Aditya Pradhan CSE D 180905350

Q1. Write a TCP client which sends a string to a server program. Server displays the string along with client IP and ephemeral port number. Server then responds to the client by echoing back the string in uppercase. The process continues until one of them types “QUIT”.

Code

Server Code

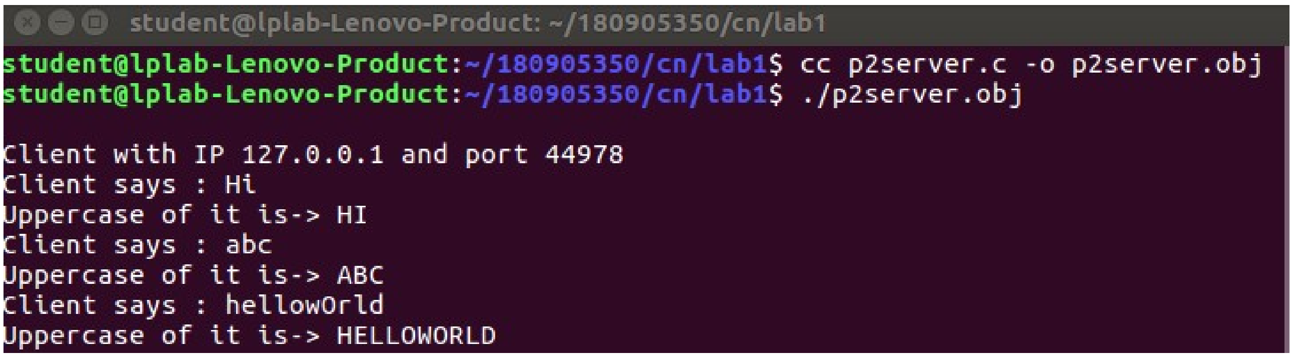
```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <string.h>
#include <ctype.h>
#define PORT 7000
#define sa struct sockaddr

int main()
{
    int sockid = socket(AF_INET, SOCK_STREAM, 0);
    int m = 0, n = 0, data_len, sockid_new;
    char buff[100];
    unsigned int len;
    struct sockaddr_in serv_addr, cli_addr;
    bzero(&serv_addr, sizeof(serv_addr));
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(PORT);
    serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
    if (bind(sockid, (sa *)&serv_addr, sizeof(serv_addr)) < 0)
    {
        printf("Could not bind socket");
        exit(0);
    }
    listen(sockid, 5);
    len = sizeof(cli_addr);
    sockid_new = accept(sockid, (sa *)&cli_addr, &len);
    printf("\nClient with IP %s and port %d ", inet_ntoa(cli_addr.sin_addr), ntohs(cli_addr.sin_port));
    for (;;)
    {
        bzero(buff, sizeof(buff));
        read(sockid_new, buff, sizeof(buff));
        printf("\nClient says : %s", buff);
        for (int i = 0; i < strlen(buff); i++)
            buff[i] = toupper(buff[i]);
        // write(sockid_new, buff, sizeof(buff));
        printf("\nUppercase of it is-> %s", buff);
        if (strncmp(buff, "QUIT", 4) == 0)
            break;
    }
    printf("\nServer Shutting Down\n");
    close(sockid);
    return 0;
}
```

Client Code

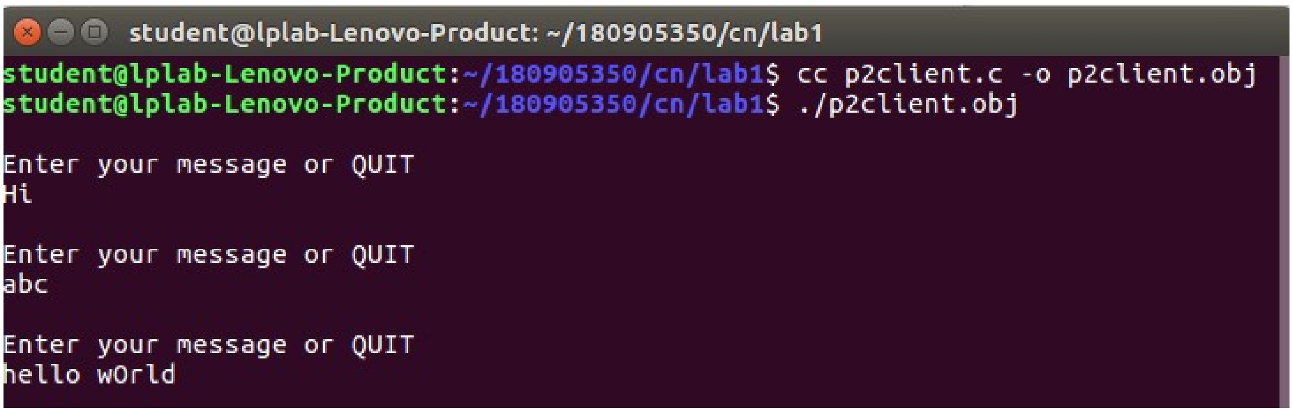
```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <string.h>
#define PORT 7000
#define sa struct sockaddr

int main()
{
    int sockid = socket(AF_INET, SOCK_STREAM, 0);
    int data_len;
    unsigned int len;
    struct sockaddr_in serv_addr, temp;
    bzero(&serv_addr, sizeof(serv_addr));
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(PORT);
    serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
    connect(sockid, (sa *)&serv_addr, sizeof(serv_addr));
    char buff[100], line[100];
    for (;;)
    {
        printf("\nEnter your message or QUIT\n");
        bzero(line, sizeof(line));
        bzero(buff, sizeof(buff));
        scanf("%s", line);
        write(sockid, line, strlen(line));
        // read(sockid, buff, sizeof(buff));
        // printf("\nServer says: %s\n", buff);
        if (strncmp(buff, "QUIT", 4) == 0)
            break;
    }
    printf("\nThank You!");
    close(sockid);
    return 0;
}
```

Aditya Pradhan CSE D 180905350

Output



```
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ cc p2server.c -o p2server.obj
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ ./p2server.obj

Client with IP 127.0.0.1 and port 44978
Client says : Hi
Uppercase of it is-> HI
Client says : abc
Uppercase of it is-> ABC
Client says : helloWOrld
Uppercase of it is-> HELLOWORLD
```



```
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ cc p2client.c -o p2client.obj
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ ./p2client.obj

Enter your message or QUIT
Hi

Enter your message or QUIT
abc

Enter your message or QUIT
hello wOrld
```

Aditya Pradhan CSE D 180905350

Q2) DayTime Server: Where client sends request to time server to send current time. Server responds by sending the current time . [Hint: read man pages of asctime() and localtime()] . Display server process id at client side along with time.

Code

Server

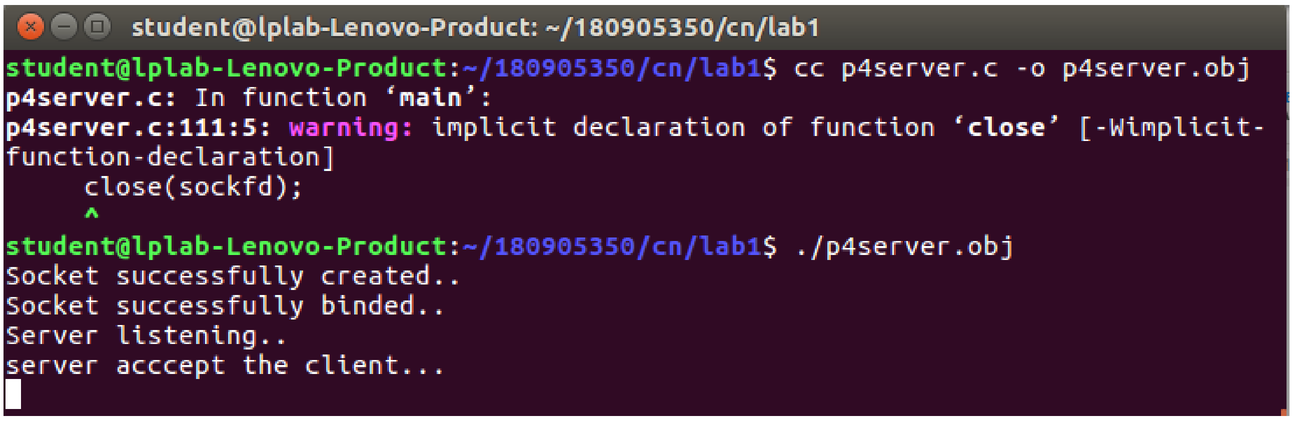
```
#include <arpa/inet.h>
#include <netdb.h>
#include <netinet/in.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <time.h>

#define MAX 80
#define PORT 5005
#define SA struct sockaddr

void servfunc(int sockfd, struct sockaddr_in *cli)
{
   char buff[MAX];
   int n;
   for (;;)
   {
      bzero(buff, sizeof(buff));
      n = recv(sockfd, buff, sizeof(buff), 0);

      buff[n] = '\0';

      if (strcmp(buff, "QUIT") == 0)
      {
         printf("Server Exit...\n");
         break;
      }
      else if (strcmp(buff, "time") == 0)
      {
         time_t rawtime;
         struct tm *info;
         time(&rawtime);
         info = localtime(&rawtime);

         char *str = asctime(info);

         ssize_t size_str = strlen(str);
         n = send(sockfd, str, size_str, 0);
         if (n == -1)
         {
            printf("Error in sending message. Try Again!\n");
```

```
                continue;
            }
        }
        else
        {
            char str[] = "ERROR";
            if (send(sockfd, str, sizeof(str), 0) == -1)
            {
                printf("Error in sending message. Try Again!\n");
                continue;
            }
        }
    }
}
int main()
{
    int sockfd, connfd, len;
    struct sockaddr_in servaddr, cli;

    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1)
    {
        printf("socket creation failed...\n");
        exit(0);
    }
    else
    {
        printf("Socket successfully created..\n");
    }
    bzero(&servaddr, sizeof(servaddr));

    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(PORT);

    if ((bind(sockfd, (SA *)&servaddr, sizeof(servaddr))) != 0)
    {
        printf("socket bind failed...\n");
        exit(0);
    }
    else
    {
        printf("Socket successfully binded..\n");
    }

    if ((listen(sockfd, 5)) != 0)
    {
        printf("Listen failed...\n");
        exit(0);
    }
    else
    {
        printf("Server listening..\n");
    }

    len = sizeof(cli);
    connfd = accept(sockfd, (SA *)&cli, &len);
    if (connfd < 0)
    {
        printf("server acccept failed...\n");
        exit(0);
    }
    else
        printf("server acccept the client...\n");

    servfunc(connfd, (struct sockaddr_in *)&cli);
    close(sockfd);
}
```

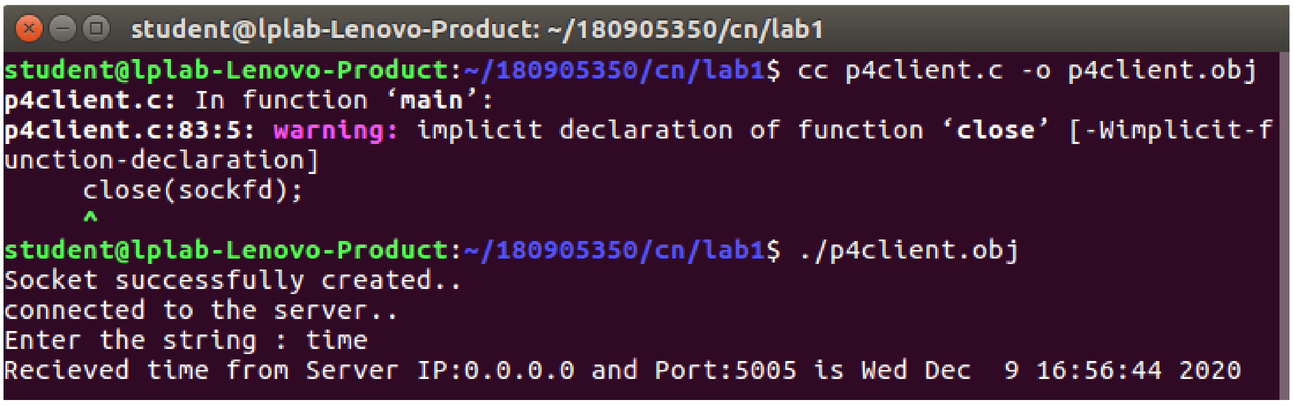
Client

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

#define MAX 80
#define PORT 5005
#define SA struct sockaddr

void clifunc(int sockfd, struct sockaddr_in *cli)
{
    char buff[MAX];
    int n;

    char *client_ip = inet_ntoa(cli->sin_addr);
    int client_port = (int)ntohs(cli->sin_port);
    for (;;)
    {
        bzero(buff, sizeof(buff));
        printf("Enter the string : ");
        n = 0;
        scanf("%[^\n]%*c", buff);
        if (send(sockfd, buff, sizeof(buff), 0) == -1)
        {
            printf("Error in sending message. Try Again!\n");
            continue;
        }
        if (strcmp(buff, "QUIT") == 0)
        {
            printf("Server Closed, client exiting!");
            break;
        }
        bzero(buff, sizeof(buff));

        n = recv(sockfd, buff, sizeof(buff), 0);
        buff[n] = '\0';

        if (strcmp(buff, "ERROR") == 0)
        {
            printf("Wrong time command. Enter `time`");
            continue;
        }
        else
        {

            printf("Recieved time from Server IP:%s and Port:%d is %s\n", client_ip,
                   client_port, buff);
        }
    }
}

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

    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1)
    {
        printf("socket creation failed...\n");
        exit(0);
    }
    else
        printf("Socket successfully created..\n");
    bzero(&servaddr, sizeof(servaddr));

    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(PORT);

    if (connect(sockfd, (SA *)&servaddr, sizeof(servaddr)) != 0)
    {
        printf("connection with the server failed...\n");
        exit(0);
    }
    else
        printf("connected to the server..\n");
    clifunc(sockfd, &servaddr);
    close(sockfd);
}
```

Output



```
student@lplab-Lenovo-Product: ~/180905350/cn/lab1
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ cc p4server.c -o p4server.obj
p4server.c: In function ‘main’:
p4server.c:111:5: warning: implicit declaration of function ‘close’ [-Wimplicit-
function-declaration]
     close(sockfd);
     ^
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ ./p4server.obj
Socket successfully created..
Socket successfully binded..
Server listening..
server acccept the client...
```



```
student@lplab-Lenovo-Product: ~/180905350/cn/lab1
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ cc p4client.c -o p4client.obj
p4client.c: In function ‘main’:
p4client.c:83:5: warning: implicit declaration of function ‘close’ [-Wimplicit-f
unction-declaration]
     close(sockfd);
     ^
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ ./p4client.obj
Socket successfully created..
connected to the server..
Enter the string : time
Recieved time from Server IP:0.0.0.0 and Port:5005 is Wed Dec  9 16:56:44 2020
```

Aditya Pradhan CSE D 180905350

Q3)
Implement concurrent Remote Math Server To perform arithmetic operations in the server and display the result at the client. The client accepts two integers and an operator from the user and sends it to the server. The server will performs the operation on integers and sends result back to the client which is displayed in the client.

Code

server
```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <unistd.h>
#define PORT 5000
int calc(int a, int b, char op)
{
    switch (op)
    {
    case '+':
        return a + b;
        break;
    case '-':
        return a - b;
        break;
    case '/':
        return a / b;
        break;
    case '*':
        return a * b;
        break;
    default:
        return 0;
        break;
    }
}
void servfunc(int sockfd, struct sockaddr_in server_address)
{
    struct sockaddr_in client_address;
    int clientfd, a, b, res, size = sizeof(client_address);
    char op;
    while (1)
    {
        clientfd = accept(sockfd, (struct sockaddr *)&client_address, &size);
        if (fork() == 0)
        {
            //in child process
            printf("Child process created with clientfd %d\n", clientfd);
            close(sockfd);
            read(clientfd, (int *)&a, sizeof(int));
            read(clientfd, (int *)&b, sizeof(int));
            read(clientfd, (char *)&op, sizeof(char));
            res = calc(a, b, op);
            write(clientfd, (int *)&res, sizeof(int));
            close(clientfd);
            printf("Child process terminated with clientfd %d\n", clientfd);
            exit(0);
        }
        else
        {
            //parent process
            close(clientfd);
        }
    }
    printf("server closing\n");
}
int main()
{
    int sockfd;
    struct sockaddr_in server_address;
    bzero(&server_address, sizeof(server_address));
    server_address.sin_family = AF_INET;
    server_address.sin_port = htons(PORT);
    server_address.sin_addr.s_addr = htonl(INADDR_ANY);
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    int res = bind(sockfd, (struct sockaddr *)&server_address, sizeof(server_address));
    if (res < 0)
    {
        printf("Server unable to bind\n");
        exit(0);
    }
    else
        printf("Server bound successfully\n");
    res = listen(sockfd, 2);
    if (res < 0)
    {
        printf("Server unable to listne\n");
        exit(0);
    }
    else
        printf("Server listening successfully\n");
    servfunc(sockfd, server_address);
    close(sockfd);
}
```

client

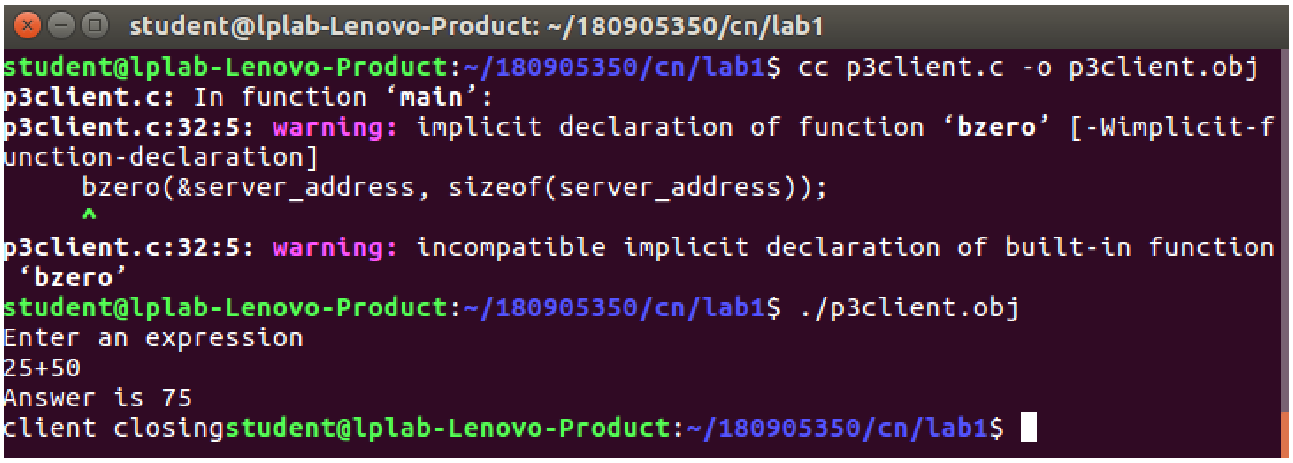
```
#include <sys/types.h>
#include <sys/socket.h>
```

Aditya Pradhan CSE D 180905350

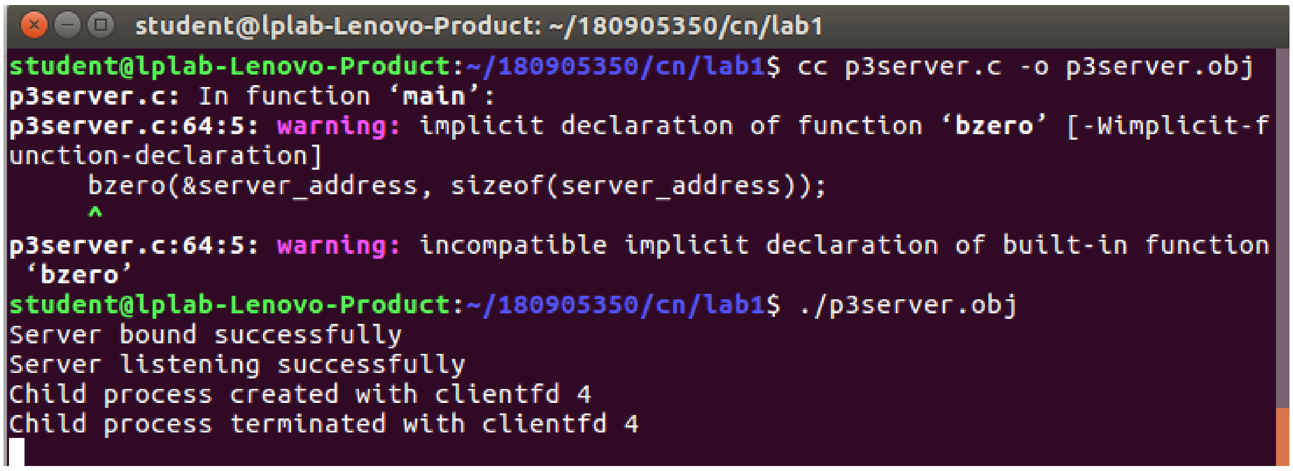
```
#include <stdio.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <stdlib.h>
#define PORT 5000

void clifunc(int sockfd)
{
    printf("Enter an expression\n");
    int a, b;
    char c;
    scanf("%d%c%d", &a, &c, &b);
    write(sockfd, (int *)&a, sizeof(int));
    write(sockfd, (int *)&b, sizeof(int));
    write(sockfd, (char *)&c, sizeof(char));
    int ans;
    read(sockfd, (int *)&ans, sizeof(int));
    printf("Answer is %d\n", ans);
    printf("client closing");
}
int main(int argc, char const *argv[])
{
    int sockfd;
    int len;
    struct sockaddr_in server_address;
    int result;
    char ch;
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    bzero(&server_address, sizeof(server_address));
    server_address.sin_family = AF_INET;
    server_address.sin_port = htons(PORT);
    server_address.sin_addr.s_addr = htonl(INADDR_ANY);
    len = sizeof(server_address);
    result = connect(sockfd, (struct sockaddr *)&server_address, len);
    if (result == -1)
    {
        printf("connection error\n");
        exit(0);
    }
    clifunc(sockfd);
    close(sockfd);
}
```

Aditya Pradhan CSE D 180905350

Output



```
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ cc p3client.c -o p3client.obj
p3client.c: In function ‘main’:
p3client.c:32:5: warning: implicit declaration of function ‘bzero’ [-Wimplicit-f
unction-declaration]
     bzero(&server_address, sizeof(server_address));
     ^
p3client.c:32:5: warning: incompatible implicit declaration of built-in function
 ‘bzero’
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ ./p3client.obj
Enter an expression
25+50
Answer is 75
client closingstudent@lplab-Lenovo-Product:~/180905350/cn/lab1$
```



```
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ cc p3server.c -o p3server.obj
p3server.c: In function ‘main’:
p3server.c:64:5: warning: implicit declaration of function ‘bzero’ [-Wimplicit-f
unction-declaration]
     bzero(&server_address, sizeof(server_address));
     ^
p3server.c:64:5: warning: incompatible implicit declaration of built-in function
 ‘bzero’
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ ./p3server.obj
Server bound successfully
Server listening successfully
Child process created with clientfd 4
Child process terminated with clientfd 4
```

Aditya Pradhan CSE D 180905350

Q4) Write a UDP client-server program where client sends rows of a matrix to the server combines them together as a two dimensional matrix and display the same.

Code

Server

```
// server program for udp connection
#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <netinet/in.h>
#define PORT 5000
#define MAXLINE 1000

// Server code
int main()
{
   char buffer[100];
   int servsockfd, len, n;
   struct sockaddr_in servaddr, cliaddr;
   bzero(&servaddr, sizeof(servaddr));

   // Create a UDP Socket
   servsockfd = socket(AF_INET, SOCK_DGRAM, 0);
   servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
   servaddr.sin_port = htons(PORT);
   servaddr.sin_family = AF_INET;

   // bind server address to socket descriptor
   bind(servsockfd, (struct sockaddr *)&servaddr, sizeof(servaddr));

   //receive the datagram
   while (1)
   {
      bzero(buffer, sizeof(buffer));
      len = sizeof(cliaddr);
      n = recvfrom(servsockfd, buffer, sizeof(buffer), 0, (struct sockaddr *)&cliaddr, &len);

      // buffer[n] = '\0';

      //Echoing back to the client
      if ((strncmp(buffer, "exit", 4)) == 0)
      {
         printf("Client Exit BYE...\n");

         break;
      }
      for (int i = 0; i < n; i++)
```

```
        {
            printf("%c", buffer[i]);
        }
        // sendto(servsockfd, buffer, n, 0, (struct sockaddr *)&cliaddr, sizeof(cliaddr));
        // getchar();
    }

    // close the descriptor
    close(servsockfd);
}
```

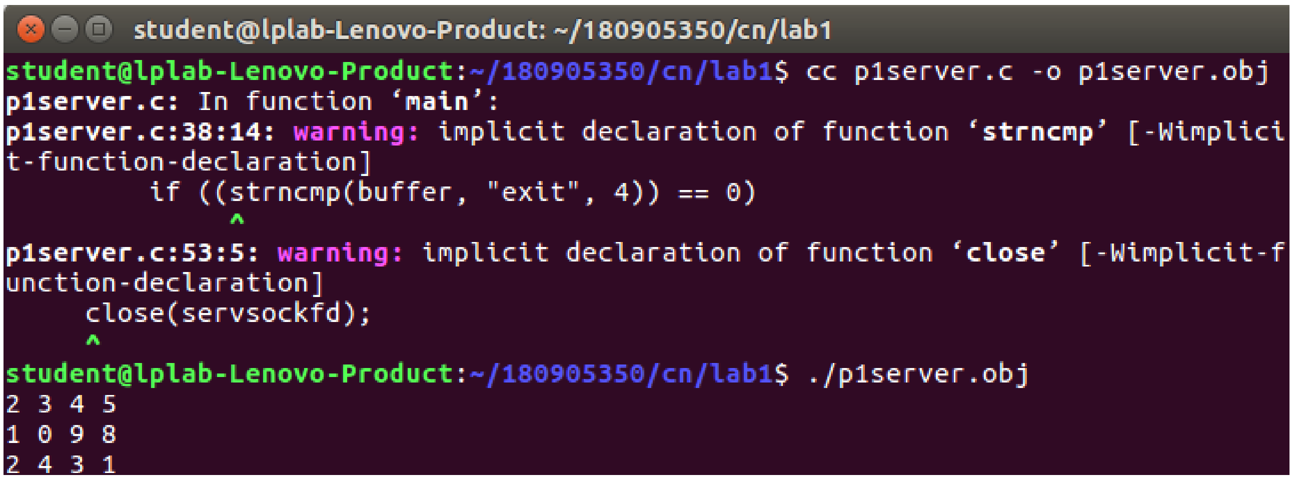
client

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

#define PORT 5000
#define MAXLINE 1000

// Driver code
int main()
{
    char buffer[100];
    //char *message = "";
    int sockfd, n, len;
    struct sockaddr_in servaddr, cliaddr;

    // clear servaddr
    bzero(&servaddr, sizeof(servaddr));
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(PORT);
    servaddr.sin_family = AF_INET;

    // create datagram socket
    sockfd = socket(AF_INET, SOCK_DGRAM, 0);
    char buff[100];
    for (;;)
    {
        bzero(buff, sizeof(buff));
        printf("Enter matrix row : ");
        n = 0;
        while ((buff[n++] = getchar()) != '\n')
            ;
        buff[n] = '\0';
```

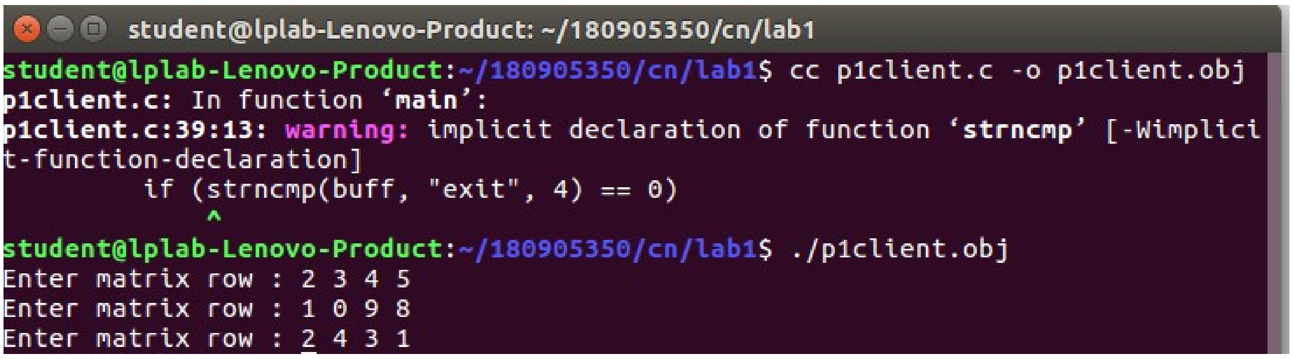
Aditya Pradhan CSE D 180905350

```
        sendto(sockfd, &buff, MAXLINE, 0, (struct sockaddr *)&servaddr, sizeof(servaddr));
        if (strncmp(buff, "exit", 4) == 0)
        {
            printf("Closing..");
            break;
        }
        bzero(buff, sizeof(buff));
        len = sizeof(cliaddr);
    }

    // close the descriptor
    close(sockfd);
}
```

Output



```
student@lplab-Lenovo-Product: ~/180905350/cn/lab1
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ cc p1server.c -o p1server.obj
p1server.c: In function ‘main’:
p1server.c:38:14: warning: implicit declaration of function ‘strncmp’ [-Wimplici
t-function-declaration]
         if ((strncmp(buffer, "exit", 4)) == 0)
              ^
p1server.c:53:5: warning: implicit declaration of function ‘close’ [-Wimplicit-f
unction-declaration]
     close(servsockfd);
     ^
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ ./p1server.obj
2 3 4 5
1 0 9 8
2 4 3 1
```



```
student@lplab-Lenovo-Product: ~/180905350/cn/lab1
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ cc p1client.c -o p1client.obj
p1client.c: In function ‘main’:
p1client.c:39:13: warning: implicit declaration of function ‘strncmp’ [-Wimplici
t-function-declaration]
         if (strncmp(buff, "exit", 4) == 0)
             ^
student@lplab-Lenovo-Product:~/180905350/cn/lab1$ ./p1client.obj
Enter matrix row : 2 3 4 5
Enter matrix row : 1 0 9 8
Enter matrix row : 2 4 3 1
```