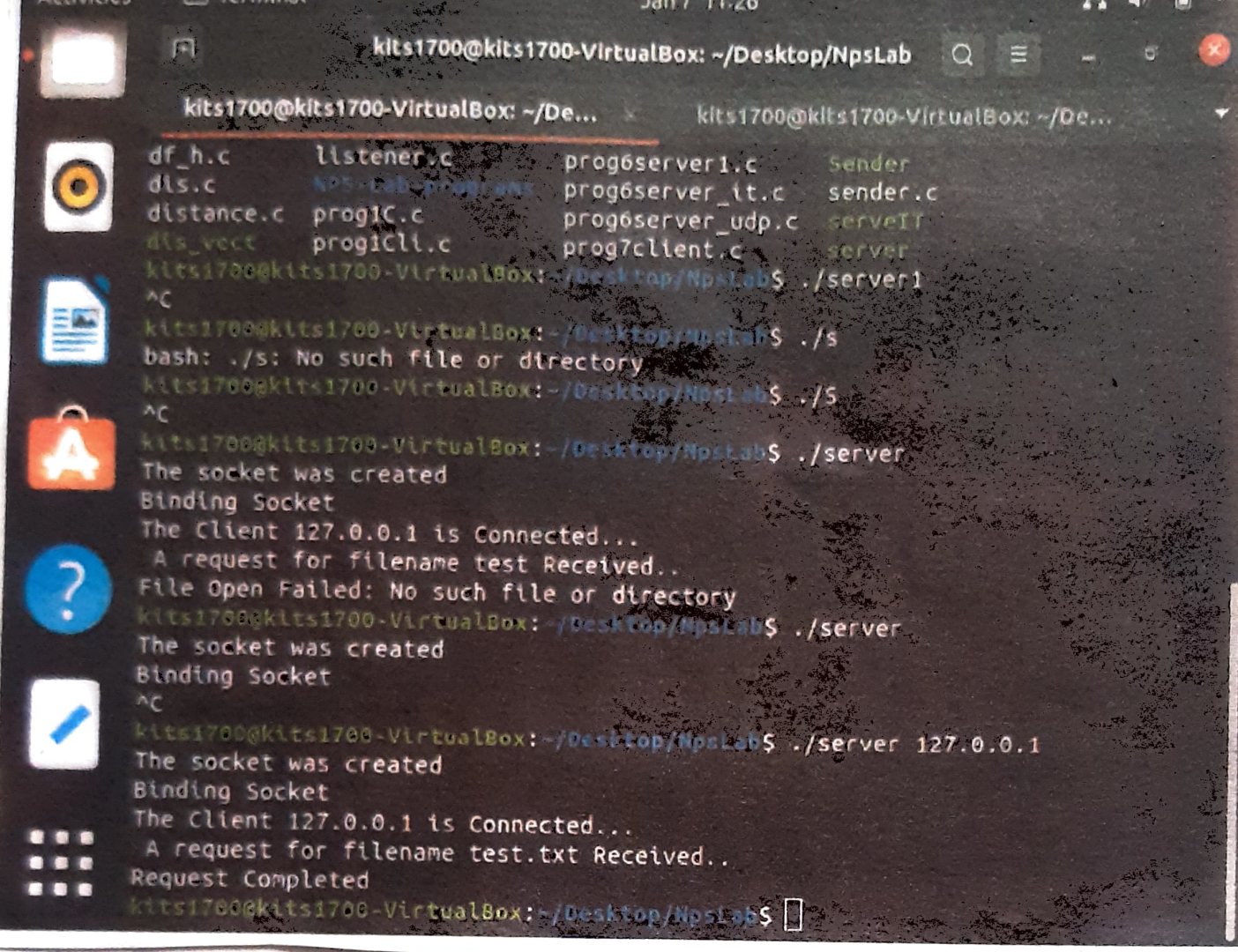
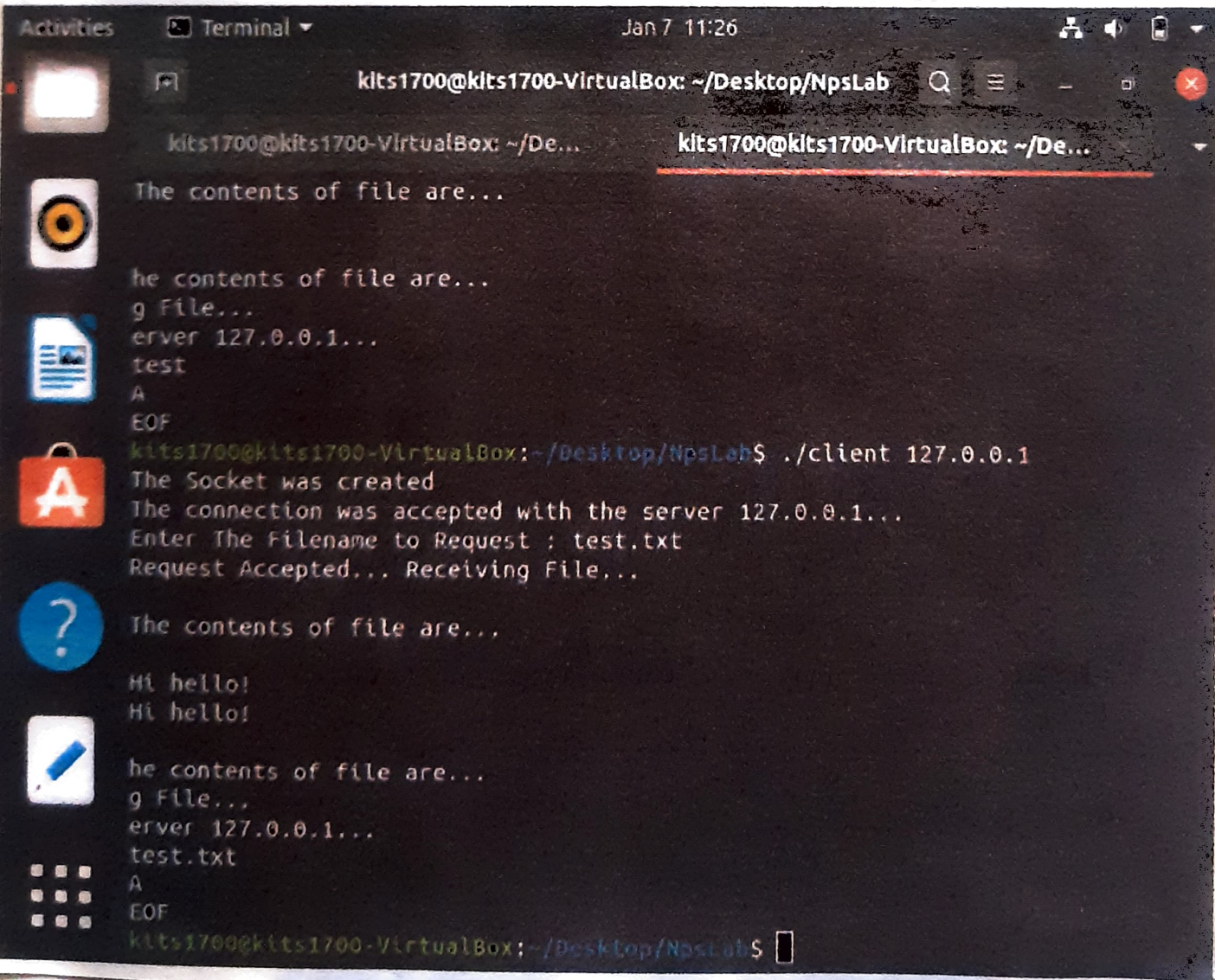

kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kits1700-VirtualBox: ~/De...
df_h.c listener.c prog6server1.c Sender
dis.c prog6server_it.c sender.c
distance.c prog1C.c prog6server_udp.c serveIT
prog1Cli.c prog7client.c server
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./server1
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./s
bash: ./s: No such file or directory
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./s
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./server
The socket was created
Binding Socket
The Client 127.0.0.1 is Connected...
A request for filename test Received..
File Open Failed: No such file or directory
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./server
The socket was created
Binding Socket
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./server 127.0.0.1
The socket was created
Binding Socket
The Client 127.0.0.1 is Connected...
A request for filename test.txt Received..
Request Completed
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$


Activities
Terminal
Jan 7 11:26
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kits1700-VirtualBox: ~/De...
kits1700@kits1700-VirtualBox: ~/De...
The contents of file are...
he contents of file are...
g File...
erver 127.0.0.1...
test
A
EOF
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./client 127.0.0.1
The Socket was created
The connection was accepted with the server 127.0.0.1...
Enter The Filename to Request : test.txt
Request Accepted... Receiving File...
The contents of file are...
Hi hello!
Hi hello!
he contents of file are...
g File...
erver 127.0.0.1...
test.txt
A
EOF
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$

EXPT NO | NAME | Page No.: | Date: | YOUVA

M T W T F S S

1) Implement a client and server communication using socket programming.

// Server

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

int main()
{ int cont, create_socket, new_socket, addrlen, fd;
  int bufsize = 1024;
  char* buffer = malloc(bufsize);
  char fname[256];
  struct sockaddr_in address;
  if((create_socket = socket(AF_INET, SOCK_STREAM, 0))>0)
          printf("The socket was created\n");
  address.sin_family = AF_INET;
  address.sin_addr.s_addr = INADDR_ANY;
  address.sin_port = htons(15000);
  if(bind(create_socket, (struct sockaddr *)&address, sizeof(address))==0
          printf("Binding Socket\n");
  listen(create_socket, 3);
  addrlen = sizeof(struct sockaddr_in);
```

Teacher's Signature:

EXPT NO | NAME | Page No | Date | YOUVA

```
new_socket = accept (create_socket, (struct sockaddr*)&address,
                                        &addrlen);
if (new_socket > 0)
    printf ("The client %d is Connected...\n", inet_ntoa(address.
                                                    sin_addr));
recv(new_socket, fname, 255, 0);
printf("A request for filename: %s Received...\n", fname);
if((fd = open(fname, O_RDONLY)) < 0)
{   perror("File Open failed"); exit(0); }
    while((cont = read(fd, buffer, bufsize)) > 0)
        { send(new_socket, buffer, cont, 0); }
printf("Request completed \n");
close(new_socket);
return close(create_socket);
}
```

//Client

```
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <arpa/inet.h>
int main(int argc, char* argv[])
{   int create_socket;
    int bufsize = 1024;
char *buffer = malloc(bufsize);
char fname[256];
```

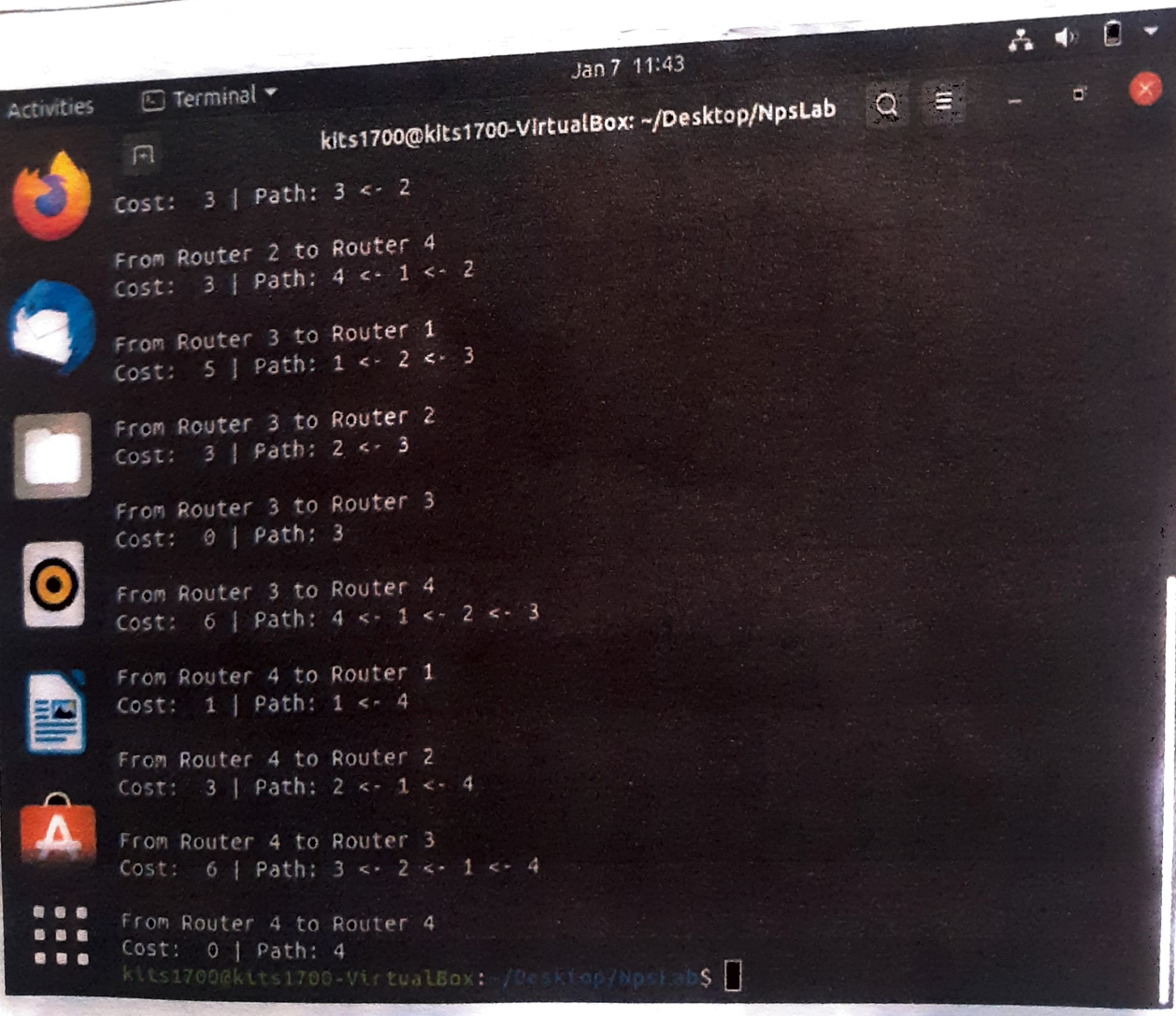
Teacher's Signature: _____

EXPT NO. | NAME | Page No.: | Date: | YOUVA

```
struct sockaddr_in address;
if(Create_socket = socket(AF_INET, SOCK_STREAM, 0)>0)
        printf("The socket was created\n");
address.sin_family = AF_INET;
address.sin_port = htons(15000);
inet_pton(AF_INET, argv[1], &address.sin_addr);
if(connect(create_socket, (struct sockaddr *) &address, sizeof(address))
                                                    ==0)
        printf("The connection was accepted with the
                server:%s...\n", argv[1]);
printf("Enter the filename to request:");
scanf("%s", fname);
send(create_socket, fname, sizeof(fname), 0);
printf("Request Accepted... Receiving File...\n\n");
printf("The contents of file are...\n\n");
Uint32_t cont;
while((cont = recv(create_socket, buffer, bufsize, 0))>0)
{   write(1, buffer, cont);
}
printf("\nEOF\n");
return close(create_socket);
}
```

Teacher's Signature:


Activities
Terminal
Jan 7 11:44
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab$ ./dis_vect
N: 4
Matrix:
0 2 999 1
2 0 3 7
999 3 0 11
1 7 11 0
From Router 1 to Router 1
Cost: 0 | Path: 1
From Router 1 to Router 2
Cost: 2 | Path: 2 <- 1
From Router 1 to Router 3
Cost: 5 | Path: 3 <- 2 <- 1
From Router 1 to Router 4
Cost: 1 | Path: 4 <- 1
From Router 2 to Router 1
Cost: 2 | Path: 1 <- 2
From Router 2 to Router 2
Cost: 0 | Path: 2
From Router 2 to Router 3
Cost: 3 | Path: 3 <- 2


Activities
Terminal
Jan 7 11:43
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
Cost: 3 | Path: 3 <- 2
From Router 2 to Router 4
Cost: 3 | Path: 4 <- 1 <- 2
From Router 3 to Router 1
Cost: 5 | Path: 1 <- 2 <- 3
From Router 3 to Router 2
Cost: 3 | Path: 2 <- 3
From Router 3 to Router 3
Cost: 0 | Path: 3
From Router 3 to Router 4
Cost: 6 | Path: 4 <- 1 <- 2 <- 3
From Router 4 to Router 1
Cost: 1 | Path: 1 <- 4
From Router 4 to Router 2
Cost: 3 | Path: 2 <- 1 <- 4
From Router 4 to Router 3
Cost: 6 | Path: 3 <- 2 <- 1 <- 4
From Router 4 to Router 4
Cost: 0 | Path: 4
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab$

EXPT. NO | NAME | Page No. | Date | YOUVA

2) Write a program to implement distance vector routing protocol for a simple topology of routers.

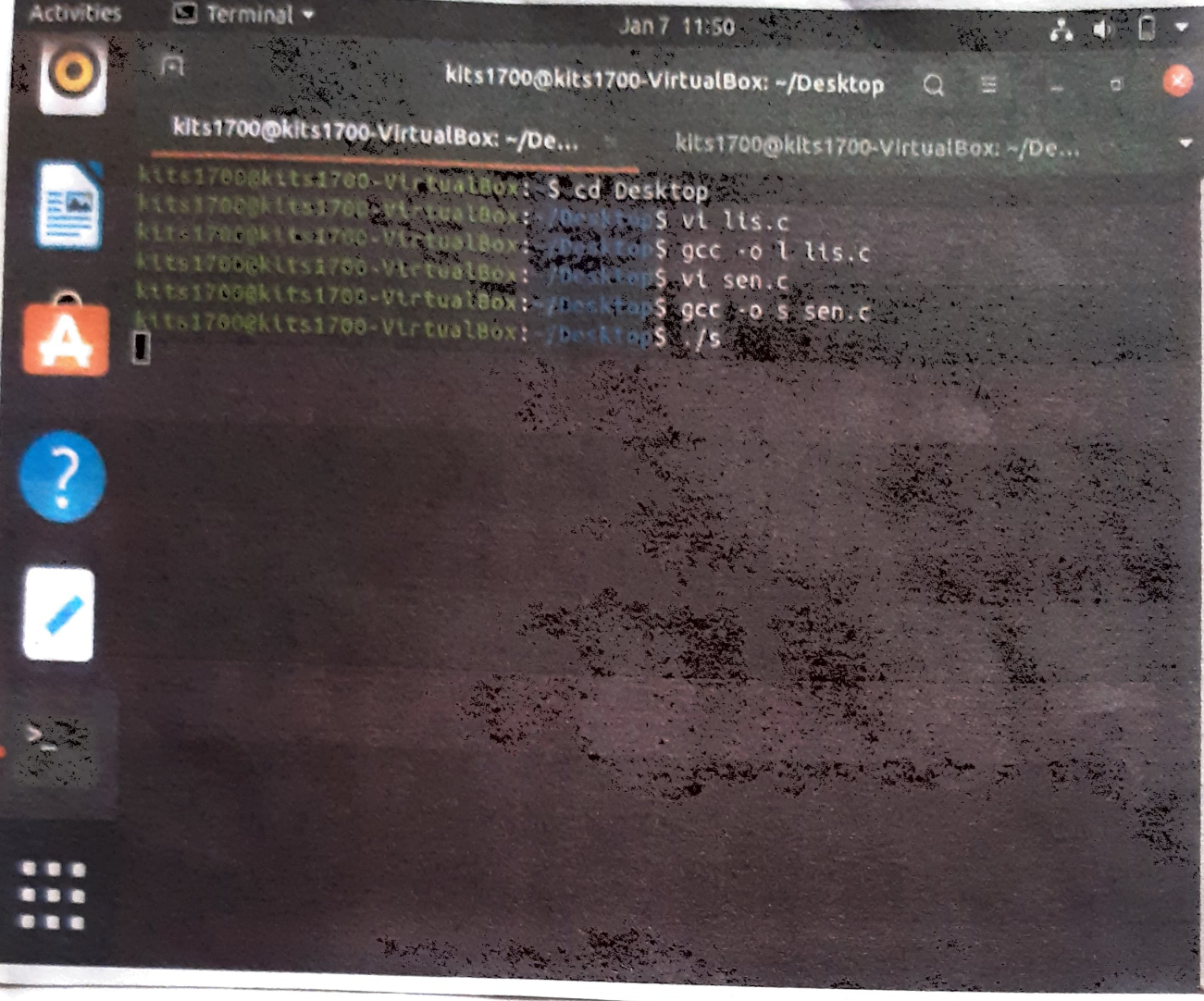
```
#include<stdio.h>
int A[10][10], n, d[10], p[10];
void BellmanFord(int &)
{
  int i, u, v;
  for(i=1; i<n; i++)
  { for(u=0; u<n; u++) {
      for(v=0; v<n; v++)
        { if (d[v] > d[u] + A[u][v])
            { d[v] = d[u] + A[u][v];
              p[v] = u; }
        }
    }
}

int main()
{ printf("Enter the no. of vertices : ");
  scanf("%d", &n);
  printf("Enter the adjacency matrix \n");
  int i, j;
  for(i=0; i<n; i++)
    for(j=0; j<n; j++)
      scanf("%d", &A[i][j]);
  int source;
  for(source=0; source<n; source++)
  { for(i=0; i<n; i++){
      d[i]=999; p[i]=-1; }
```

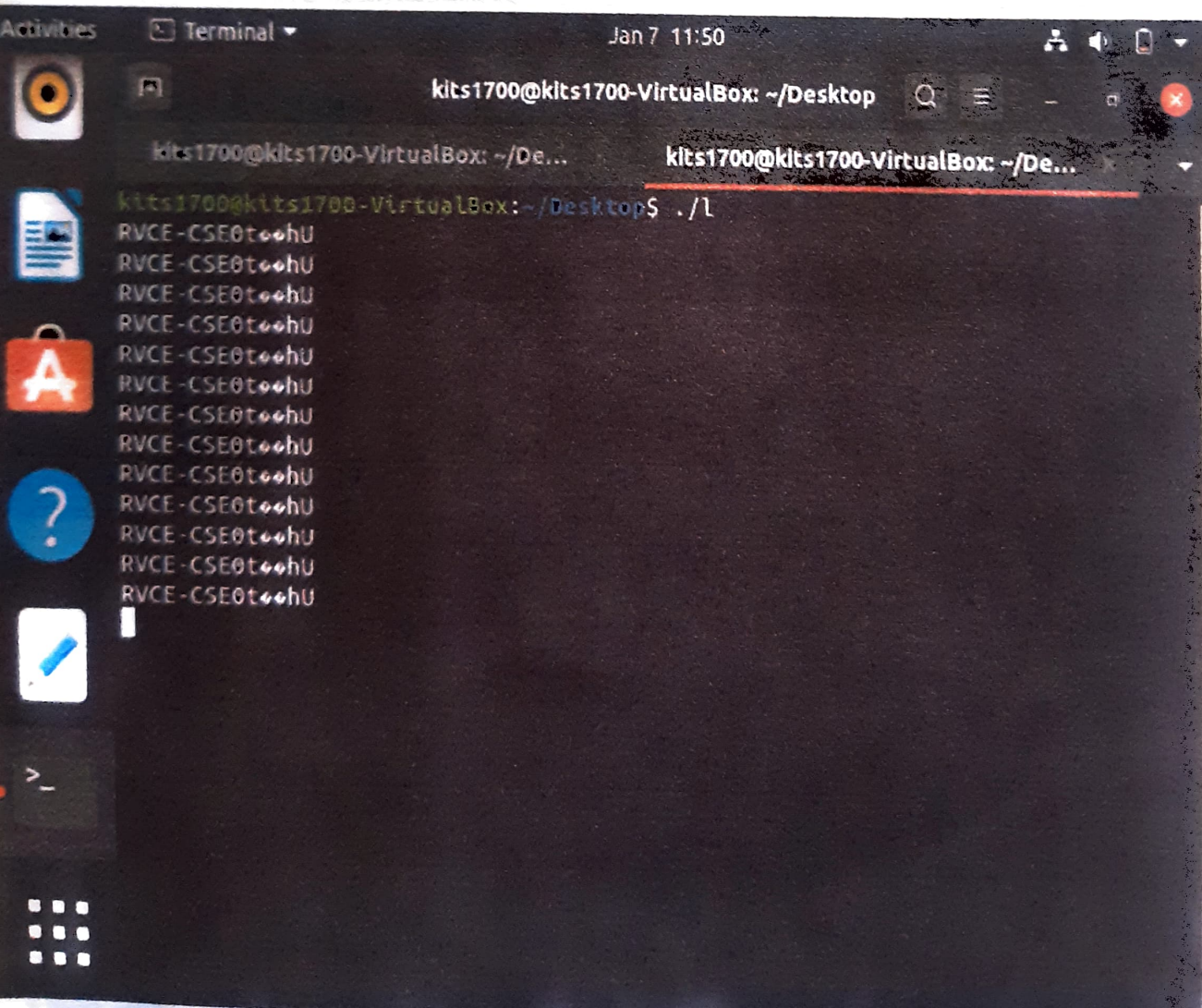
Teacher's Signature:

EXPT. NO. | NAME | Page No.: | Date: | YOUVA

```
d[source] = 0;
BellmanFord(source);
printf("Router %d\n", source);
for(i=0; i<n; i++)
if(i != source)
{ j=i;
  while(p[j] != -1)
  { printf("%d<-", j);
    j = p[j];
  }
printf("%d \t cost %d\n\n", source, d[i]);
}
```

Teacher's Signature:


Activities
Terminal
Jan 7 11:50
kits1700@kits1700-VirtualBox: ~/Desktop
kits1700@kits1700-VirtualBox: ~/De...
kits1700@kits1700-VirtualBox: ~/De...
kits1700@kits1700-VirtualBox:~/Desktop$ cd Desktop
kits1700@kits1700-VirtualBox:~/Desktop$ vi lis.c
kits1700@kits1700-VirtualBox:~/Desktop$ gcc -o l lis.c
kits1700@kits1700-VirtualBox:~/Desktop$ vi sen.c
kits1700@kits1700-VirtualBox:~/Desktop$ gcc -o s sen.c
kits1700@kits1700-VirtualBox:~/Desktop$ ./s


Activities
Terminal
Jan 7 11:50
kits1700@kits1700-VirtualBox: ~/Desktop
kits1700@kits1700-VirtualBox: ~/De...
kits1700@kits1700-VirtualBox: ~/De...
kits1700@kits1700-VirtualBox:~/Desktop$ ./l
RVCE-CSE0toohU
RVCE-CSE0toohU
RVCE-CSE0toohU
RVCE-CSE0toohU
RVCE-CSE0toohU
RVCE-CSE0toohU
RVCE-CSE0toohU
RVCE-CSE0toohU
RVCE-CSE0toohU
RVCE-CSE0toohU
RVCE-CSE0toohU
RVCE-CSE0toohU
RVCE-CSE0toohU

EXPT NO | NAME | Page No: | Date | YOUVA

M T W T F S S

4) Implement a simple multicast routing mechanism

// Listener

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<time.h>
#include<string.h>
#include<stdio.h>
#include<stdlib.h>
#define HELLO_PORT 12345
#define HELLO_GROUP "225.0.0.37"
#define MSGBUFSIZE 25
int main(int argc, char* argv[])
{ struct sockaddr_in addr;
  int fd, nbytes, addrlen;
  struct ip_mreq mreq;
  char msgbuf[MSGBUFSIZE];
  u_int yes=1;
if((fd=socket(AF_INET,SOCK_DGRAM,0))<0)
  { perror("Socket");
    exit(1); }
if(setsockopt(fd, SOL_SOCKET, SO_REUSEADDR, &yes, sizeof(yes))<0)
  { perror("Reusing ADDR failed");
    exit(1); }
memset(&addr, 0, sizeof(addr));
addr.sin_family = AF_INET;
addr.sin_addr.s_addr = htonl(INADDR_ANY);
```

Teacher's Signature:

```
addr.sin_port = htons (HELLO_PORT);
if(bind(fd,(struct sockaddr *) &addr, sizeof(addr))<0)
{ perror("bind");
    exit(1);}
mreq.imr_multiaddr.s_addr = inet_addr(HELLO_GROUP);
mreq.imr_interface.s_addr = htonl(INADDR_ANY);
if(setsockopt(fd, IPPROTO_IP, IP_ADD_MEMBERSHIP, &mreq, sizeof(mreq))<0)
  {    perror("setsockopt");
          exit(1);}
while(1)
{addrlen = sizeof(addr);
  if((nbytes =recvfrom(fd, msgbuf, MSGBUFSIZE, 0, (struct sockaddr*)&addr,
                    &addrlen))<0)
    { perror("recvfrom"); }
    puts(msgbuf); }
                  }
```

//Sender

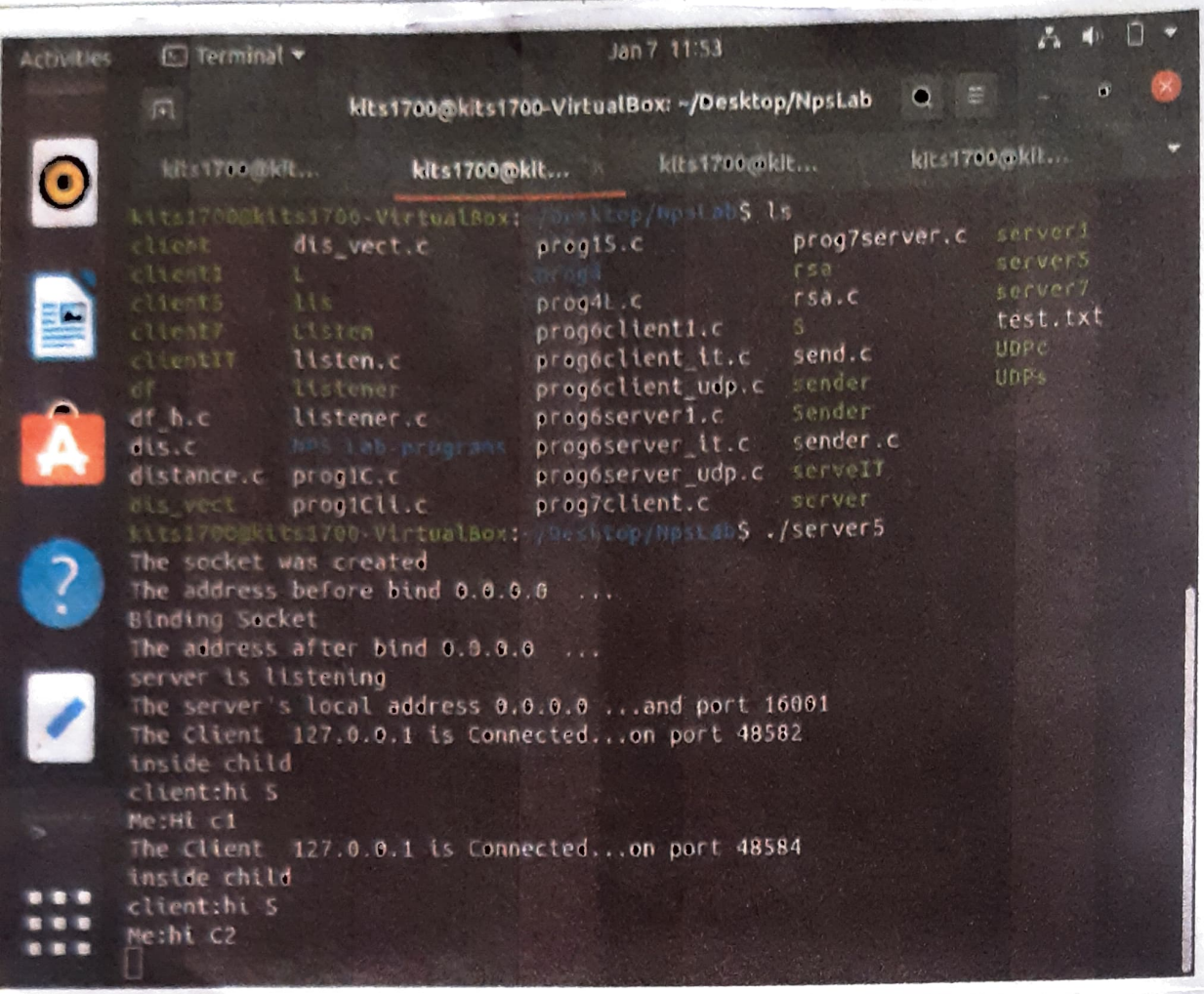
```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<time.h>
#include<string.h>
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#define HELLOPORT 12345
```

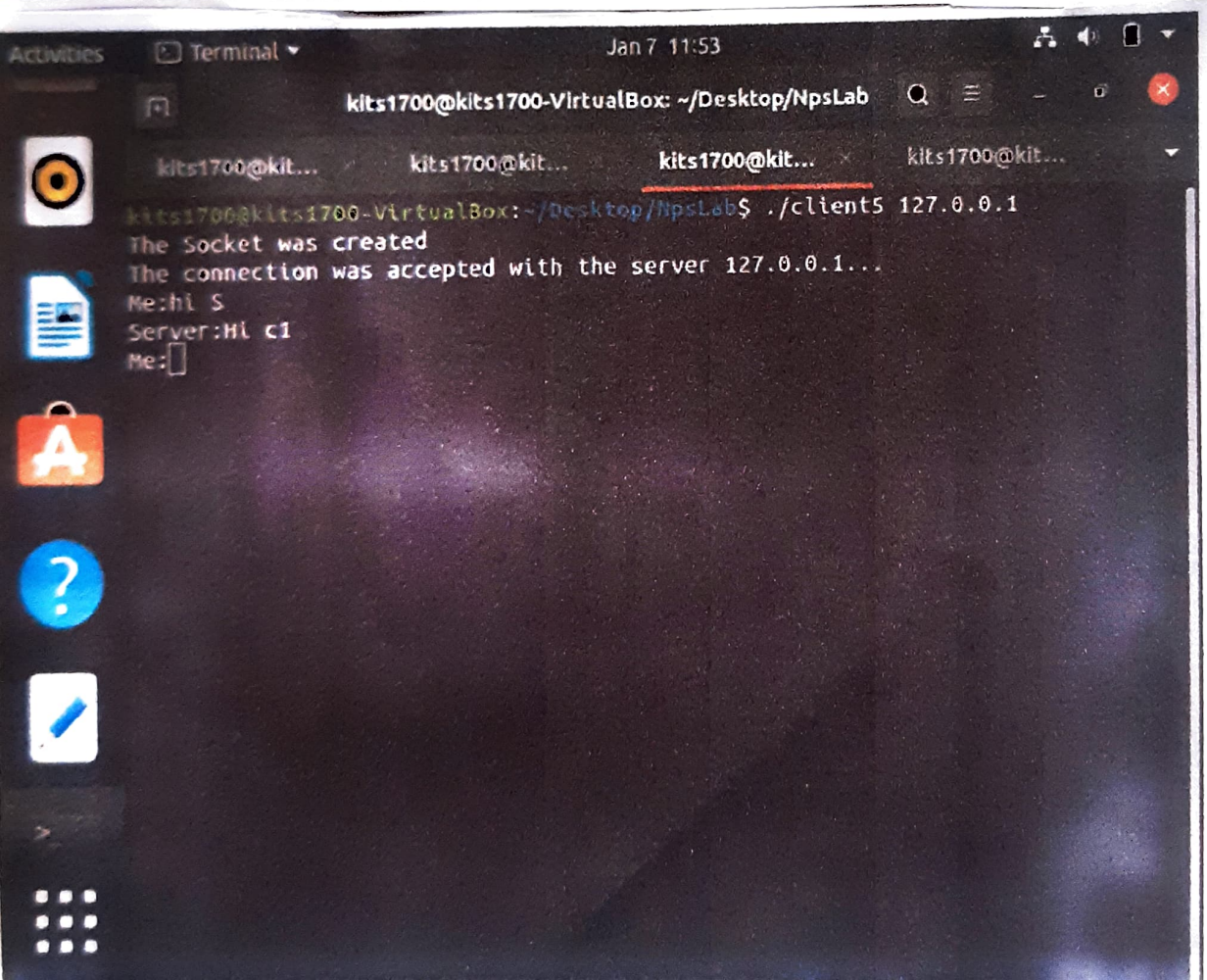
Teacher's Signature:

EXPT NO | NAME | Page No. | Date | YOUVA

```
#define HELLO_GROUP "225.0.0.37"
int main (int argc, char *argv[])
{ struct ip_mreq mreq;
  int fd, cnt;
char* message = "RVCE-CSE";
if ((fd= socket (AF_INET,SOCK_DGRAM,0)) <0)
      { perror("socket");
        exit(1); }
memset (&addr, 0 ,sizeof (addr));
addr.sin_family = AF_INET;
addr.sin_addr.s_addr = inet_addr(HELLO_GROUP);
addr.sin_port = htons(HELLO_PORT);
while(1)
{ if (sendto( fd, message ,sizeof(message), 0, (struct sockaddr *)&addr,
            sizeof (addr)) <0)
   { perror ("sendto");
     exit(1);
   sleep(1);
   }
}
```

Teacher's Signature: _____


Activities
Terminal
Jan 7 11:53
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ls
client dis_vect.c prog1S.c prog7server.c server1
client1 i prog4 rsa server5
client5 lis prog4L.c rsa.c server7
client7 listen prog6client1.c S test.txt
clientIT listen.c prog6client_it.c send.c UDPc
df listener prog6client_udp.c sender UDPs
df_h.c listener.c prog6server1.c Sender
dis.c NPS Lab programs prog6server_it.c sender.c
distance.c prog1C.c prog6server_udp.c serveIT
dis_vect prog1Cli.c prog7client.c server
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./server5
The socket was created
The address before bind 0.0.0.0 ...
Binding Socket
The address after bind 0.0.0.0 ...
server is listening
The server's local address 0.0.0.0 ...and port 16001
The Client 127.0.0.1 is Connected...on port 48582
inside child
client:hi S
Me:Hi c1
The Client 127.0.0.1 is Connected...on port 48584
inside child
client:hi S
Me:hi C2


Activities
Terminal
Jan 7 11:53
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./client5 127.0.0.1
The Socket was created
The connection was accepted with the server 127.0.0.1...
Me:hi S
Server:Hi c1
Me:

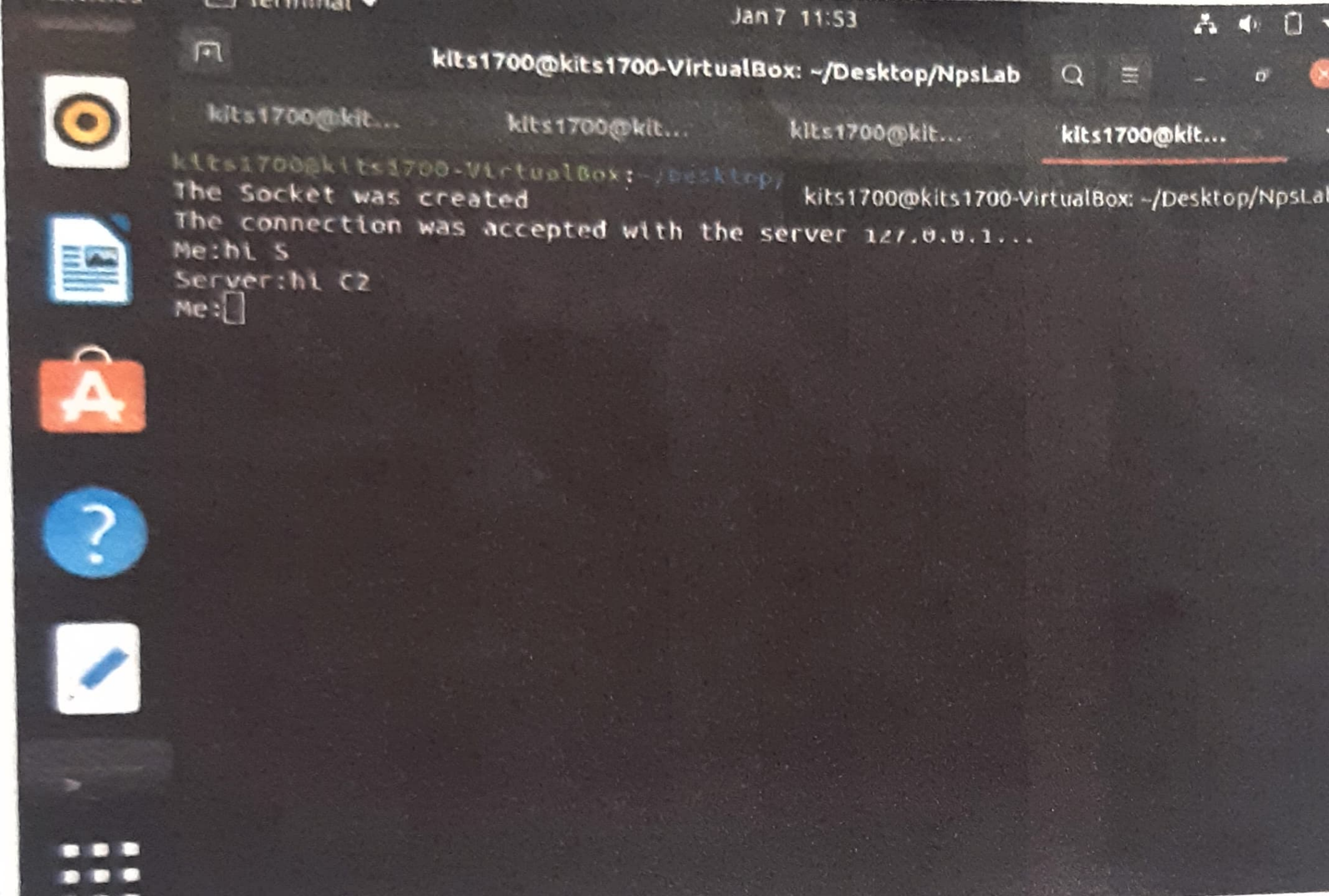
EXPT. NO. | NAME | Page No.: | Date: | YOUVA

M T W T F S S

5) Write a program to implement concurrent chat server that allows current logged in users to communicate one with other.

```
//Server
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <sys/stat.h>
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <fcntl.h>
#include <arpa/inet.h>
#include <string.h>
void str_echo(int connfd)
{ int n; int bufsize = 10240;
  char *buffer = malloc(bufsize);
  while((n=recv(connfd, buffer, bufsize,0))>0){
        fputs("client:", stdout);
        fputs(buffer, stdout);
        fputs("Me:", stdout);
  if(fgets(buffer, bufsize, stdin)!=NULL)
      { send(connfd, buffer, sizeof(buffer),0);}
    bzero(buffer, 10240);}}

int main()
{ int cont, listenfd, connfd, addrlen, addrlen2, fd, pid, addrlen3;
```

Teacher's Signature:


Terminal
Jan 7 11:53
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kits1700-VirtualBox:~/Desktop/
The Socket was created
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLa
The connection was accepted with the server 127.0.0.1...
Me:hi S
Server:hi C2
Me:

EXPT NO | NAME | Page No.: | Date: | YOUVA

```
struct sockaddr_in address, cli_address.
if( listenfd = socket (AF_INET, SOCK_STREAM, 0)) >0)
    printf("The socket was created.\n");
address.sin_family = AF_INET.
address.sin_addr.s_addr = INADDR_ANY;
address.sin_port= htons (16001);
printf(" The address before bind %s...\n", inet_ntoa(address.sin_addr));
if bind( listenfd ,(struct sockaddr*) &address, sizeof(address)) ==0)
  printf("Binding socket\n");
    printf(" The address after bind %s...\n", inet_ntoa(address.sin
                                                  _addr));
listen( listenfd,3);
printf("sever is listening\n");
getsockname(listenfd, (struct sockaddr*) & address, &addrlen);
printf(" The severs local address %s... and port %d\n", inet_ntoa
      (address.sin_addr), htons(address.sin_port));
for(;;) {
  addrlen = sizeof(struct sockaddr_in);
  connfd = accept (listenfd, (struct sockaddr*) &cli_address,
  addrlen2 = sizeof(struct sockaddr_in).          &addrlen);
  int ? = getpeername (connfd, (struct sockaddr*) &cli_address,
printf(" The client %s is connected... on port %d\n",      &addrlen);
    inet_ntoa(cli_address.sin_addr), htons(cli_address.sin_port));
if(pid=fork()) ==0)
{printf("inside child\n");
  close(listenfd);
  str_echo(connfd);
  exit(0); }
 close(connfd);
return 0; }
```

Teacher's Signature:

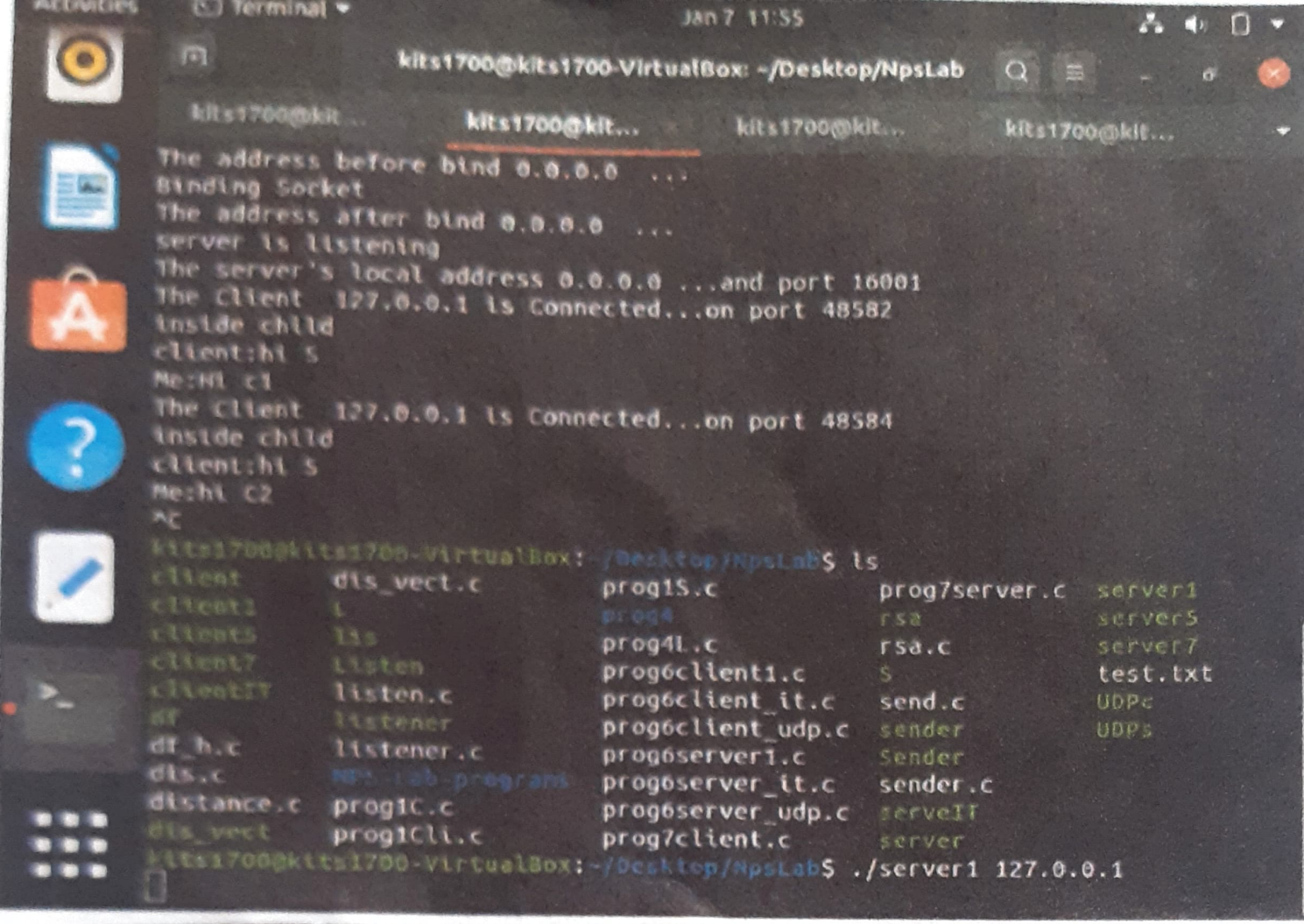
EXPT NO | NAME | Page No: | Date | YOUVA

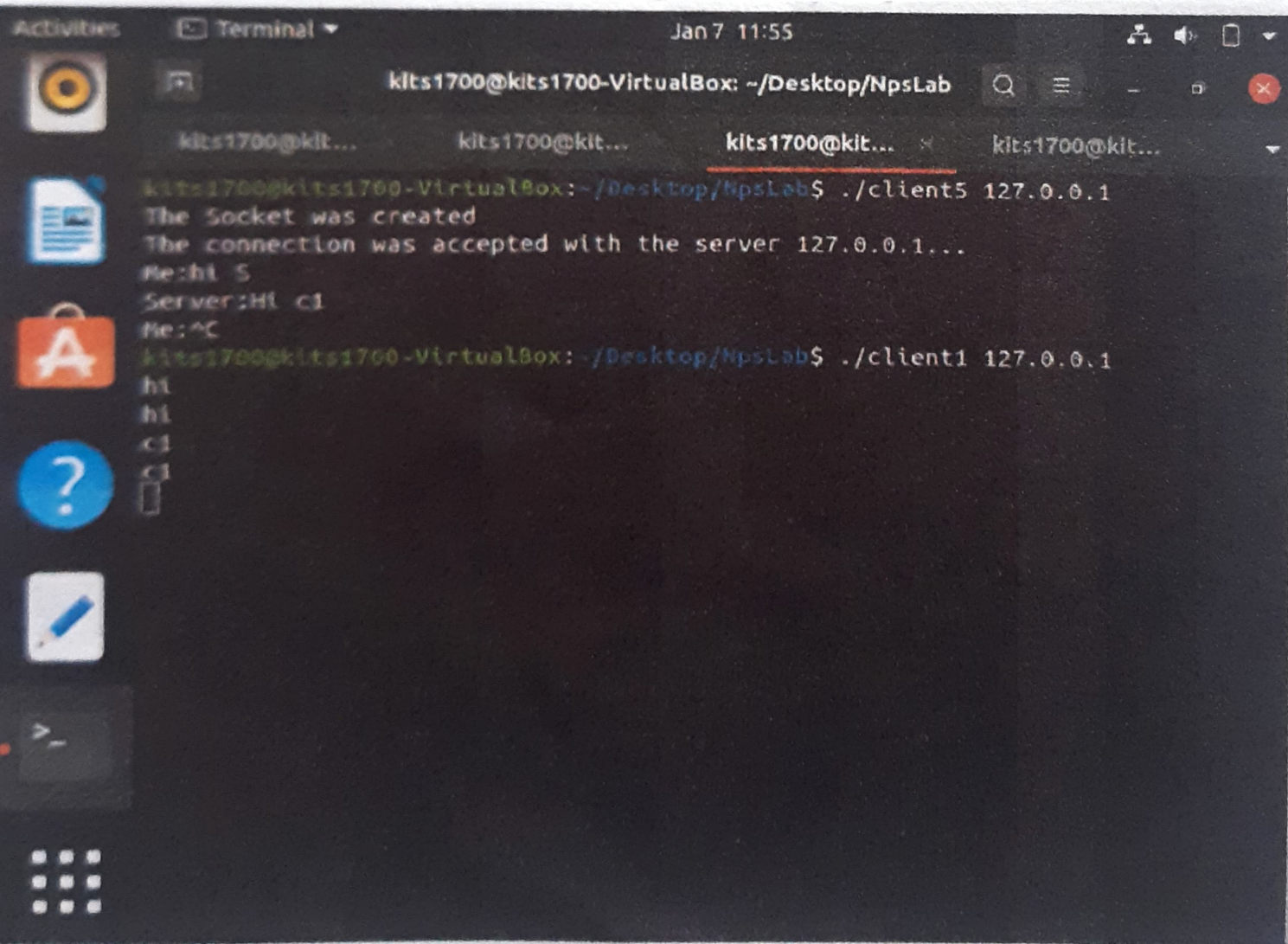
```
//client
#include<sys/socket.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<unistd.h>
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
#include<arpa/inet.h>
void sta_cli (FILE*fp, int sockfd)
{ int bufsize=1024, cont;
  char * buffer = malloc(bufsize);
  fputs("Me:", stdout);
  while(fgets(buffer, bufsize, fp)!=NULL)
{     send(sockfd, buffer, sizeof(buffer), 0);
  if((cont=recv(sockfd, buffer, bufsize, 0))>0)
    { fputs("Server:", stdout);
      fputs(buffer, stdout); }
        fputs("me:", stdout); }
printf("\nEOF\n");
        }
int main (int argc, char*argv[])
{  int create_socket;
   struct sockaddr_in address;
if((create_socket = socket(AF_INET, SOCK_STREAM, 0))>0)
   printf("The socket was created\n");
```

Teacher's Signature:

```
address.sin_family = AF_INET;
address.sin_port = htons(16001);
inet_pton(AF_INET, argv[1], &address.sin_addr);
if(connect(create_socket, (struct sockaddr*) &address, sizeof(address)) == 0)
        printf("The connection was accepted with the
                server %s ... \n", argv[1]);
else
    printf("error in connect \n");
shutdown(stdin, create_socket);
return close(create_socket);
}
```

Teacher's Signature: _____


Activities
Terminal
Jan 7 11:55
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
The address before bind 0.0.0.0 ...
Binding Socket
The address after bind 0.0.0.0 ...
server is listening
The server's local address 0.0.0.0 ...and port 16001
The Client 127.0.0.1 is Connected...on port 48582
inside child
client:hi S
Me:Hi c1
The Client 127.0.0.1 is Connected...on port 48584
inside child
client:hi S
Me:hi C2
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ls
client dis_vect.c prog1S.c prog7server.c server1
client1 i prog4 rsa server5
client5 lis prog4l.c rsa.c server7
client7 Listen prog6client1.c S test.txt
clientIT listen.c prog6client_it.c send.c UDPc
df listener prog6client_udp.c sender UDPs
df_h.c listener.c prog6server1.c Sender
dis.c NPS_Lab_programs prog6server_it.c sender.c
distance.c prog1C.c prog6server_udp.c serveIT
dis_vect prog1Cli.c prog7client.c server
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./server1 127.0.0.1


Activities
Terminal
Jan 7 11:55
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./client5 127.0.0.1
The Socket was created
The connection was accepted with the server 127.0.0.1...
Me:hi S
Server:Hi c1
Me:^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./client1 127.0.0.1
hi
hi
c1
c1

EXPT NO | NAME | Page No. | Date | YOUVA

M T W T F S S

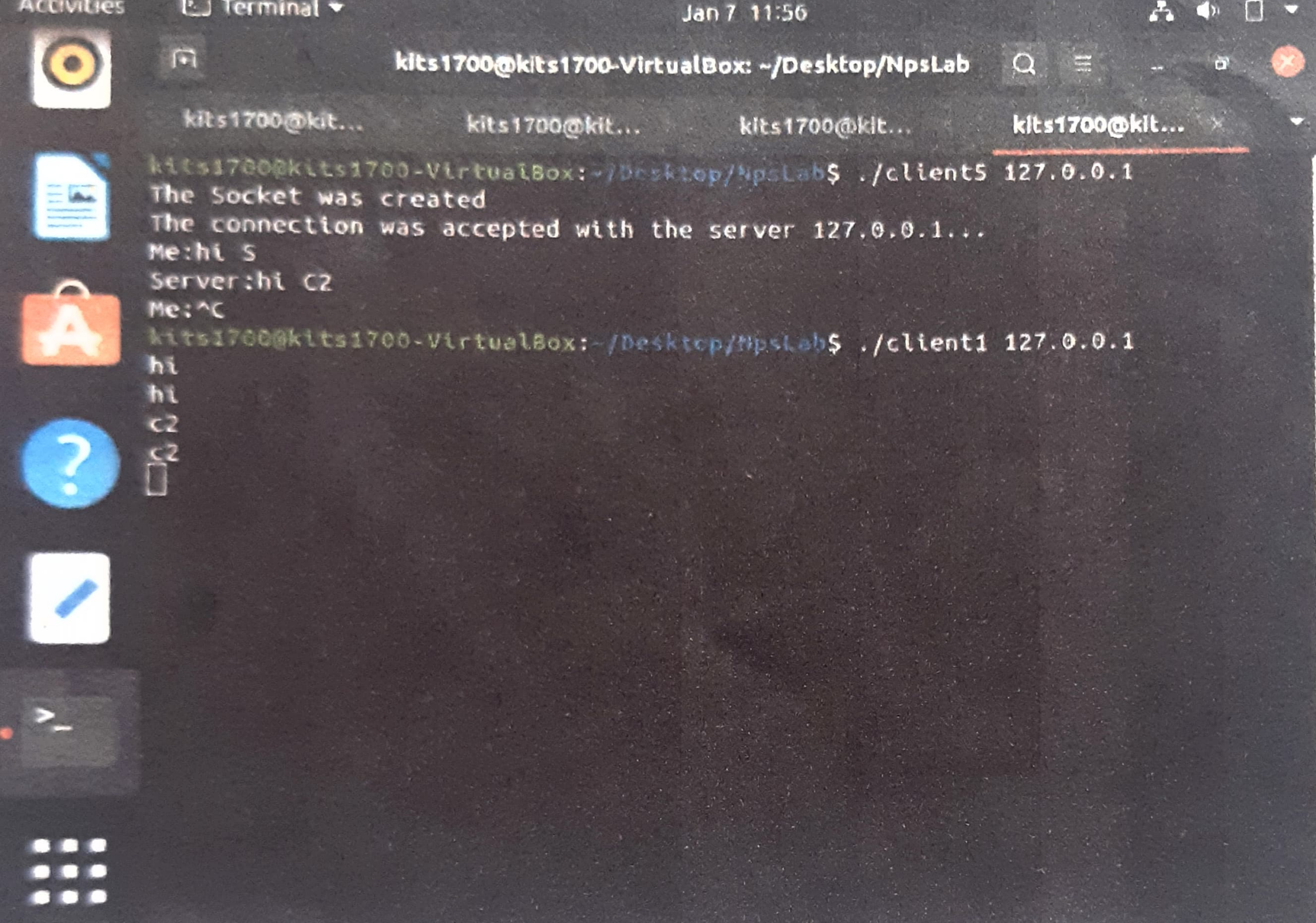
6) Implementation of concurrent and iterative echoserver using both connection & connectionless socket system calls.

a) Concurrent TCP

Server

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<string.h>
#include<arpa/inet.h>
#include<sys/socket.h>
#include<sys/types.h>
#include<netinet.h>
#include<errno.h>
#define SERV_PORT 9002
#define LISTENQ 1024
void str_echo(int sockfd)
{  ssize_t n;
   char buf[1000];
 again:
  while((n=read(sockfd,buf,1000))>0)
  { write(sockfd, buf, n);
    memset(buf, '0', strlen(buf));
    }
if (n<0 && errno == EINTR)
        goto again;
   else if (n<0)
     perror("str_echo: read error");
}
```

Teacher's Signature: _____


Activities
Terminal
Jan 7 11:56
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./client5 127.0.0.1
The Socket was created
The connection was accepted with the server 127.0.0.1...
Me:hi S
Server:hi C2
Me:^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./client1 127.0.0.1
hi
hi
c2
c2

EXPT. NO | NAME | Page No: | Date: | YOUVA

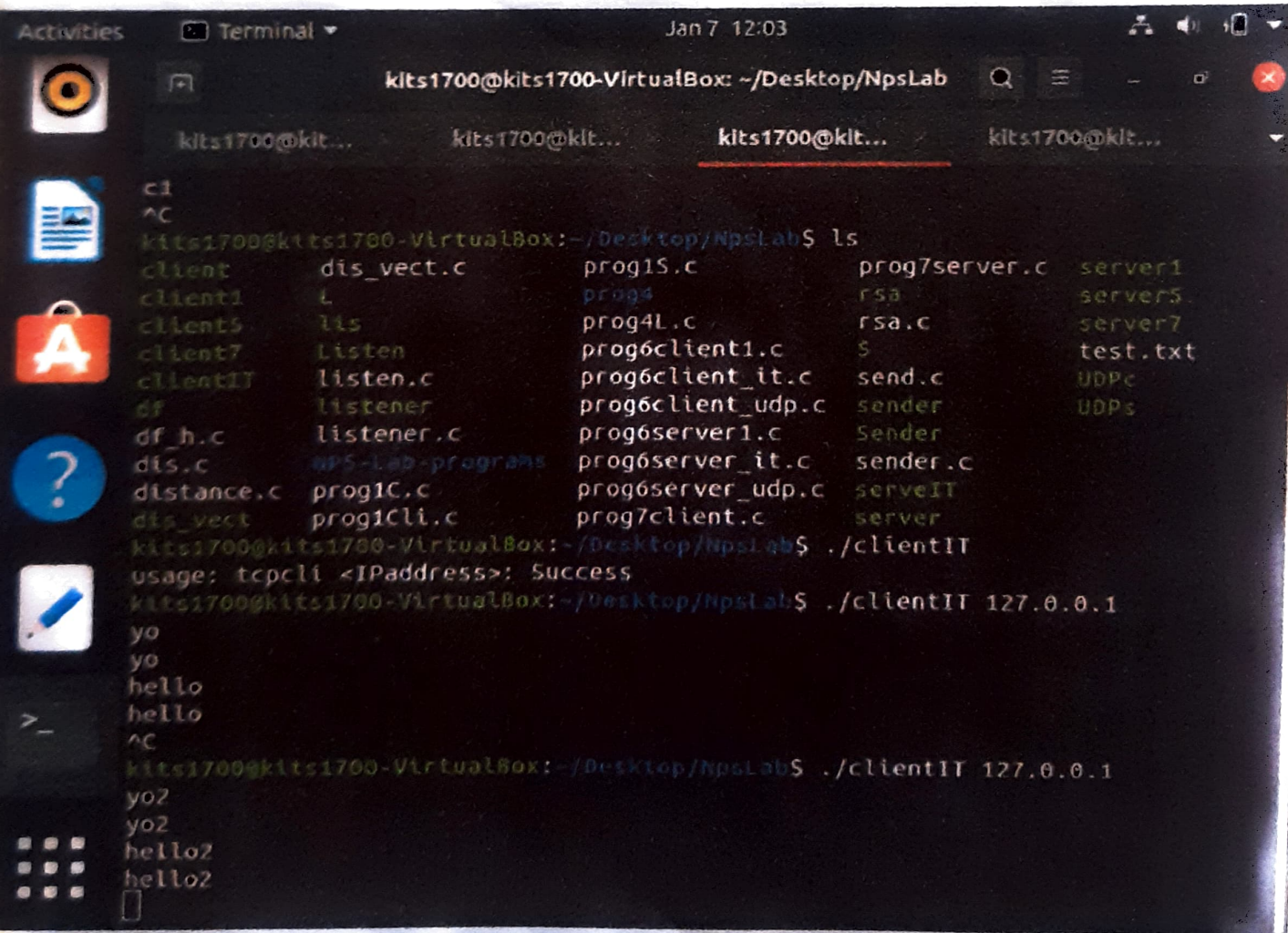
```
int main(int argc, char **argv)
{ int listenfd, connfd;
  pid_t childpid;
  socklen_t clilen;
  struct sockaddr_in cliaddr, servaddr;
listenfd = socket(AF_INET, SOCK_STREAM, 0);
bzero(&servaddr, sizeof(servaddr));
servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
servaddr.sin_port = htons(SERV_PORT);
bind(listenfd, (struct sockaddr*) &servaddr, sizeof(servaddr));
listen(listenfd, LISTENQ);
for(;;){
      clilen = sizeof(cliaddr);
  connfd = accept(listenfd, (struct sockaddr*) &cliaddr, &clilen);
 if((childpid = fork()) == 0)
 { close(listenfd);
   str_echo(connfd);
   exit(0); }

  close(connfd); } }
```

Teacher's Signature:

```
//client
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<string.h>
#include<arpa/inet.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<errno.h>
#define MAXLINE 1000
#define SERV_PORT 9002
void str_cli(FILE *fp, int sockfd)
{ char sendline[MAXLINE], recvline[MAXLINE];
  while(fgets(sendline, MAXLINE, fp))
  { write(sockfd, sendline, strlen(sendline));
    if(read(sockfd, recvline, MAXLINE)==0)
    { perror("str_cli: server terminated prematurely");
      exit(EXIT_FAILURE);
    }
    fputs(recvline, stdout);
  }
}
int main(int argc, char **argv)
{ int sockfd; struct sockaddr_in servaddr;
  if(argc!=2)
  { perror("usage: tcpcli<IPaddress>");
    exit(EXIT_FAILURE);
  }
```

Teacher's Signature:


Activities
Terminal
Jan 7 12:02
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
The address after bind 0.0.0.0 ...
server is listening
The server's local address 0.0.0.0 ...and port 16001
The Client 127.0.0.1 is Connected...on port 48582
inside child
client:hi S
Me:Hi c1
The Client 127.0.0.1 is Connected...on port 48584
inside child
client:hi S
Me:hi C2
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ls
client dis_vect.c prog1S.c prog7server.c server1
client1 L prog4 rsa server5
client5 lis prog4L.c rsa.c server7
client7 Listen prog6client1.c S test.txt
clientIT listen.c prog6client_it.c send.c UDPc
df listener prog6client_udp.c sender UDPs
df_h.c listener.c prog6server1.c Sender
dis.c NPS-Lab-programs prog6server_it.c sender.c
distance.c prog1C.c prog6server_udp.c serveIT
dis_vect prog1Cli.c prog7client.c server
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./server1 127.0.0.1
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./serveIT


Activities
Terminal
Jan 7 12:03
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
c1
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ls
client dis_vect.c prog1S.c prog7server.c server1
client1 L prog4 rsa server5
client5 lis prog4L.c rsa.c server7
client7 Listen prog6client1.c S test.txt
clientIT listen.c prog6client_it.c send.c UDPc
df listener prog6client_udp.c sender UDPs
df_h.c listener.c prog6server1.c Sender
dis.c NPS-Lab-programs prog6server_it.c sender.c
distance.c prog1C.c prog6server_udp.c serveIT
dis_vect prog1Cli.c prog7client.c server
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./clientIT
usage: tcpcli <IPaddress>: Success
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./clientIT 127.0.0.1
yo
yo
hello
hello
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./clientIT 127.0.0.1
yo2
yo2
hello2
hello2

```
sockfd = socket (AF_INET, SOCK_STREAM, 0);
bzero (&servaddr, sizeof (servaddr));
servaddr.sin_family = AF_INET;
servaddr.sin_port = htons (SERV_PORT);
inet_pton (AF_INET, argv[1], &servaddr.sin_addr);
connect (sockfd, (struct sockaddr *) &servaddr, sizeof (servaddr));
str_cli (stdin, sockfd);
    exit(0);
}
```

A) // Iterative TCP

// server

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <arpa/inet.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <errno.h>
#define SERV_PORT 9002
#define LISTENQ 1024
void str_echo (int sockfd)
{ ssize_t n; char buf[1000];
    again:
        while ( (n = read (sockfd, buf, 1000)) > 0)
    { write (sockfd, buf, n); memset (buf, '\0', strlen (buf)); }
```

Teacher's Signature:

EXPT NO | NAME | Page No. | Date | YOUVA

```
int main(int argc, char * * argv)
{  int listenfd, connfd;
   socklen_t clilen;
   struct sockaddr_in cliaddr, servaddr;
   listenfd = socket(AF_INET, SOCK_STREAM, 0);
   bzero(&servaddr, sizeof(servaddr));
   servaddr.sin_family = AF_INET;
   servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
   servaddr.sin_port = htons(SERV_PORT);
   bind(listenfd, (struct sockaddr*) &servaddr, sizeof(servaddr));
   listen(listenfd, LISTENQ);
   for(;;)
   { clilen = sizeof(cliaddr);
     connfd = accept(listenfd, (struct sockaddr*) &cliaddr, &clilen);
      str_echo(connfd);
     close(connfd);
   }
}
```

//client

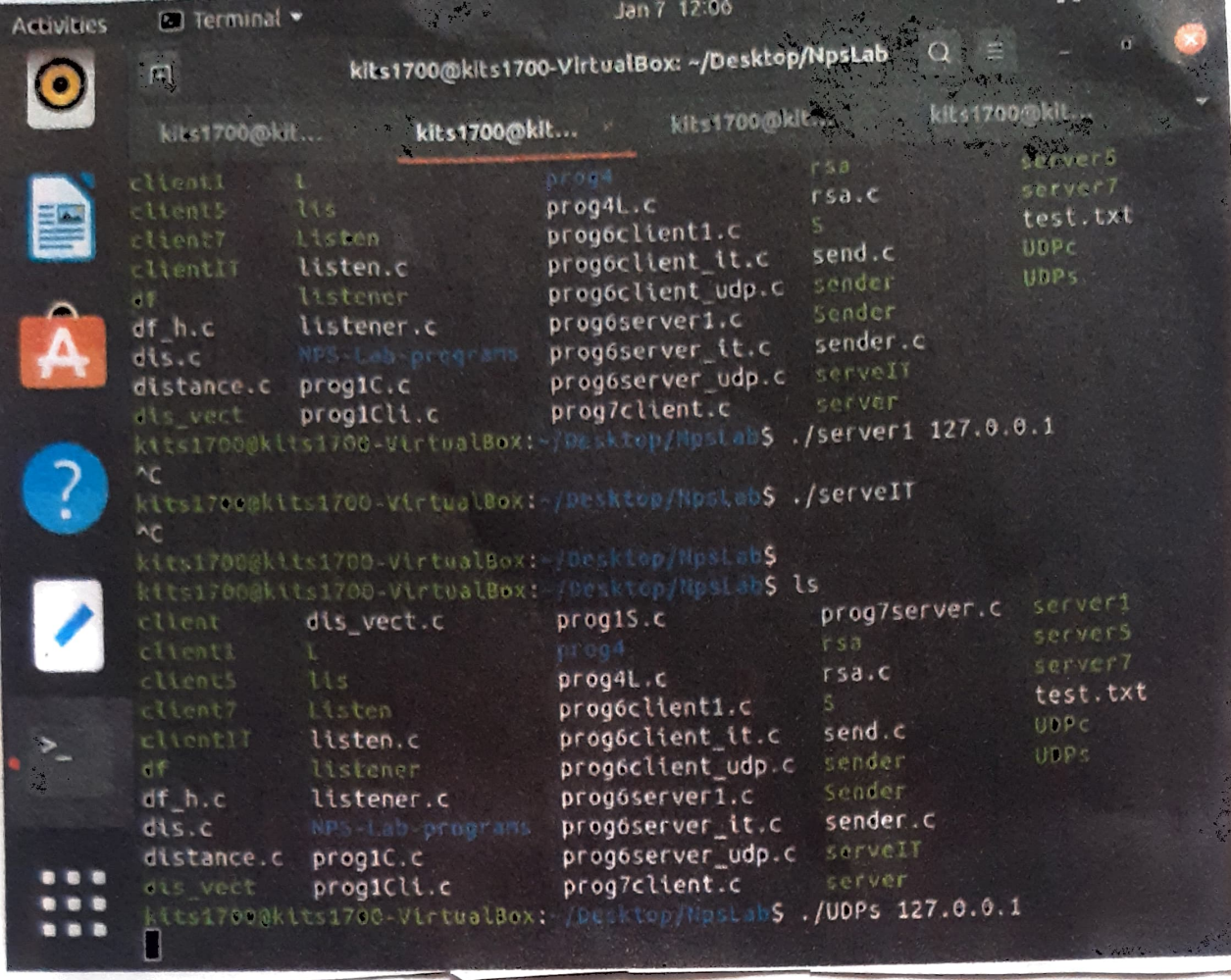
```
#include<stdio.h>
#include<stdlib.h>
#include<sys/socket.h>
#include<sys/types.h>
#include<arpa/inet.h>
#include<netinet/in.h>
#include<string.h>
#include<unistd.h>
#include<errno.h>
```

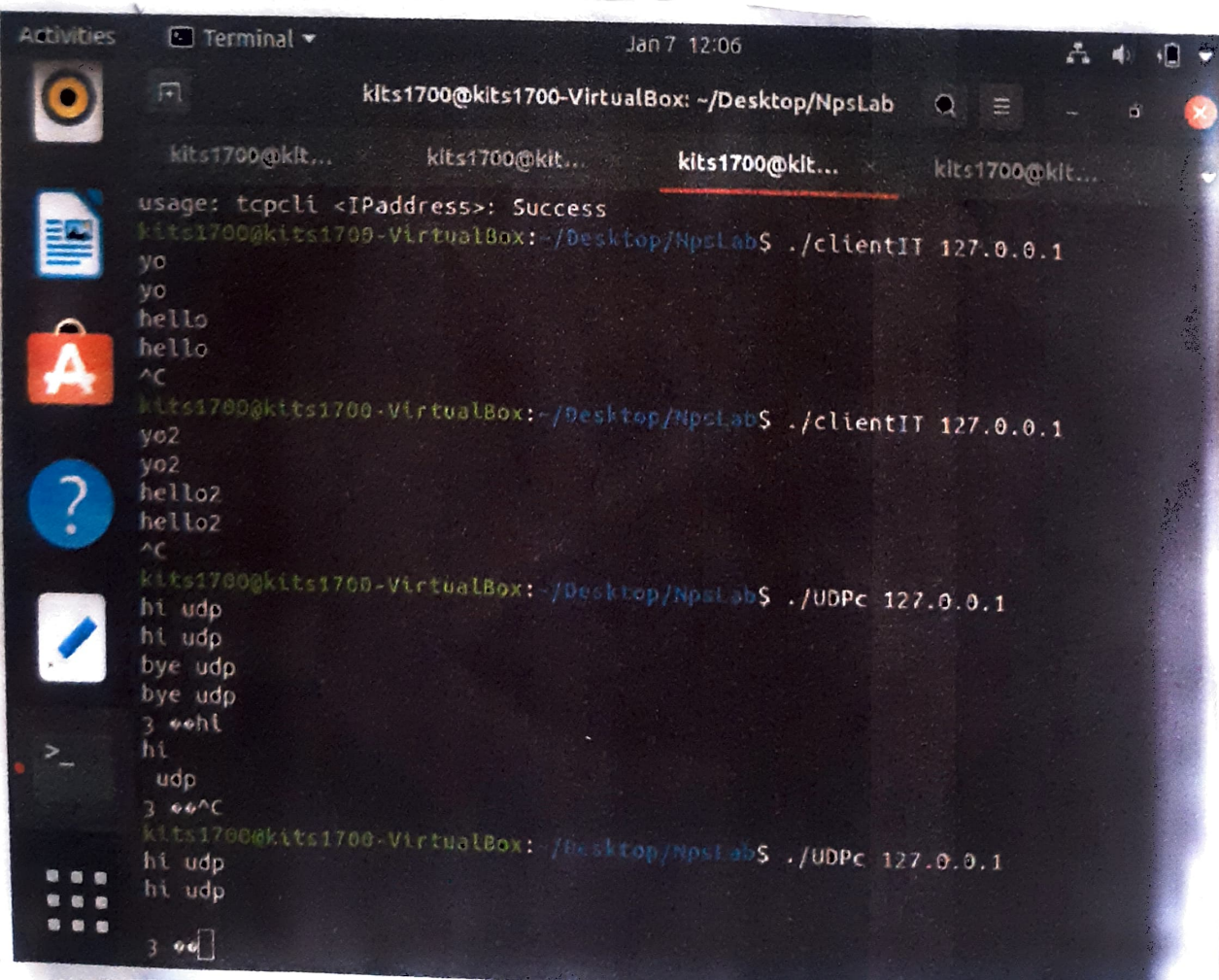
Teacher's Signature

EXPT. NO | NAME | Page No. | Date | YOUVA

```
#define MAXLINE 1000
#define SERV_PORT 9002
void sta_cli(FILE *fp, int sockfd)
{ char sendline[MAXLINE], recvline[MAXLINE];
  while(fgets(sendline, MAXLINE, fp))
  { write(sockfd, sendline, strlen(sendline));
    if(read(sockfd, recvline, MAXLINE)==0)
    { perror("sta_cli: server terminated prematurely");
      exit(EXIT_FAILURE);
    }
    fputs(recvline, stdout);
  }
}
int main(int argc, char **argv)
{ int sockfd;
  struct sockaddr_in servaddr;
  if(argc!=2)
  { perror("usage: tcpcli<IPaddress>");
    exit(EXIT_FAILURE);
  }
sockfd = socket(AF_INET, SOCK_STREAM, 0);
bzero(&servaddr, sizeof(servaddr));
servaddr.sin_family= AF_INET;
servaddr.sin_port = htons(SERV_PORT);
inet_pton(AF_INET, argv[1], &servaddr.sin_addr);
connect(sockfd, (struct sockaddr*)&servaddr, sizeof(servaddr));
sta_cli(stdin, sockfd);
exit(0);
}
```

Teacher's Signature:


Activities
Terminal
Jan 7 12:06
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
client1 1 prog4 rsa server5
client5 lis prog4L.c rsa.c server7
client7 listen prog6client1.c s test.txt
clientIT listen.c prog6client_it.c send.c UDPc
df listener prog6client_udp.c sender UDPs
df_h.c listener.c prog6server1.c Sender
dis.c NPS-Lab-programs prog6server_it.c sender.c
distance.c prog1C.c prog6server_udp.c serveIT
dis_vect prog1Cli.c prog7client.c server
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./server1 127.0.0.1
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./serveIT
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ls
client dis_vect.c prog1S.c prog7server.c server1
client1 1 prog4 rsa server5
client5 lis prog4L.c rsa.c server7
client7 listen prog6client1.c s test.txt
clientIT listen.c prog6client_it.c send.c UDPc
df listener prog6client_udp.c sender UDPs
df_h.c listener.c prog6server1.c Sender
dis.c NPS-Lab-programs prog6server_it.c sender.c
distance.c prog1C.c prog6server_udp.c serveIT
dis_vect prog1Cli.c prog7client.c server
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./UDPs 127.0.0.1


Activities
Terminal
Jan 7 12:06
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
usage: tcpcli <IPaddress>: Success
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./clientIT 127.0.0.1
yo
yo
hello
hello
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./clientIT 127.0.0.1
yo2
yo2
hello2
hello2
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./UDPc 127.0.0.1
hi udp
hi udp
bye udp
bye udp
3 ^^hi
hi
udp
3 ^^^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./UDPc 127.0.0.1
hi udp
hi udp
3 ^^

EXPT NO | NAME | Page No. | Date | YOUVA

M T W T F S S

ii) UDP Echo Server

```
//Server
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<string.h>
#include<arpa/inet.h>
#include<sys/socket.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<errno.h>
#define SERV_PORT 9002
#define MAXLINE 1024
void dg_echo(int sockfd, struct sockaddr* pcliaddr, socklen_t clilen)
{ int n;
  socklen_t len;
  char mesg[MAXLINE];
  for(;;) { len = clilen;
            n=recvfrom(sockfd, mesg, MAXLINE, 0, pcliaddr, &len);
            sendto(sockfd, mesg, strlen(mesg), 0, pcliaddr, len);

          }
}
int main(int argc, char **argv)
{ int sockfd; struct sockaddr_in servaddr, cliaddr;
  sockfd = socket(AF_INET, SOCK_DGRAM, 0);
  bzero(&servaddr, sizeof(servaddr));
  servaddr.sin_family = AF_INET;
  servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
```

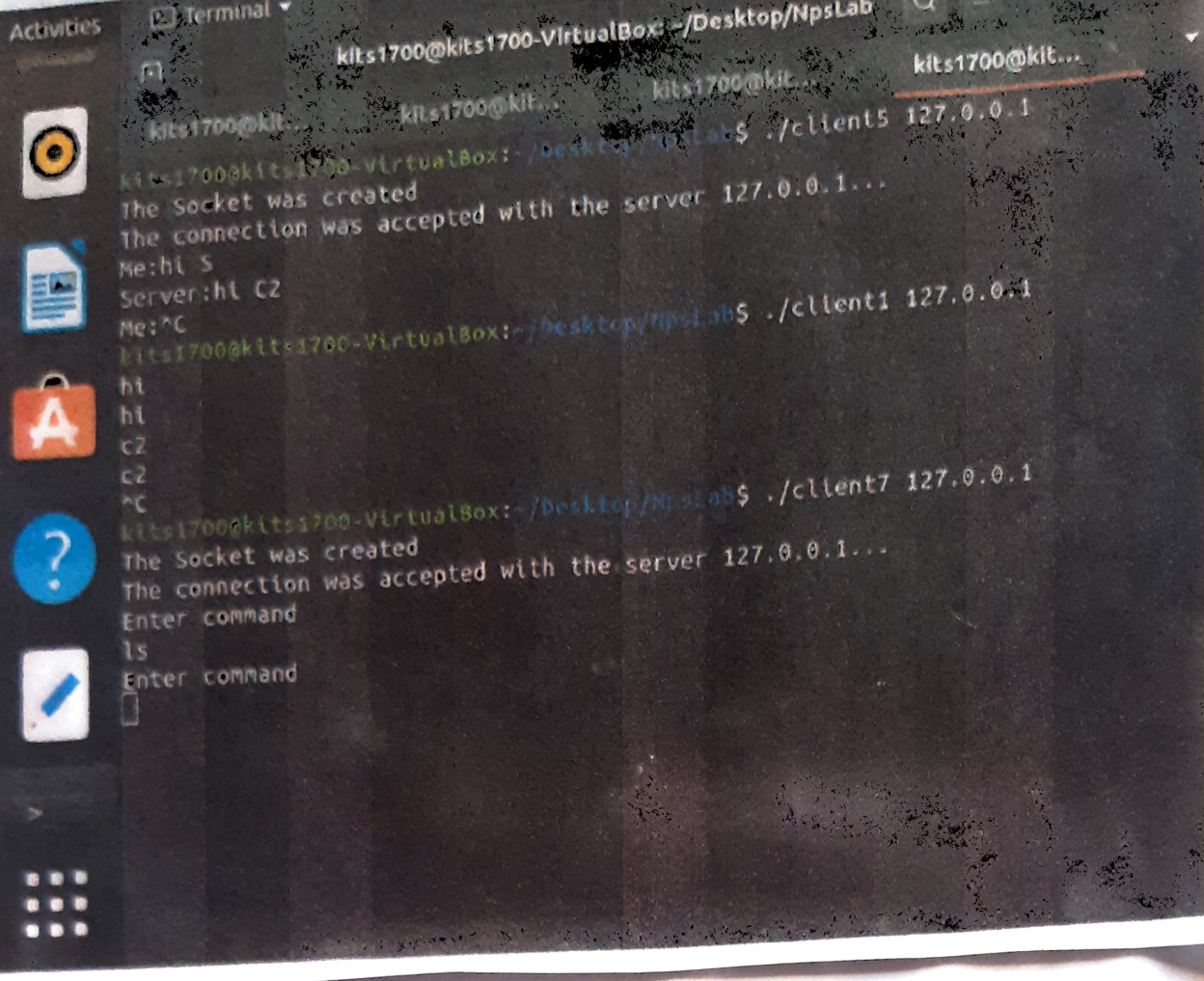
Teacher's Signature:

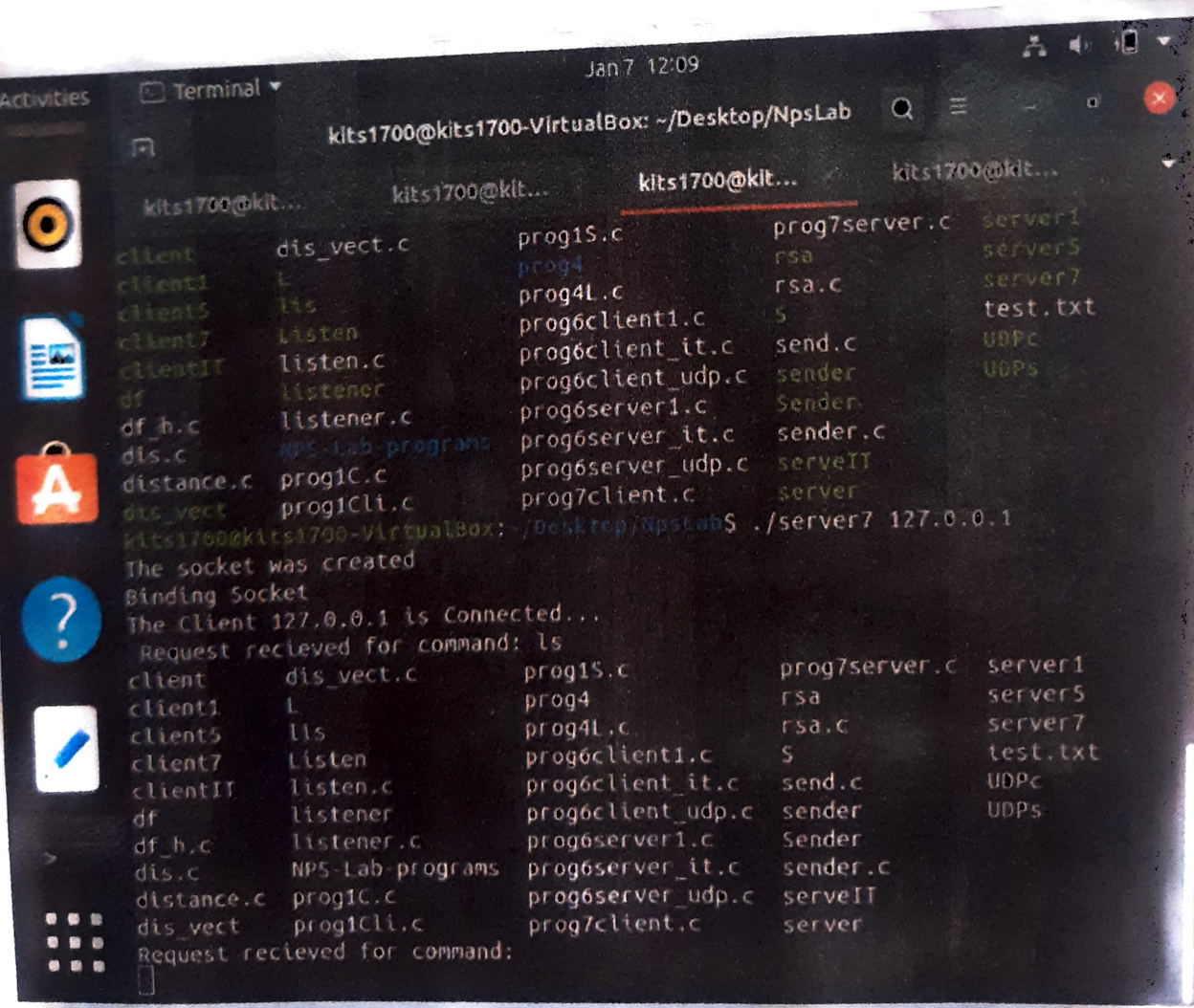
servaddr.sin_port = htons(SERV_PORT);
bind(sockfd, (struct sockaddr *)&servaddr, sizeof(servaddr));
dg_echo(sockfd, (struct sockaddr*)&cliaddr, sizeof(cliaddr));
}

// client
void dg_cli(FILE *fp, int sockfd, const struct sockaddr* pservaddr, socklen_t servlen)
{ int n; char sendline[MAXLINE], recvline[MAXLINE+1];
while(fgets(sendline, MAXLINE, fp) != NULL) {
sendto(sockfd, sendline, strlen(sendline), 0, pservaddr, servlen);
n = recvfrom(sockfd, recvline, MAXLINE, 0, NULL, NULL);
recvline[n] = 0;
fputs(recvline, stdout);
}
}

int main(int argc, char** argv)
{ int sockfd; struct sockaddr_in servaddr;
if(argc != 2)
{ perror("Usage: udpcli IPaddress");
exit(-1); }
bzero(&servaddr, sizeof(servaddr));
servaddr.sin_family = AF_INET;
servaddr.sin_port = htons(SERV_PORT);
inet_pton(AF_INET, argv[1], &servaddr.sin_addr);
sockfd = socket(AF_INET, SOCK_DGRAM, 0);
dg_cli(stdin, sockfd, (struct sockaddr*)&servaddr, sizeof(servaddr));
exit(0);
}

Teacher's Signature: _____


Activities
Terminal
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./client5 127.0.0.1
The Socket was created
The connection was accepted with the server 127.0.0.1...
Me:hi S
Server:hi C2
Me:^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./client1 127.0.0.1
hi
hi
c2
c2
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./client7 127.0.0.1
The Socket was created
The connection was accepted with the server 127.0.0.1...
Enter command
ls
Enter command


Activities
Terminal
Jan 7 12:09
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
client dis_vect.c prog1S.c prog7server.c server1
client1 L prog4 rsa server5
client5 lis prog4L.c rsa.c server7
client7 Listen prog6client1.c S test.txt
clientII listen.c prog6client_it.c send.c UDPc
df listener prog6client_udp.c sender UDPs
df_h.c listener.c prog6server1.c Sender
dis.c NP5-Lab-programs prog6server_it.c sender.c
distance.c prog1C.c prog6server_udp.c serveIT
dis_vect prog1Cli.c prog7client.c server
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./server7 127.0.0.1
The socket was created
Binding Socket
The Client 127.0.0.1 is Connected...
Request recieved for command: ls
client dis_vect.c prog1S.c prog7server.c server1
client1 L prog4 rsa server5
client5 lis prog4L.c rsa.c server7
client7 Listen prog6client1.c S test.txt
clientII listen.c prog6client_it.c send.c UDPc
df listener prog6client_udp.c sender UDPs
df_h.c listener.c prog6server1.c Sender
dis.c NP5-Lab-programs prog6server_it.c sender.c
distance.c prog1C.c prog6server_udp.c serveIT
dis_vect prog1Cli.c prog7client.c server
Request recieved for command:

EXPT NO | NAME | Page No: | Date | YOUVA

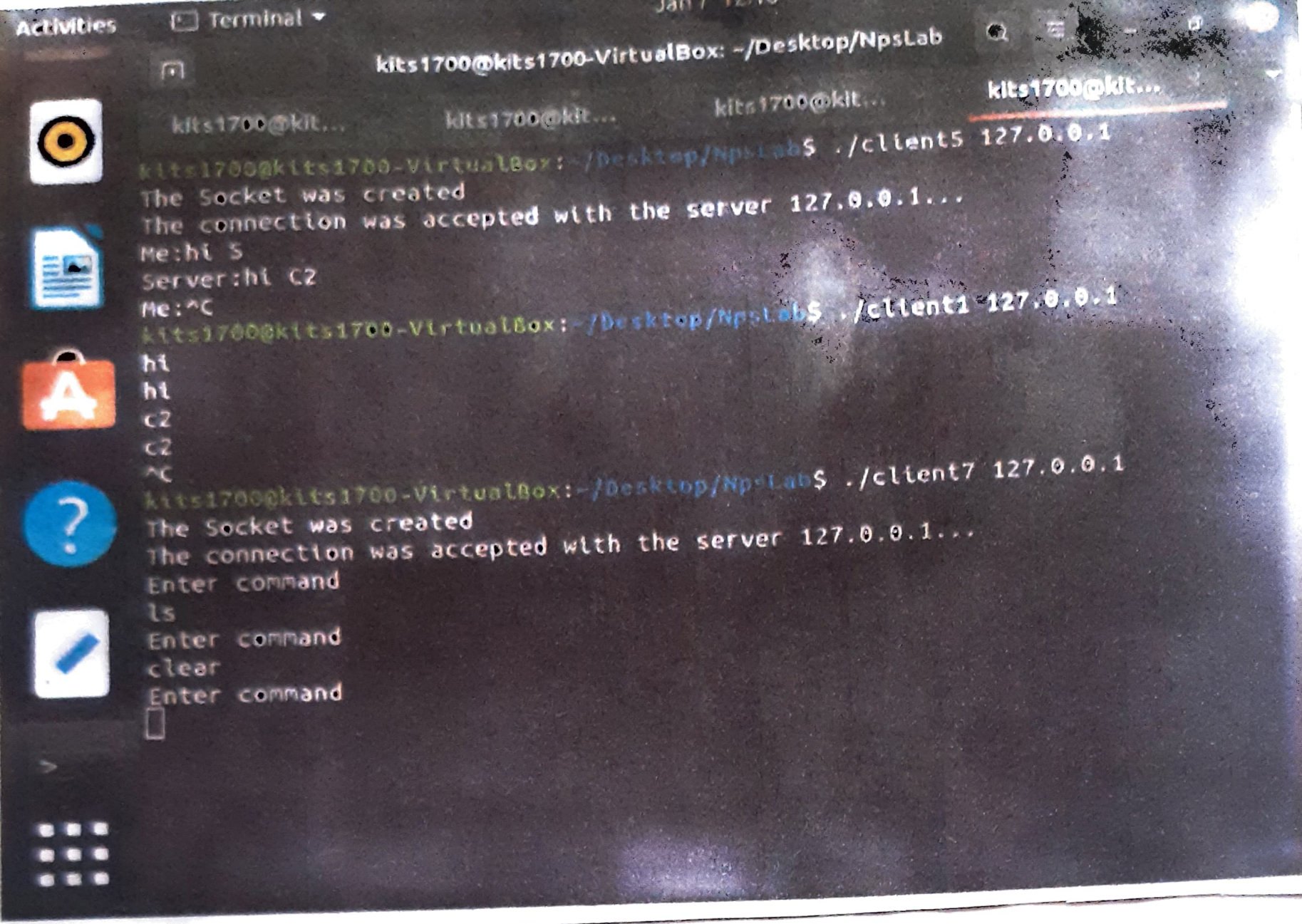
M T W T F S S

7) Implementation of remote command execution using socket system calls.

//Server

```
#include<sys/types.h>
#include<sys/socket.h>
#include<unistd.h>
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<sys/stat.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<fcntl.h>
int main()
{ int port, create_socket, new_socket, addrlen, fd;
  int bufsize = 1024; char *buffer = malloc(bufsize);
  char recv[256]; struct sockaddr_in address;
if ((create_socket = socket (AF_INET, SOCK_STREAM, 0))>0)
        printf("The socket was created\n");
  address.sin_family = AF_INET;
  address.sin_addr.s_addr = INADDR_ANY;
  address.sin_port = htons(15000);
if (bind (create_socket, (struct sockaddr *) &address, sizeof (address))
        printf("Binding Socket\n");                                    ==0)
 listen(create_socket, 3);
 addrlen = sizeof (struct sockaddr_in);
new_socket = accept ( create_socket, (struct sockaddr *) &address,
                                                    &addrlen);
```

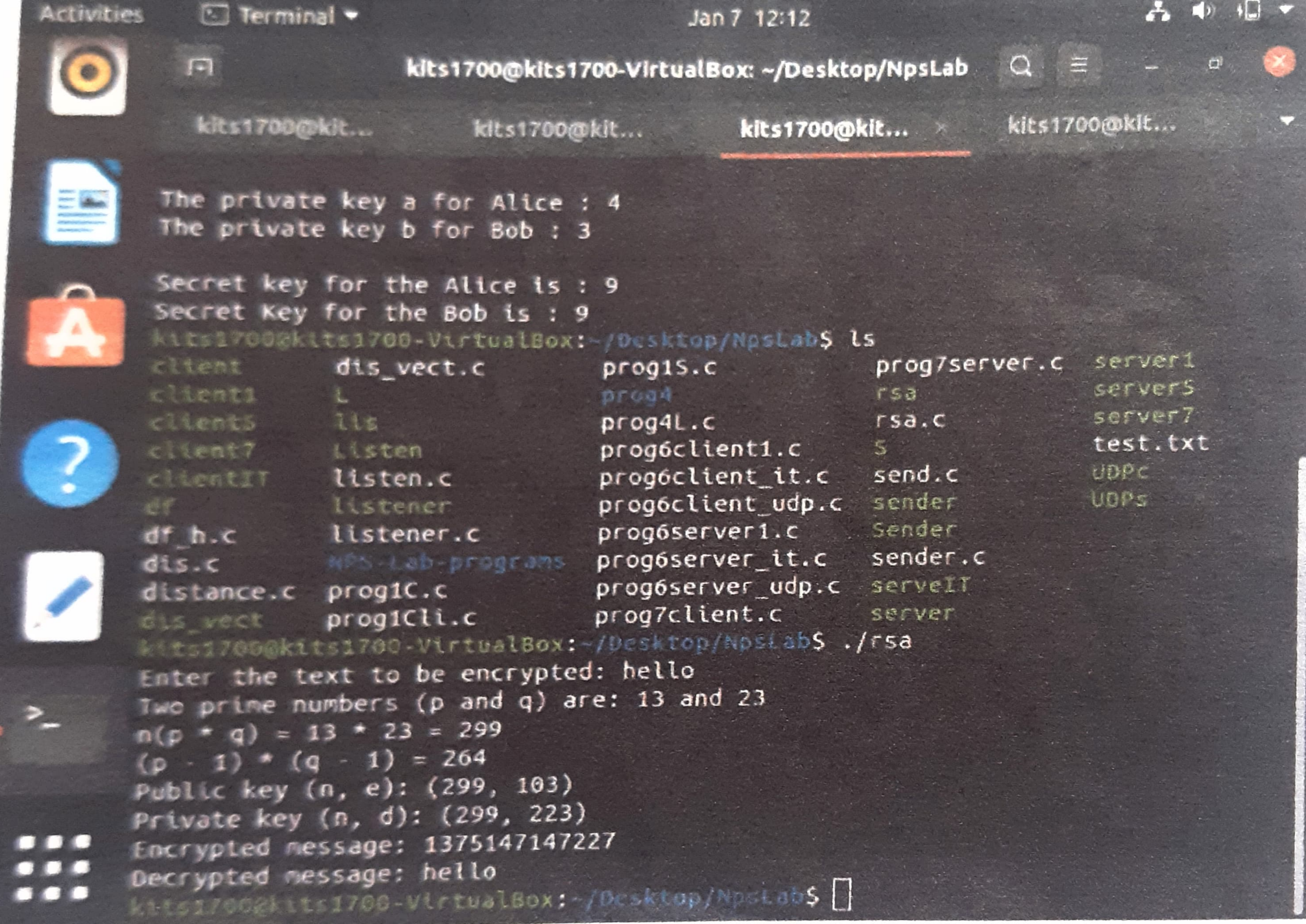
Teacher's Signature: _____


Activities
Terminal
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./client5 127.0.0.1
The Socket was created
The connection was accepted with the server 127.0.0.1...
Me:hi S
Server:hi C2
Me:^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./client1 127.0.0.1
hi
hi
c2
c2
^C
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./client7 127.0.0.1
The Socket was created
The connection was accepted with the server 127.0.0.1...
Enter command
ls
Enter command
clear
Enter command

```
if (new_socket>0)
  printf("The Client %s is Connected ...\n", inet_ntoa(address.sin_addr));

  int bytes;
while((bytes = recv(new_socket, receiv, 255,0))>0)
{ receiv[bytes] = '\0';
   if(strcmp(receiv,"end")==0)
   { close(new_socket);
      close(create_socket);
      exit(0); }
else
{ printf("Request received for cmd: %s\n", receiv);
      system(receiv);
                    }
}
Close(new_socket);
return close(create_socket);
}

// Client
int main(int argc, char **argv)
{ int create_socket; int bufsize=1024;
  char * buffer = malloc(bufsize);
  char com[256];
  struct sockaddr_in address;
  if((create_socket = socket(AF_INET, SOCK_STREAM,0))>0)
        printf("The socket was created\n");
address.sin_family = AF_INET;
address.sin_port = htons(15000);
```

Teacher's Signature: _____

```
inet_pton(AF_INET, argv[1], &address.sin_addr);
if(connect(create_socket, (struct sockaddr *)&address, sizeof(address)) == 0)
    printf("The connection was accepted with server %s\n", argv[1]);

while(1)
{
    printf("Enter cmd\n");
    scanf("%s", com);
    send(create_socket, com, strlen(com), 0);
}
printf("\n EOF \n");
return close(create_socket);
}
```


Activities
Terminal
Jan 7 12:12
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
The private key a for Alice : 4
The private key b for Bob : 3
Secret key for the Alice is : 9
Secret Key for the Bob is : 9
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ls
client
client1
client5
client7
clientIT
df
df_h.c
dis.c
distance.c
dis_vect
dis_vect.c
L
lis
Listen
listen.c
listener
listener.c
NPS-Lab-programs
prog1C.c
prog1Cli.c
prog1S.c
prog4
prog4L.c
prog6client1.c
prog6client_it.c
prog6client_udp.c
prog6server1.c
prog6server_it.c
prog6server_udp.c
prog7client.c
prog7server.c
rsa
rsa.c
S
send.c
sender
Sender
sender.c
serveIT
server
server1
server5
server7
test.txt
UDPc
UDPs
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./rsa
Enter the text to be encrypted: hello
Two prime numbers (p and q) are: 13 and 23
n(p * q) = 13 * 23 = 299
(p - 1) * (q - 1) = 264
Public key (n, e): (299, 103)
Private key (n, d): (299, 223)
Encrypted message: 1375147147227
Decrypted message: hello
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$

EXPT NO | NAME | Page No. | Date | YOUVA

8) Write a program to encrypt & decrypt data using RSA and exchange the key securely using Diffie-Hellman Key exchange protocol

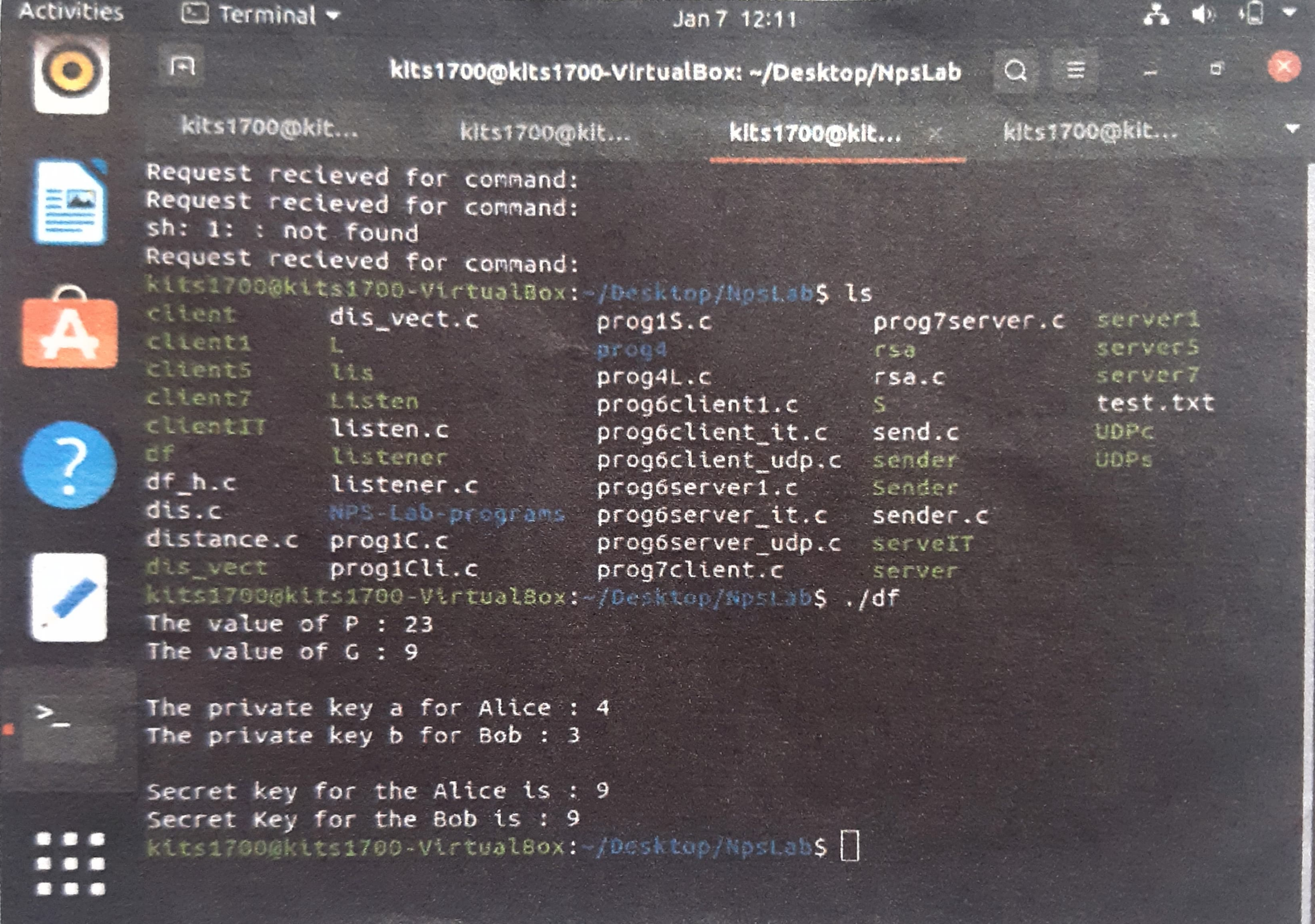
```
//RSA
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
#include<string.h>
long int gcd(long int a, long int b)
{ if(a==0)
    return b;
  if(b==0)
    return a;
  return gcd(b, a%b);
}
long int isprime(long int a)
{ int i;
  for(int i=2; i<a; i++)
  { if(a%i==0)
      return 0;
  }
  return 1;
}
long int encrypt(long int ch, long int n, long int e)
{ long int i, temp=ch;
  for(i=1; i<e; i++)
    temp = (temp * ch)%n;
  return temp;
}
```

Teacher's Signature: _____

EXPT. NO | NAME | Page No: | Date | YOUVA

```
long int decrypt(long int ch, long int n, long int d)
{
    long int i, temp = ch;
    for (i=1; i<d; i++)
        temp = (temp*ch) % n;
    return temp;
}

int main()
{
long int p; int len; long int p, q, n, phi, e, d, cipher[50];
 char text[50];
printf("Enter text to be encrypted: \n");
 scanf("%s", text);
len = strlen(text);
do {
    p = rand()%30;
} while(!isprime(p));
n=p*q;
do
{
    q = rand%30; } while(!isprime(q));
n=p*q; phi=(p-1)*(q-1);
do
{ e = rand%phi; } while(gcd(phi,e)!=1);

do { d = rand()%phi; } while((d*e%phi)!=1);
printf("2 prime no.s are: %ld and %ld \n", P, q);
printf(p, q, p*q);
printf(phi);
printf("PU = (n,e) : (%ld, %ld) \n", n, e);
printf("PR = (n,d) : (%ld, %ld) \n", n, d);
```

Teacher's Signature: _____


Activities
Terminal
Jan 7 12:11
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kit...
kits1700@kit...
kits1700@kit...
kits1700@kit...
Request recieved for command:
Request recieved for command:
sh: 1: : not found
Request recieved for command:
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ls
client
client1
client5
client7
clientIT
df
df_h.c
dis.c
distance.c
dis_vect
dis_vect.c
L
lis
Listen
listen.c
listener
listener.c
NPS-Lab-programs
prog1C.c
prog1Cli.c
prog1S.c
prog4
prog4L.c
prog6client1.c
prog6client_it.c
prog6client_udp.c
prog6server1.c
prog6server_it.c
prog6server_udp.c
prog7client.c
prog7server.c
rsa
rsa.c
S
send.c
sender
Sender
sender.c
serveIT
server
server1
server5
server7
test.txt
UDPc
UDPs
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./df
The value of P : 23
The value of G : 9
The private key a for Alice : 4
The private key b for Bob : 3
Secret key for the Alice is : 9
Secret Key for the Bob is : 9
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$

```
for(i=0; i<len; i++)
cipher[i] = encypt(text[i], n, e);
printf("Encrypted msg: \n");
for(i=0; i<len; i++)
printf("%ld", cipher[i]);
for(i=0; i<len; i++)
text[i] = decypt(cipher[i], n, d);
printf("\n");
printf("Decypted msg \n");
for(i=0; i<len; i++)
printf("%c", text[i]);
printf("\n");
return 0; }
```

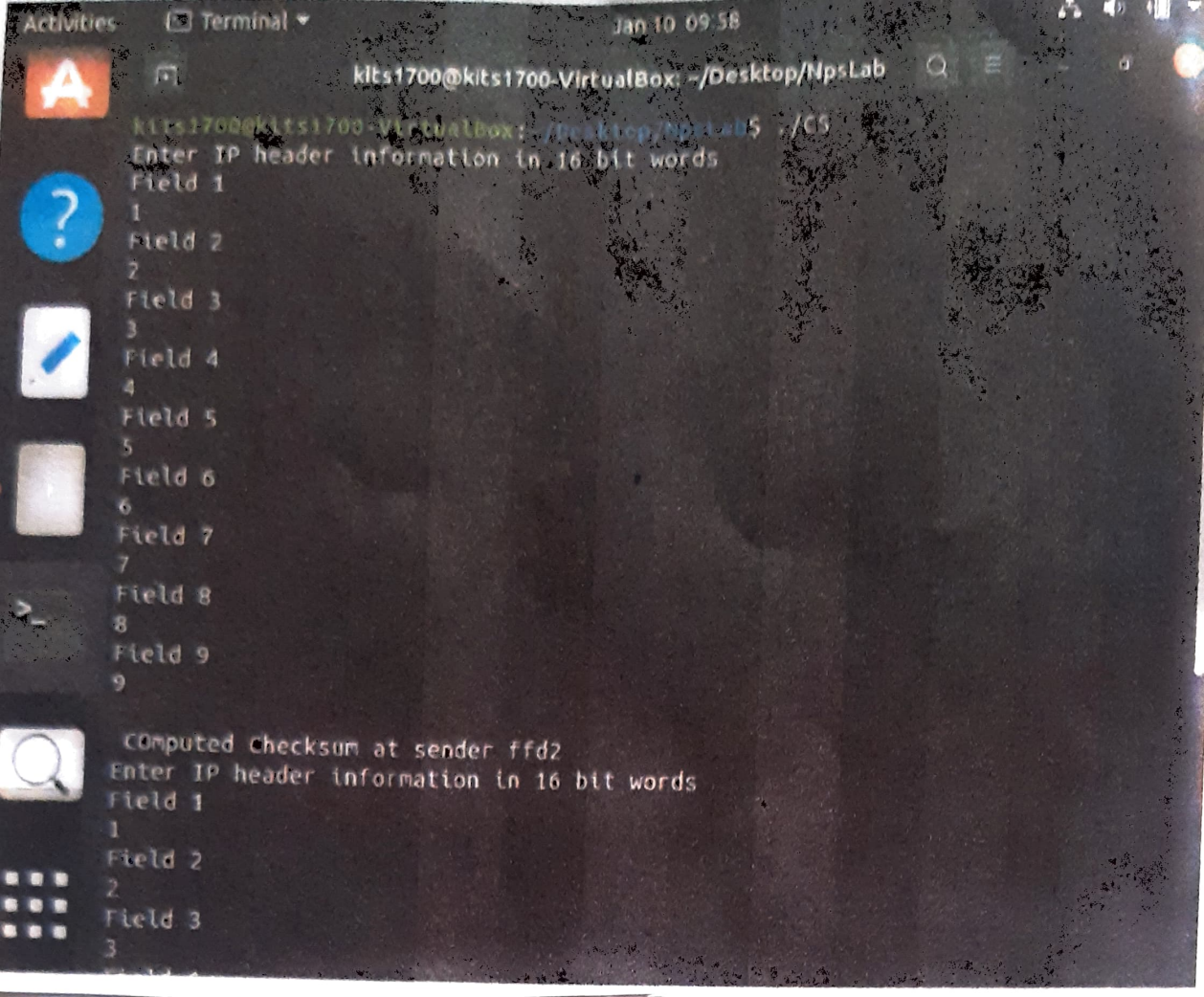
```
//Diffie-Hellman Key exchange
#include<stdio.h>
#include<math.h>
long long int power(long long int a, long long int b, long long int P)
{ if(b==1)
    return a;
  else
   return(((long long int)pow(a,b))%P);
}
```

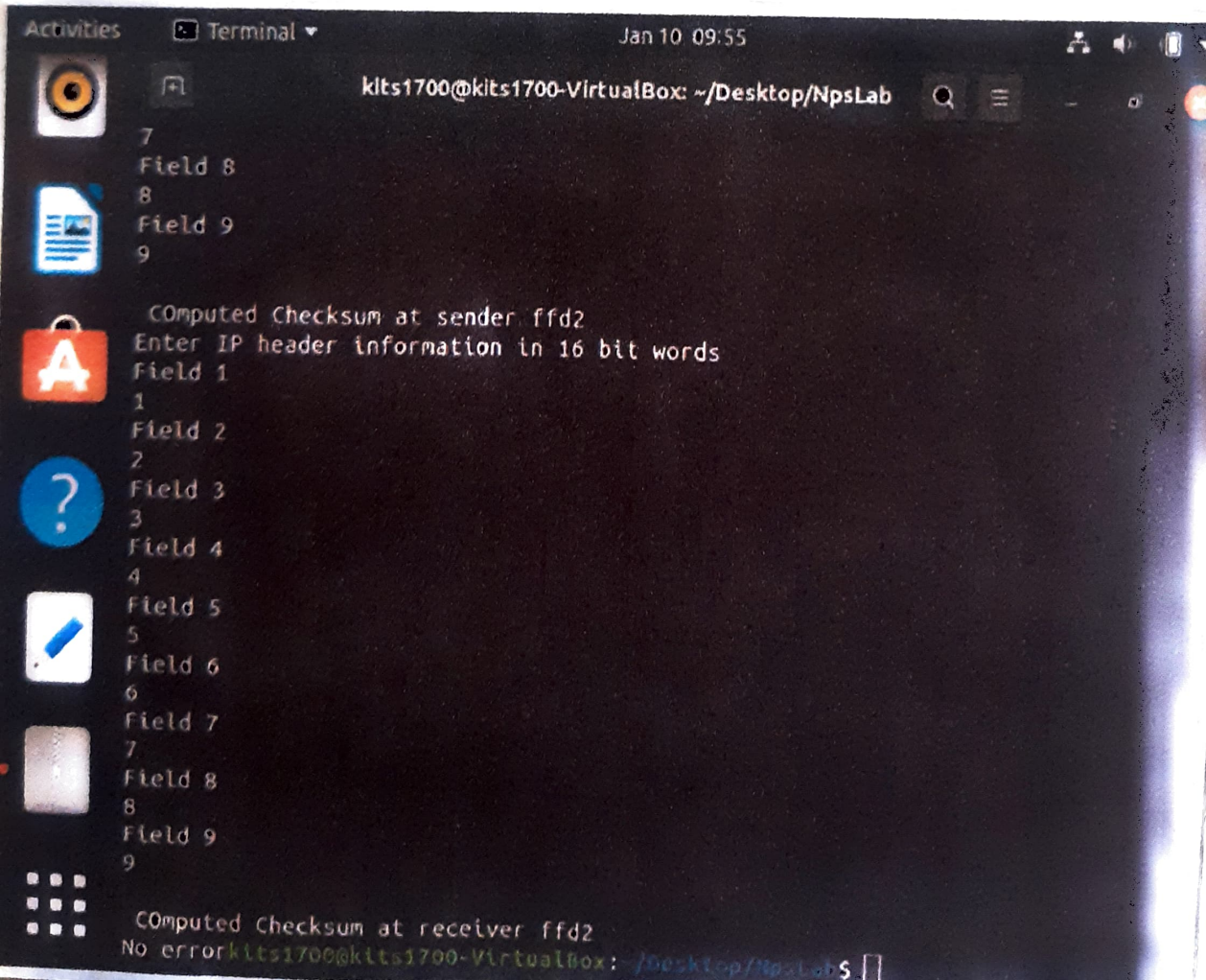
Teacher's Signature: _____

EXPT NO | NAME | Page No. | Date | YOUVA

```
int main()
{ long long int P, G, x, a, y, b, ka, kb;
  P=23;
  printf(P);
  G=9
  printf(G);
  a=4;
  x=power(G, a, P);
  b=3;
  y=power(G, b, P);

  ka = power(y, a, P);
  kb = power(x, b, P);
  printf("Secret info/key for Alice is %lld\n", ka);
  printf("Secret info/key for Bob is %lld\n", kb);

  return 0;
}
```

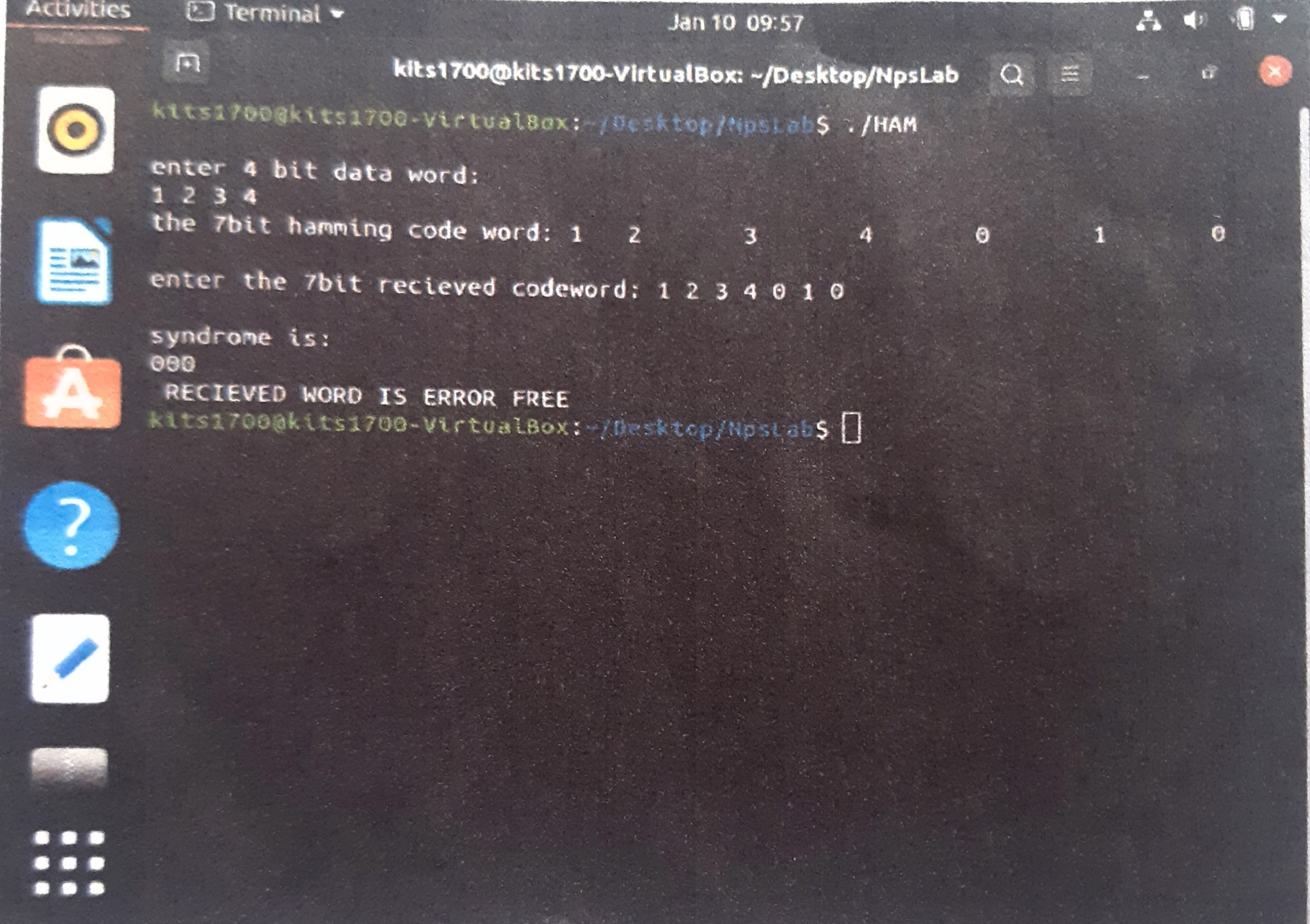
Teacher's Signature:


Activities
Terminal
Jan 10 09:58
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab$ ./CS
Enter IP header information in 16 bit words
Field 1
1
Field 2
2
Field 3
3
Field 4
4
Field 5
5
Field 6
6
Field 7
7
Field 8
8
Field 9
9
COmputed Checksum at sender ffd2
Enter IP header information in 16 bit words
Field 1
1
Field 2
2
Field 3
3


Activities
Terminal
Jan 10 09:55
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
7
Field 8
8
Field 9
9
COmputed Checksum at sender ffd2
Enter IP header information in 16 bit words
Field 1
1
Field 2
2
Field 3
3
Field 4
4
Field 5
5
Field 6
6
Field 7
7
Field 8
8
Field 9
9
COmputed Checksum at receiver ffd2
No errorkits1700@kits1700-VirtualBox: ~/Desktop/NpsLab$

3) Write a program to determine error detection & correction concept using Checksum and hamming code.

```
//Checksum
#include<stdio.h>
unsigned fields[10];
unsigned short checksum()
{ int i;
  int sum=0;
  printf("Enter IP header information in 16 bit words\n");
  for(i=0;i<9;i++)
  { printf("Field%d\n",i+1);
    scanf("%x",&fields[i]);
    sum = sum + (unsigned short) fields[i];
  while(sum>>16)
  sum = (sum & 0XFFFF) + (sum>>16); }

  sum = ~sum;
  return(unsigned short)sum; }

int main()

{ unsigned short result1, result2;
  //Sender
  result1 = checksum();
  printf("\n Computed Checksum at sender %x\n", result1);
  //Receiver
  result2 = checkSum();
```


Activities
Terminal
Jan 10 09:57
kits1700@kits1700-VirtualBox: ~/Desktop/NpsLab
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$ ./HAM
enter 4 bit data word:
1 2 3 4
the 7bit hamming code word: 1 2 3 4 0 1 0
enter the 7bit recieved codeword: 1 2 3 4 0 1 0
syndrome is:
000
RECIEVED WORD IS ERROR FREE
kits1700@kits1700-VirtualBox:~/Desktop/NpsLab$

```
printf("\n Computed Checksum at receiver:%x\n", result2);
if(result1 == result2)
 printf("No error");
else
 printf("Error in data received.");
}
```

```
// Hamming Code
#include<stdio.h>
#include<stdlib.h>
int main()
{ int a[4], b[4], r[3], s[3], i, q[3], c[7];
 printf("\n Enter 4 bit data word:\n");
 for(i=3; i>=0; i--)
{ scanf("%d", &a[i]); }
r[0] = (a[3] + a[1] + a[0]) %2;
r[1] = (a[0] + a[2] + a[3]) %2;
r[2] = (a[1] + a[2] + a[3]) %2;
printf("\n The 7-bit hamming code word:\n");
for(i=3; i>=0; i--)
{ printf("%d\t", a[i]); }
for(i=2; i>=0; i--)
{ printf("%d\t", r[i]); }
printf("\n");
printf("\n Enter 7 bit received codeword:");
for(i=7; i>0; i--)
 scanf("%d", &c[i]);
```

Teacher's Signature

```
b[3]=b[7]; b[2]=c[6]; b[1]=c[5]; b[0]=c[4]; r[2]=c[3]; r[1]=c[2];
r[0]=c[1];

s[0]=(b[0]+b[1]+b[3]+r[0])%2;
s[1]=(b[0]+b[2]+b[3]+r[1])%2;
s[2]=(b[1]+b[2]+b[3]+r[2])%2;
printf("\n Syndrome is:\n");
for(i=2;i>=0;i--)
printf("%d", s[i]);
if((s[2]==0)&&(s[1]==0)&&(s[0]==0))
printf("\n Received Word is Error free\n");
if((s[2]==1)&&(s[1]==1)&&(s[0]==1))
{ printf("\n error in received codeword, position-7th bit from right\n");
  if(c[7]==0)
  c[7]=1;
  else
  c[7]=0;
printf("\n corrected codeword is\n");
for(i=7;i>0;i--)
printf("%d\t", c[i]);
}
if((s[2]==1)&&(s[1]==1)&&(s[0]==0))
{printf("\n Error in received codeword, position-6th bit from right\n");
  if(c[6]==0)
  c[6]=1;
  else
  c[6]=0;
printf("\n Codeword corrected\n");
```

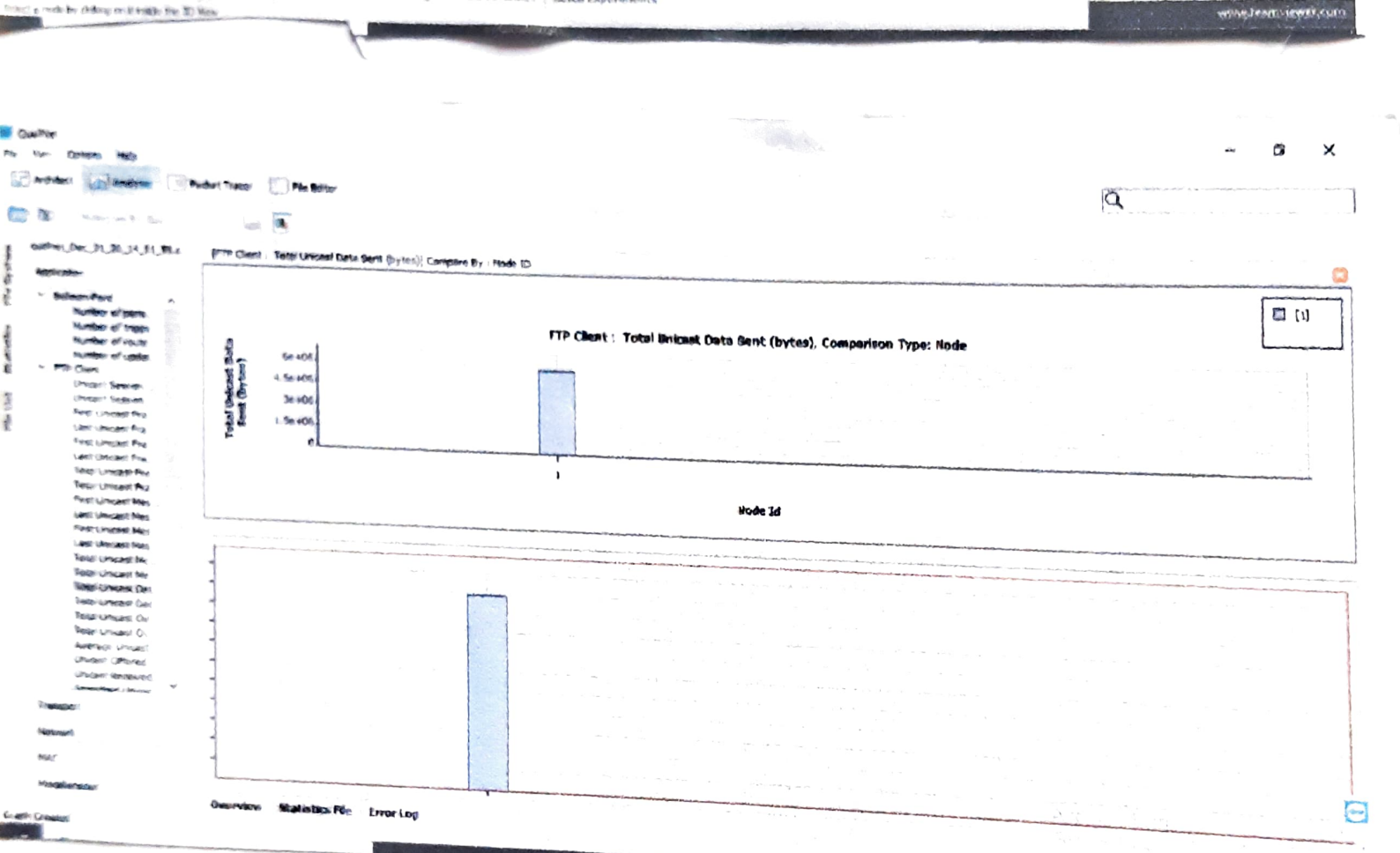
Teacher's Signature: _____

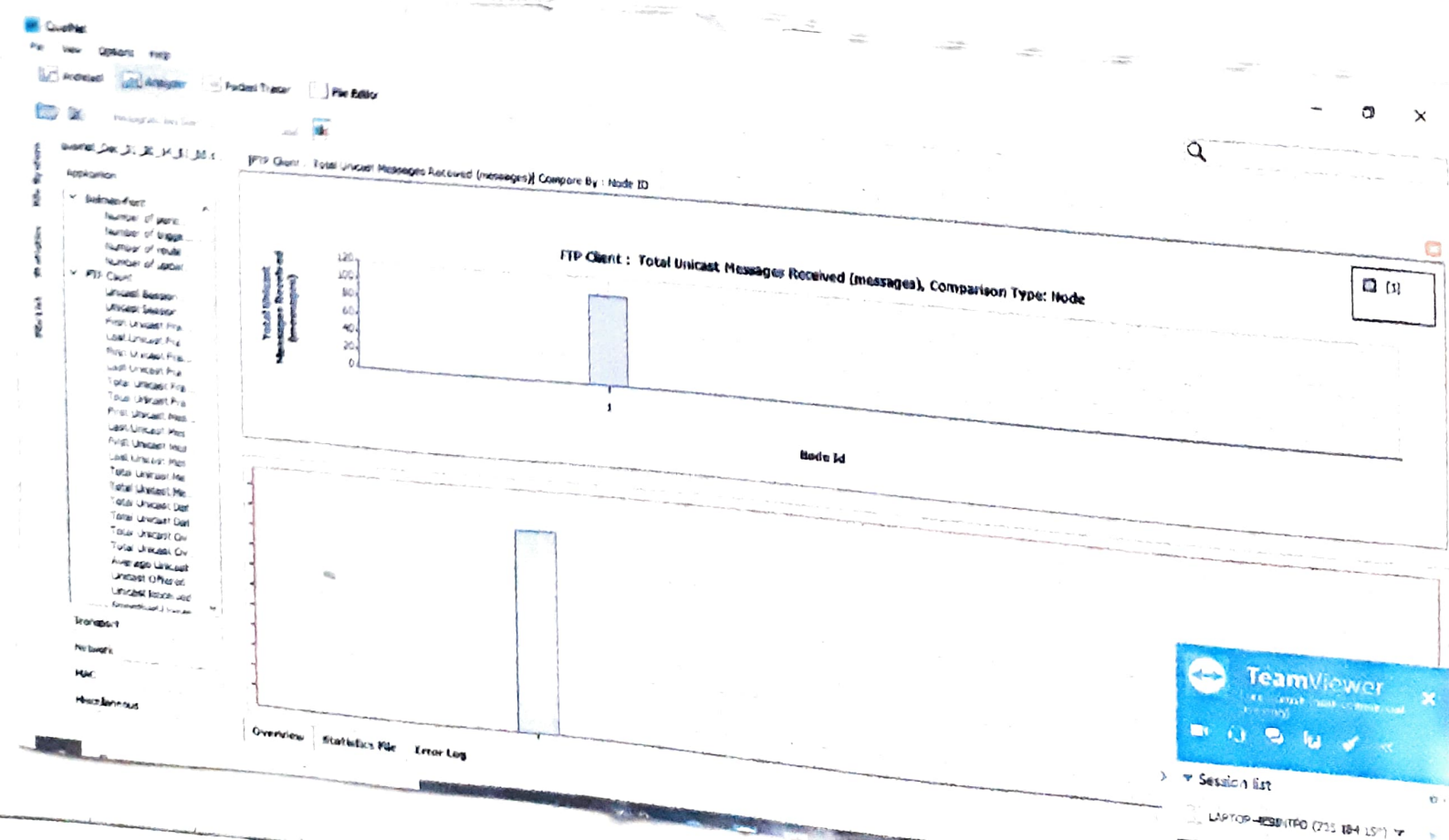
```
for(i=7;i>0;i--)
printf("%d\t",c[i]);
if((s[2]==1)&&(s[1]==0)&&(s[0]==1))
{ printf("\n Error in received codeword, position-5th bit from right\n");
  if(c[5]==0)
    c[5]=1;
  else
  c[5]=0; printf("\n Corrected codeword is \n");
for(i=7;i>0;i--)
printf("%d\t",c[i]);}

if((s[2]==1)&&(s[1]==0)&&(s[0]==0))
{printf("\n Error in received codeword, Position-4th bit from right\n");
  if(c[4]==0)
    c[4]=1;
    else
    c[4]=0;
printf("\n Corrected codeword is \n");
for(i=7;i>0;i--)
  printf("%d\t",c[i]);
    }
if((s[2]==0)&&(s[1]==1)&&(s[0]==1))
{ printf("\n Error in received codeword, Position-3rd bit from right\n");
  if(c[3]==0)
    c[3]=1;
else
c[3]=0; printf(
printf("\n Corrected codeword\n");
```

Teacher's Signature:

```
for(i=7;i>0;i--)
 printf("%d\t",c[i]);}

if((s[2]==0)&&(s[1]==1)&&(s[0]==0))
{ printf("\nError in received codeword, Position -2nd bit from right\n");
   if(c[2]==0)
     c[2]=1;
   else
   c[2]=0.
printf("\nCorrected codeword is\n");
for(i=7;i>0;i--)
 printf("%d\t",c[i]);
                                    }
if((s[2]==0)&&(s[1]==0)&&(s[0]==1))
{ printf("\nError in received codeword, Position -1st bit from
      right\n");
  if(c[1]==0)
    c[1]=1;
  else
    c[1]=0;
printf("\nCorrected codeword is\n");
 for(i=7;i>0;i--)
    printf("%d\t",c[i]);
          }
 return(1);}
```

Teacher's Signature:


FTP Client : Total Unicast Data Sent (bytes), Comparison Type: Node
[1]
Total Unicast Data Sent (bytes)
Node Id
Overview
Statistics File
Error Log


FTP Client : Total Unicast Messages Received (messages), Comparison Type: Node
[1]
Total Unicast Messages Received (messages)
Node Id
Overview
Statistics File
Error Log
TeamViewer
Session list

EXPT NO | NAME | Page No. | Date | YOUVA

Part B

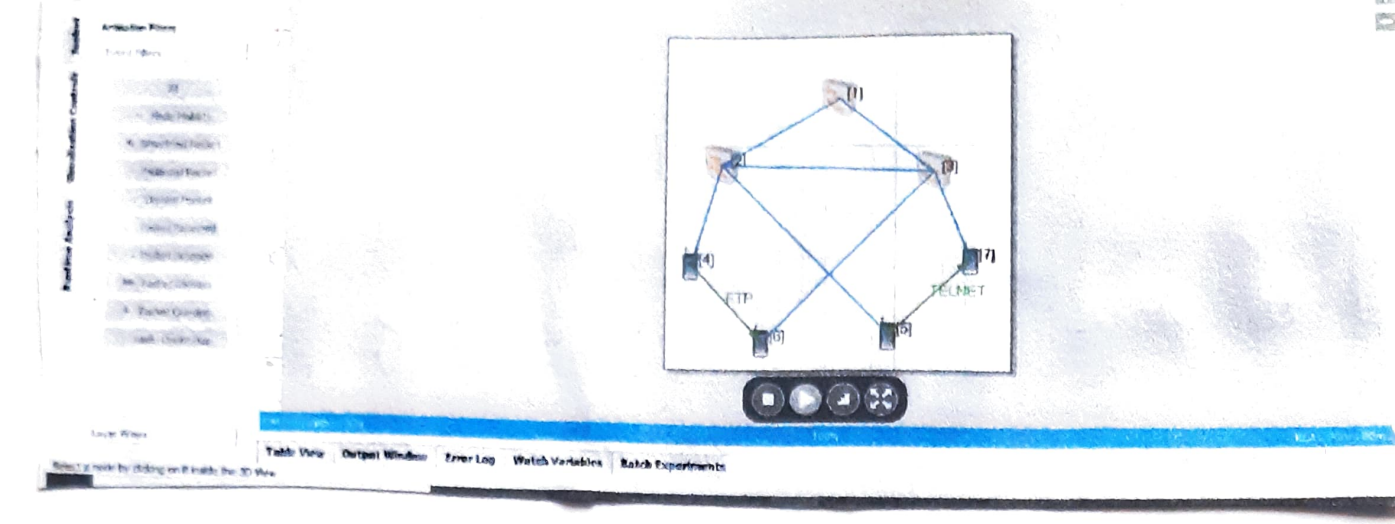
1. Set up an IEEE 802.3 network with a) hub b) switch c) Hierarchy of switch. Apply FTP, Telnet applications between nodes. Vary the bandwidth, queue size and observe the packet drop.

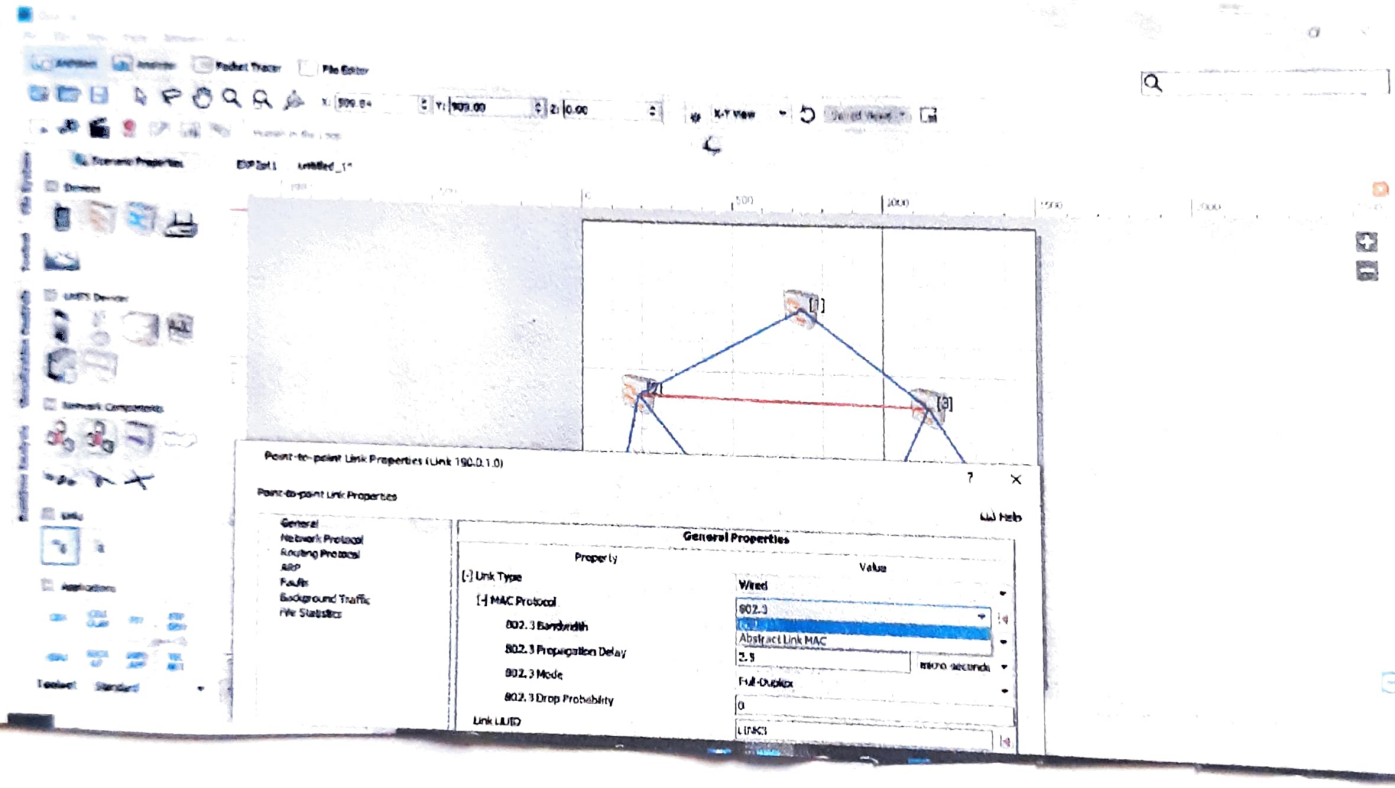
① → Design of the required layout with one hub/switch connected to four devices via connecting lines.

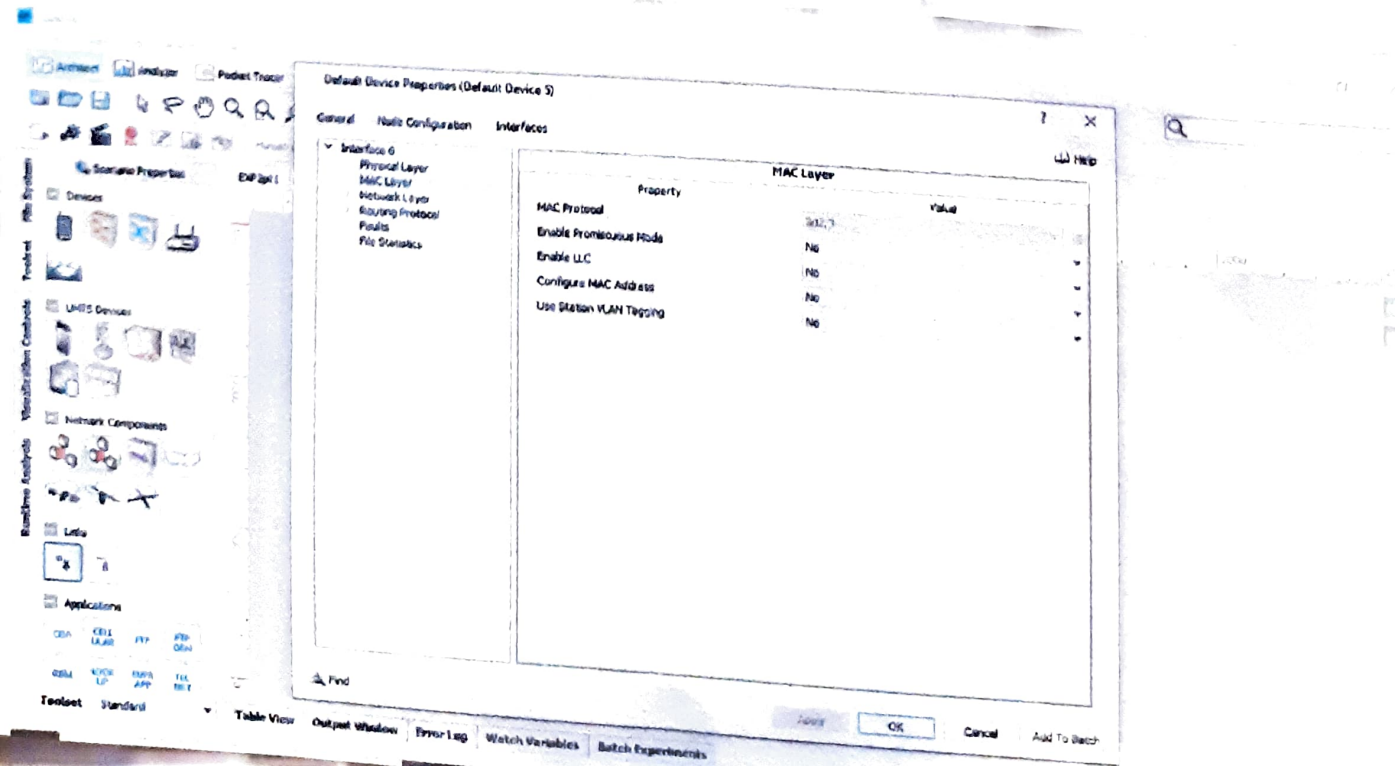
② → Graph of Total Unicast Data Sent vs Node ID. (Bytes)

③ → Graph of Total Unicast Data Received vs Node ID (Bytes)

Teacher's Signature: _____


[1]
[2]
[3]
[4]
[5]
[6]
[7]
FTP
TELNET
Table View
Output Window
Error Log
Watch Variables
Batch Experiments


Point-to-point Link Properties (Link 190.0.1.0)
Point-to-point Link Properties
Help
General
Network Protocol
Routing Protocol
ARP
Faults
Background Traffic
Pkt Statistics
General Properties
Property
Value
[-] Link Type
Wired
[-] MAC Protocol
802.3
802.3 Bandwidth
802.3 Propagation Delay
802.3 Mode
802.3 Drop Probability
Link UUID
Abstract Link MAC
Full-Duplex


Default Device Properties (Default Device 5)
General
Node Configuration
Interfaces
Interface 0
Physical Layer
MAC Layer
Network Layer
Routing Protocol
Faults
Pkt Statistics
MAC Layer
Property
Value
MAC Protocol
Enable Promiscuous Mode
No
Enable LLC
No
Configure MAC Address
No
Use Station VLAN Tagging
No
Find
OK
Cancel
Add To Batch
Table View
Output Window
Error Log
Watch Variables
Batch Experiments

EXPT. NO. | NAME | Page No.: | Date: | YOUVA

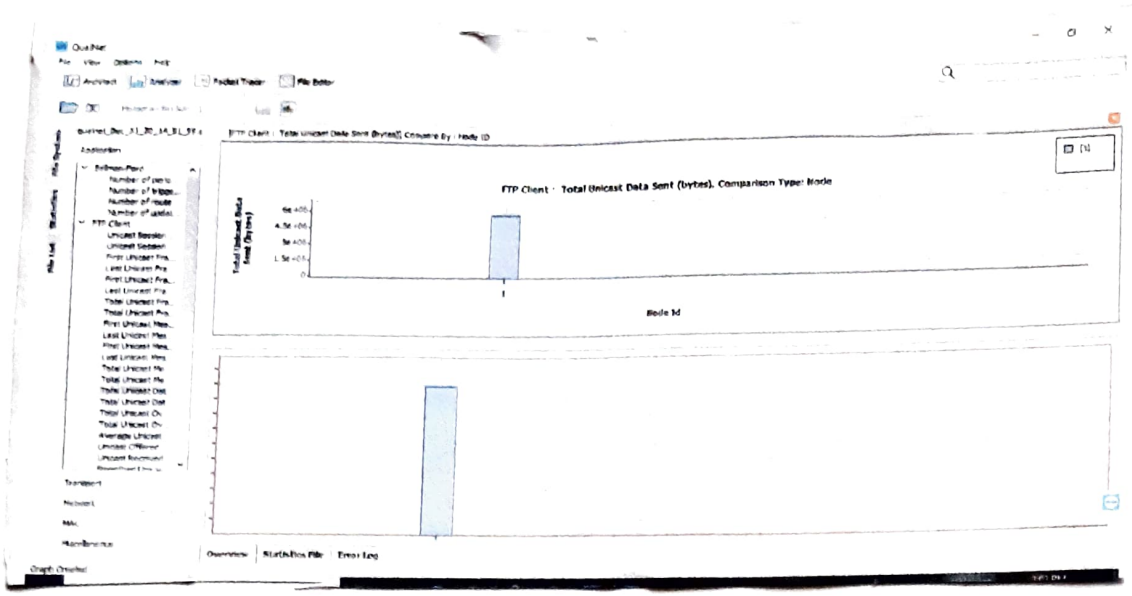
M T W T F S S

④ → Design layout of three switches connected to each other and to four devices, out of which FTP is applied on 4 and 6 devices and TELNET on 5 and 7 devices

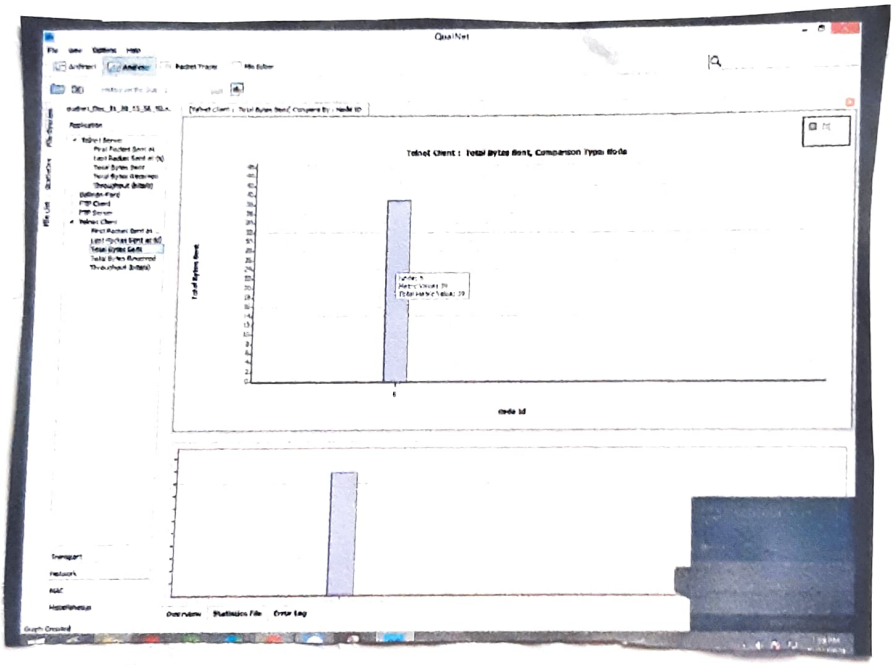
⑤ → Configuring the connecting lines to 802.3 MAC Protocol

⑥ → Checking the configuration properties of the devices and ensuring it is 802.3, i.e. its MAC Protocol.

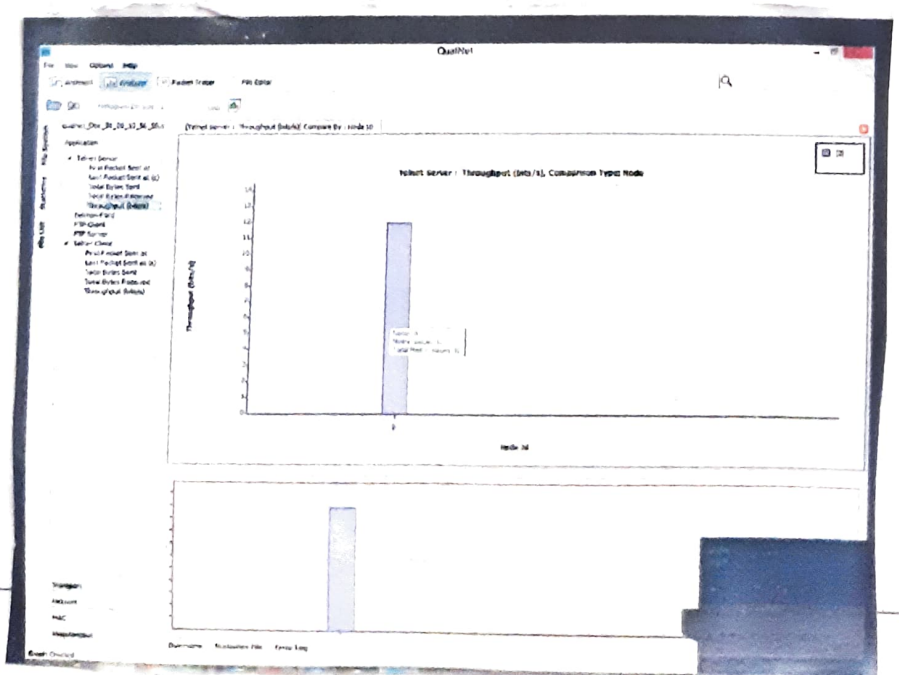
Teacher's Signature: _____


QualNet
FTP Client : Total Unicast Data Sent (bytes), Comparison Type: Node
Node Id
Overview
Statistics File
Error Log

7


QualNet
Telnet Client : Total Bytes Sent, Comparison Type: Node
Node Id
Overview
Statistics File
Error Log

8


QualNet
Telnet Server : Throughput (bits/s), Comparison Type: Node
Node Id

9

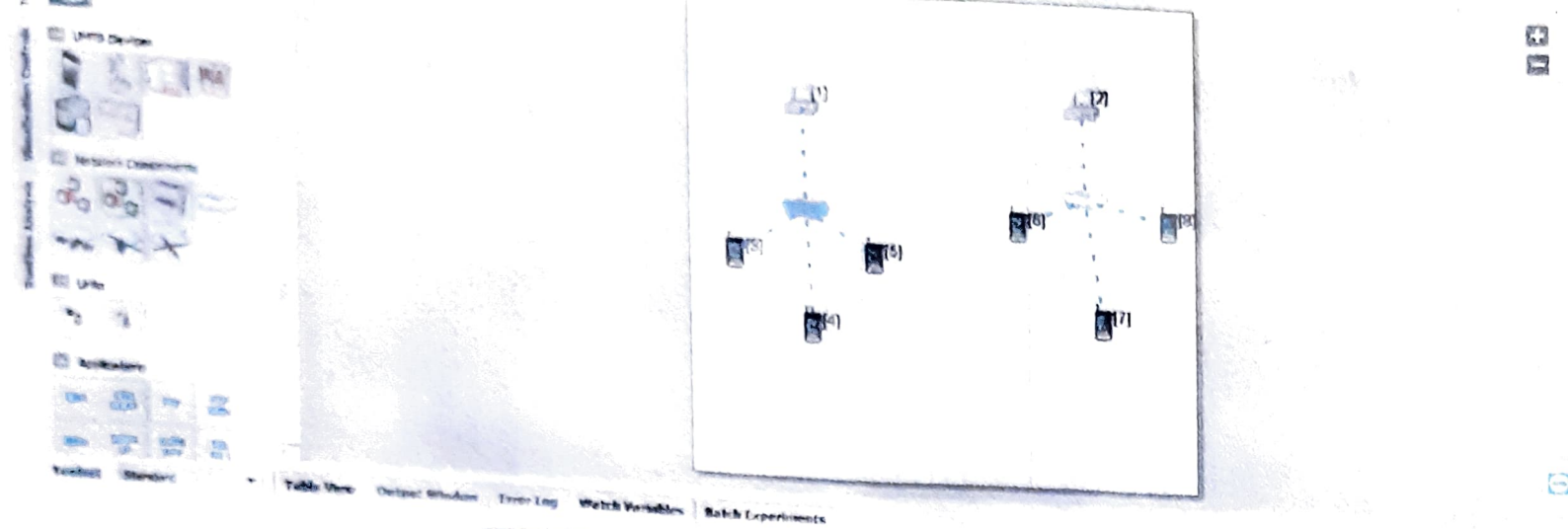
EXPT NO | NAME | Page No. | Date | YOUVA

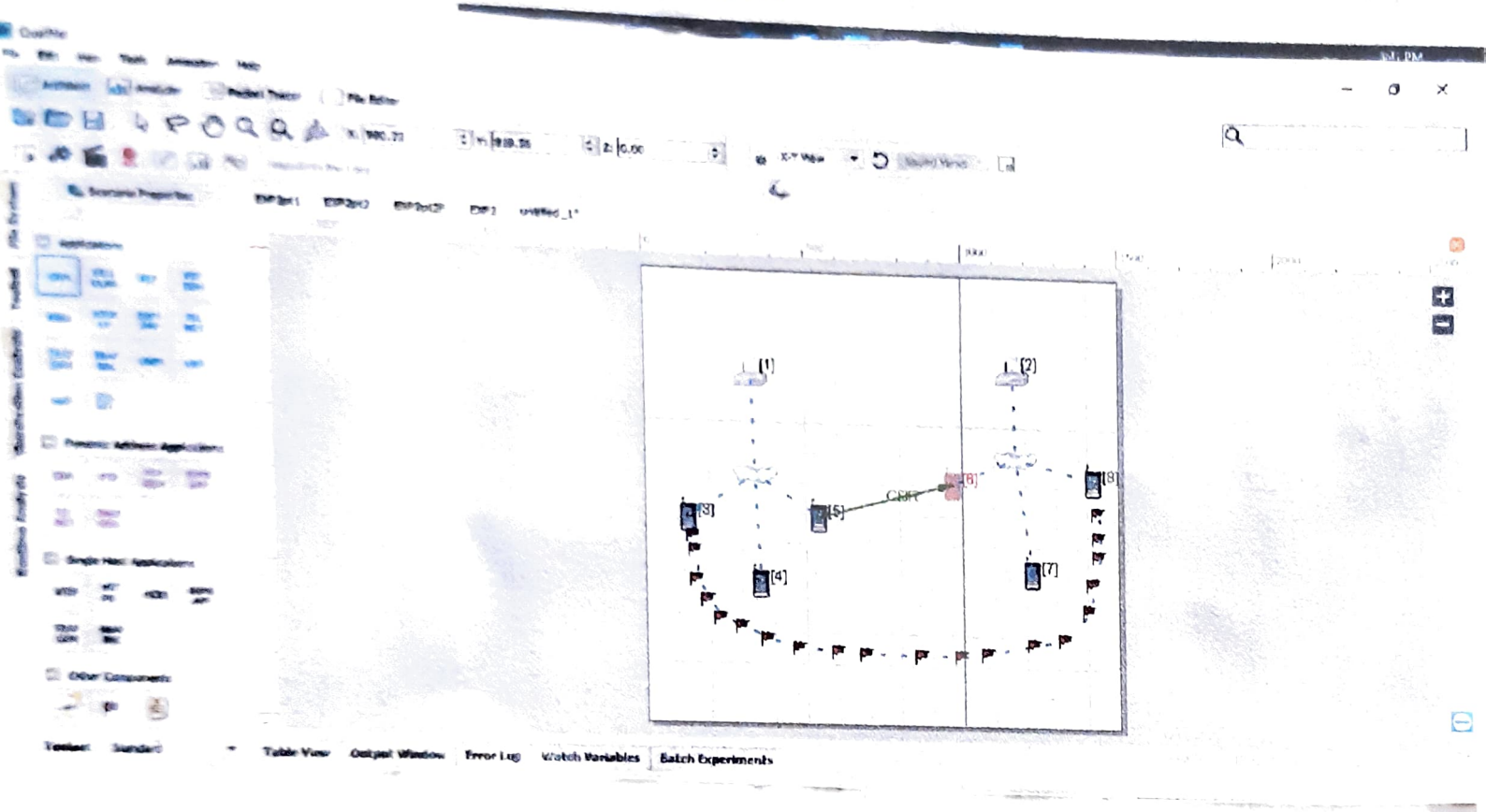
(7) → Graph of FTP Client's Total Unicast Data Sent (Bytes) vs Node ID

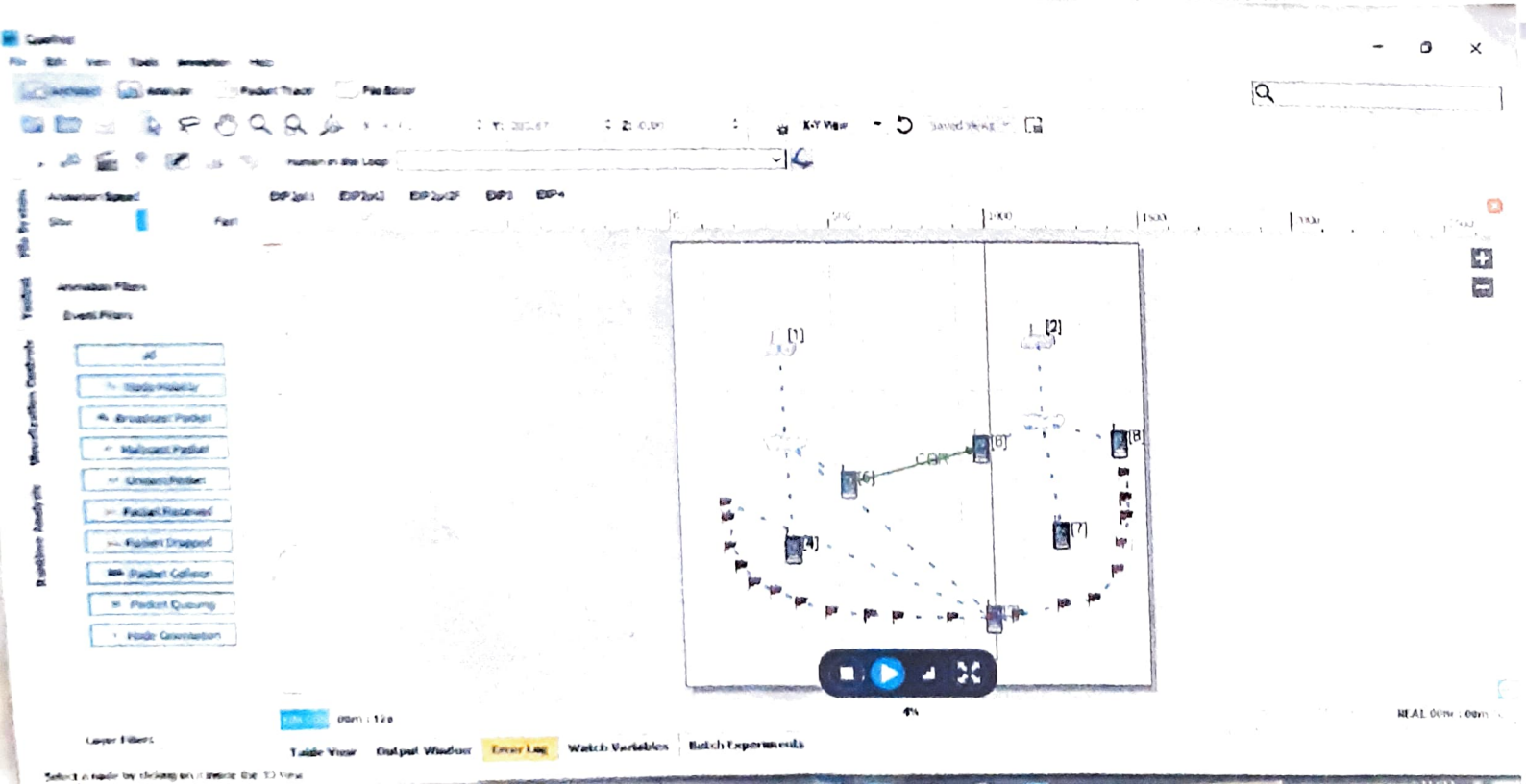
(8) → Graph of TELNET Clients : Total Bytes Sent vs Node ID

(9) → Graph of TELNET Servers : Throughput (bits/s) vs Node ID.

Teacher's Signature: _____


[1]
[2]
[3]
[5]
[6]
[8]
[4]
[7]
Table View
Output Window
Error Log
Watch Variables
Batch Experiments


[1]
[2]
[3]
[5]
CBR
[6]
[8]
[4]
[7]
Table View
Output Window
Error Log
Watch Variables
Batch Experiments


[1]
[2]
[6]
CBR
[8]
[4]
[7]
Table View
Output Window
Error Log
Watch Variables
Batch Experiments

EXPT NO | NAME | Page No.: | Date | YOUVA

M T W T F S S

2) ~~Setup a wireless sensor networks with atleast 2 device~~

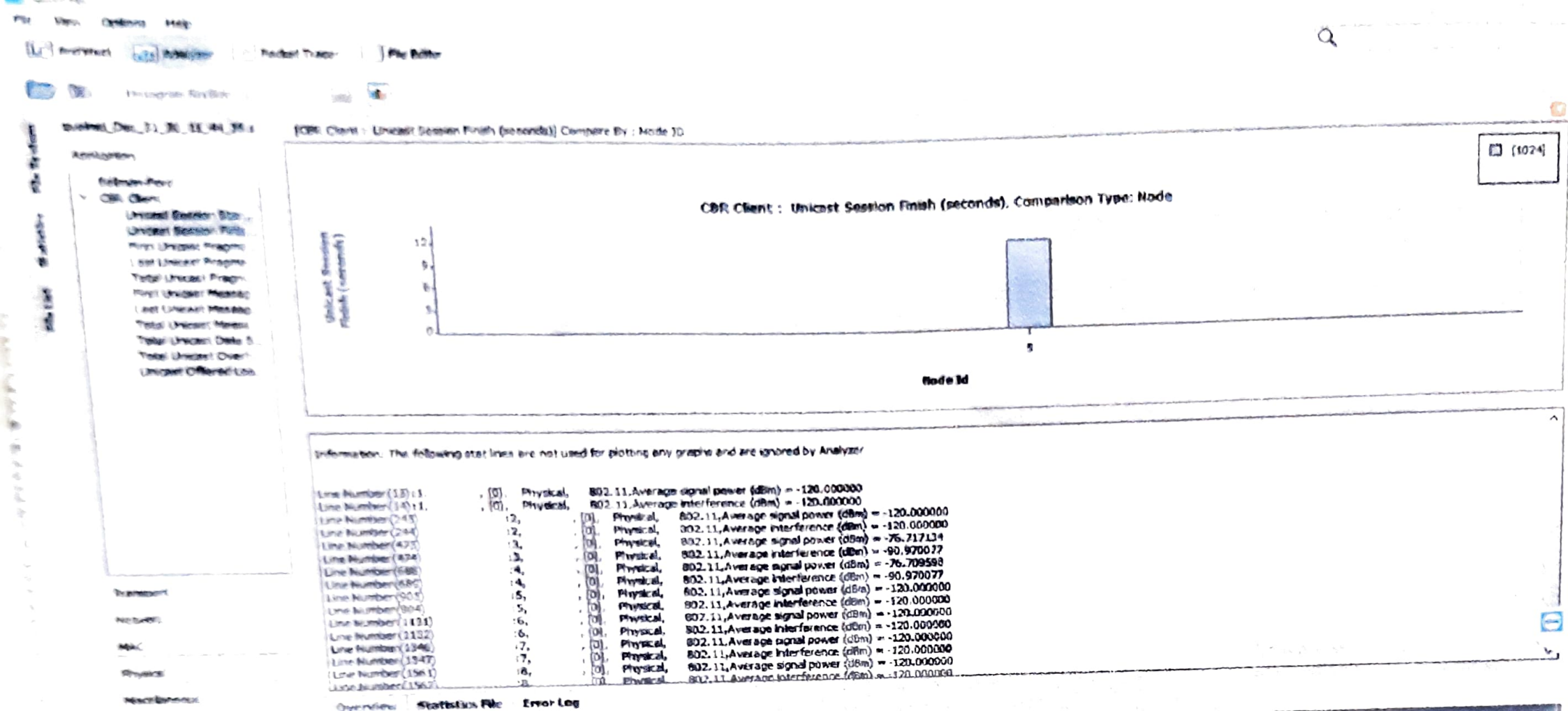
2) Set up an IEEE 802.11 network with at least two access points. Apply the CBR, VBR applications between devices belonging to same access points and different access points. Provide roaming of any device. Vary the number of access points and devices. Find out the delay in MAC layer, packet drop probability.

(1) → Design layout of the required network with 2 access points.

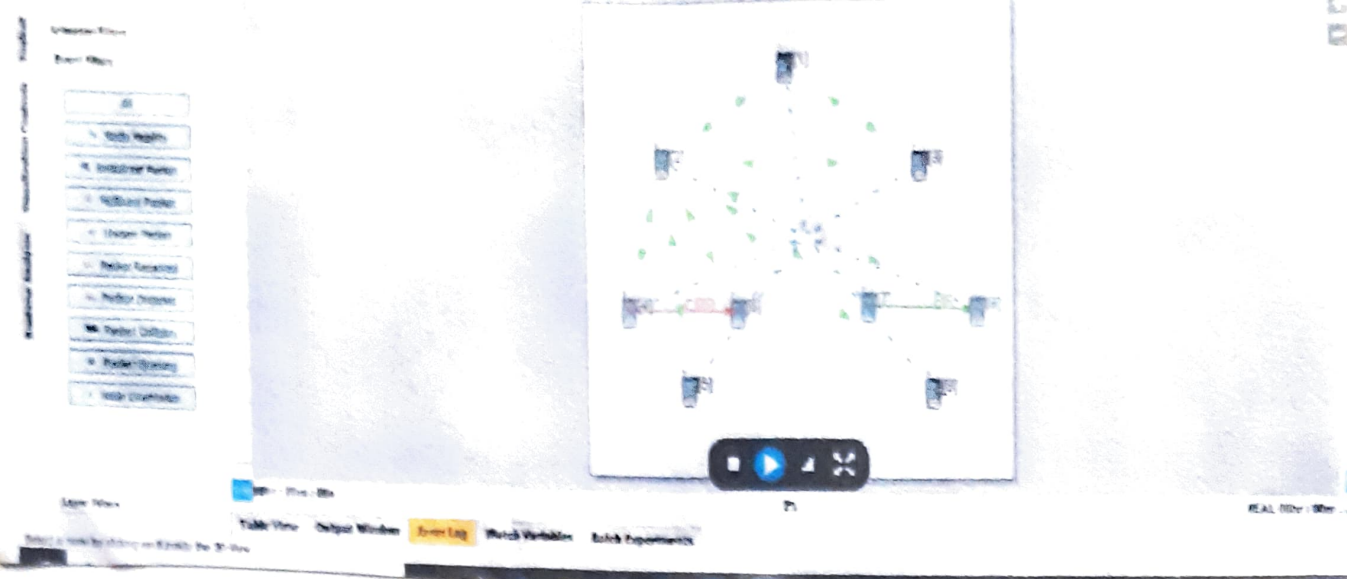
(2) → Application of CBR protocol to 5 and 6 devices and creating a path.

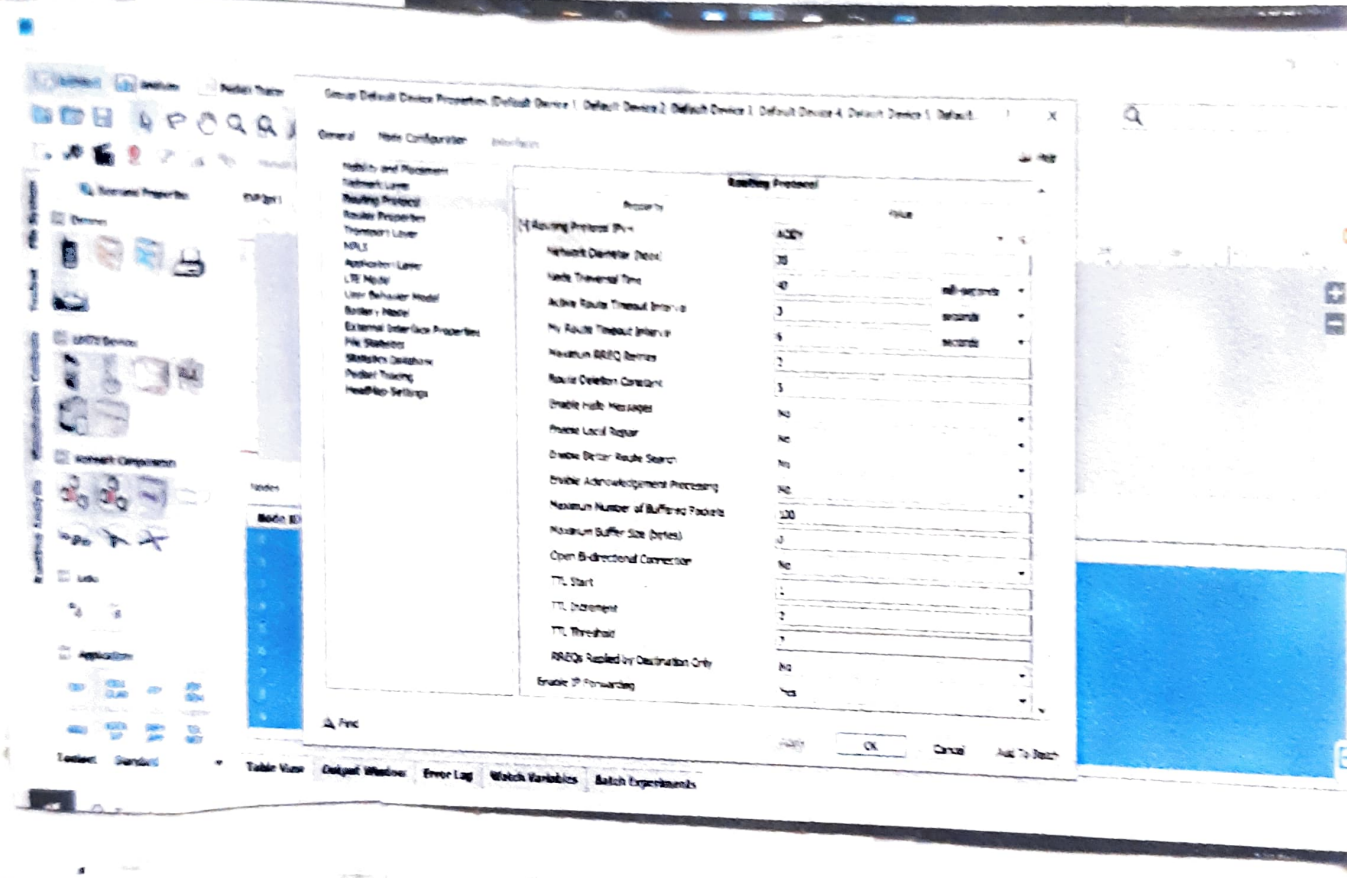
(3) → Watching the packet transfer in the network.

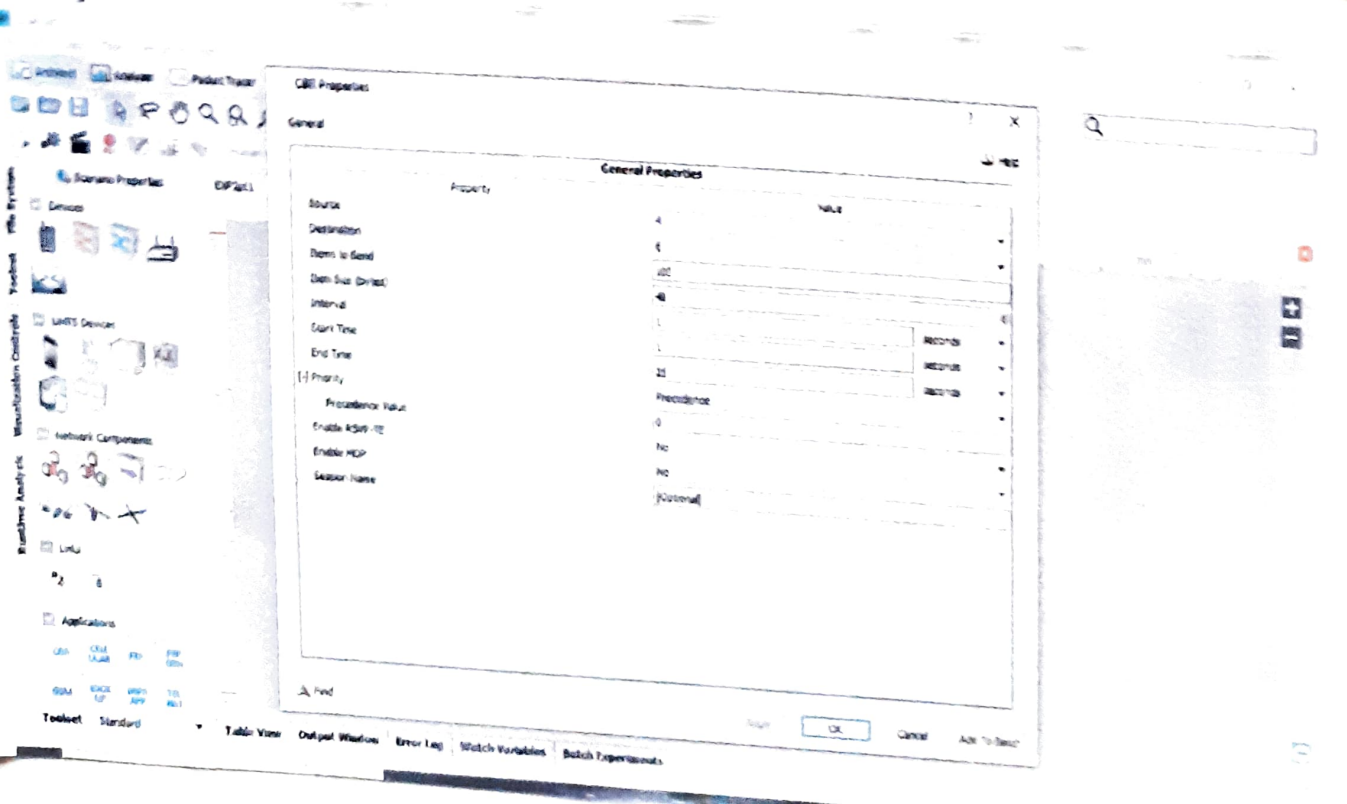
Teacher's Signature


[CBR Client : Unicast Session Finish (seconds)] Compare By : Node ID
[1024]
CBR Client : Unicast Session Finish (seconds), Comparison Type: Node
Unicast Session Finish (seconds)
12
9
6
3
0
5
Node Id
Information: The following stat lines are not used for plotting any graphs and are ignored by Analyzer
Line Number (13) : 1 , [0], Physical, 802.11,Average signal power (dBm) = -120.000000
Line Number (14) : 1 , [0], Physical, 802.11,Average interference (dBm) = -120.000000
Line Number (243) : 2 , [0], Physical, 802.11,Average signal power (dBm) = -120.000000
Line Number (244) : 2 , [0], Physical, 802.11,Average interference (dBm) = -120.000000
Line Number (423) : 3 , [0], Physical, 802.11,Average signal power (dBm) = -76.717134
Line Number (424) : 3 , [0], Physical, 802.11,Average interference (dBm) = -90.970077
Line Number (688) : 4 , [0], Physical, 802.11,Average signal power (dBm) = -76.709598
Line Number (689) : 4 , [0], Physical, 802.11,Average interference (dBm) = -90.970077
Line Number (903) : 5 , [0], Physical, 802.11,Average signal power (dBm) = -120.000000
Line Number (904) : 5 , [0], Physical, 802.11,Average interference (dBm) = -120.000000
Line Number (2131) : 6 , [0], Physical, 802.11,Average signal power (dBm) = -120.000000
Line Number (2132) : 6 , [0], Physical, 802.11,Average interference (dBm) = -120.000000
Line Number (2346) : 7 , [0], Physical, 802.11,Average signal power (dBm) = -120.000000
Line Number (2347) : 7 , [0], Physical, 802.11,Average interference (dBm) = -120.000000
Line Number (2561) : 8 , [0], Physical, 802.11,Average signal power (dBm) = -120.000000
Line Number (2562) : 8 , [0], Physical, 802.11,Average interference (dBm) = -120.000000
Overview
Statistics File
Error Log

④ → Graph of CBR clients Unicast session Finish (seconds) vs Node ID




Routing Protocol
Property
Value
Table View
Output Window
Error Log
Watch Variables
Batch Experiments

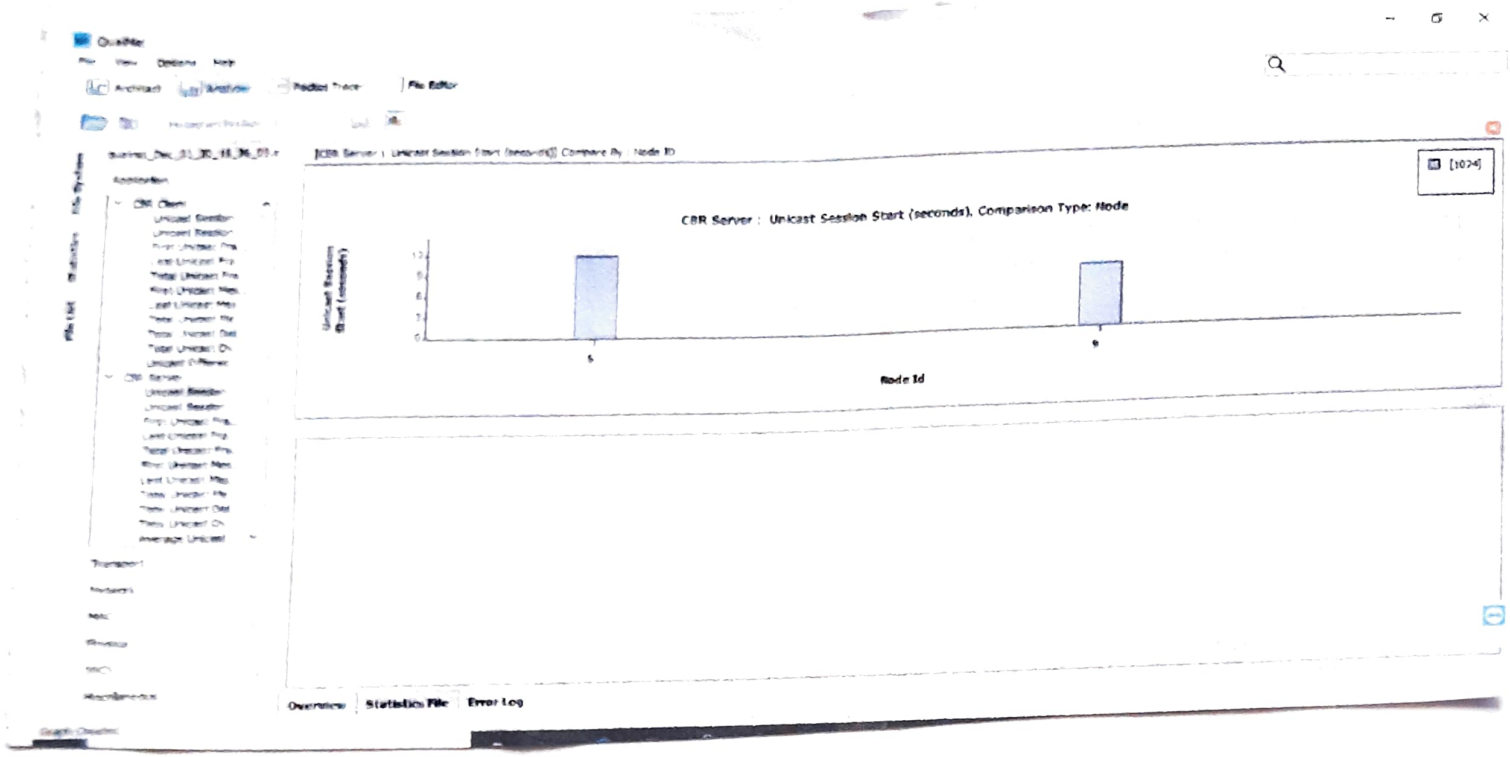

General Properties
Property
Value
Table View
Output Window
Error Log
Watch Variables
Batch Experiments

3) Setup a wireless sensor networks with atleast two device co-ordinators and nodes. Provide Constant Bit Rate (CBR), Variable Bit Rate (VBR) application between several nodes. Increase the number of co-ordinators and nodes in the same area and observe the performance at physical and MAC layers.

(1) → Design layout for the required network with one main coordinator and two PAN coordinators.

(2) → Configuration properties for all the devices in the network.

(3) → Configuration properties for all devices in the network.


CBR Server : Unicast Session Start (seconds), Comparison Type: Node
Node Id
Overview
Statistics File
Error Log

4

④ → Graph of CBR Server : Unicast Session Start (seconds) vs Node ID.