

Computer Networks
RA1911030010014
Experiment - 2
Study of Basic Functions of Socket Programming

Aim: To create a simple socket program, client-server communication.

Server Code:

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

int main(int argc, char *argv[])
{
    int listenfd = 0, connfd = 0;
    struct sockaddr_in serv_addr;

    char sendBuff[1025];
    time_t ticks;

    listenfd = socket(AF_INET, SOCK_STREAM, 0);
    memset(&serv_addr, '0', sizeof(serv_addr));
    memset(sendBuff, '0', sizeof(sendBuff));

    serv_addr.sin_family = AF_INET;
    serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
    serv_addr.sin_port = htons(5000);

    bind(listenfd, (struct sockaddr *)&serv_addr, sizeof(serv_addr));

    listen(listenfd, 10);

    while (1)
    {
        connfd = accept(listenfd, (struct sockaddr *)NULL, NULL);

        ticks = time(NULL);
        snprintf(sendBuff, sizeof(sendBuff), "%.24s\r\n", ctime(&ticks));
        write(connfd, sendBuff, strlen(sendBuff));

        close(connfd);
        sleep(1);
    }
}
```

```
}
```

Client Code:

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

int main(int argc, char *argv[])
{
    int sockfd = 0, n = 0;
    char recvBuff[1024];
    struct sockaddr_in serv_addr;

    if (argc != 2)
    {
        printf("\n Usage: %s <ip of server> \n", argv[0]);
        return 1;
    }

    memset(recvBuff, '0', sizeof(recvBuff));
    if ((sockfd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        printf("\n Error : Could not create socket \n");
        return 1;
    }

    memset(&serv_addr, '0', sizeof(serv_addr));

    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(5000);

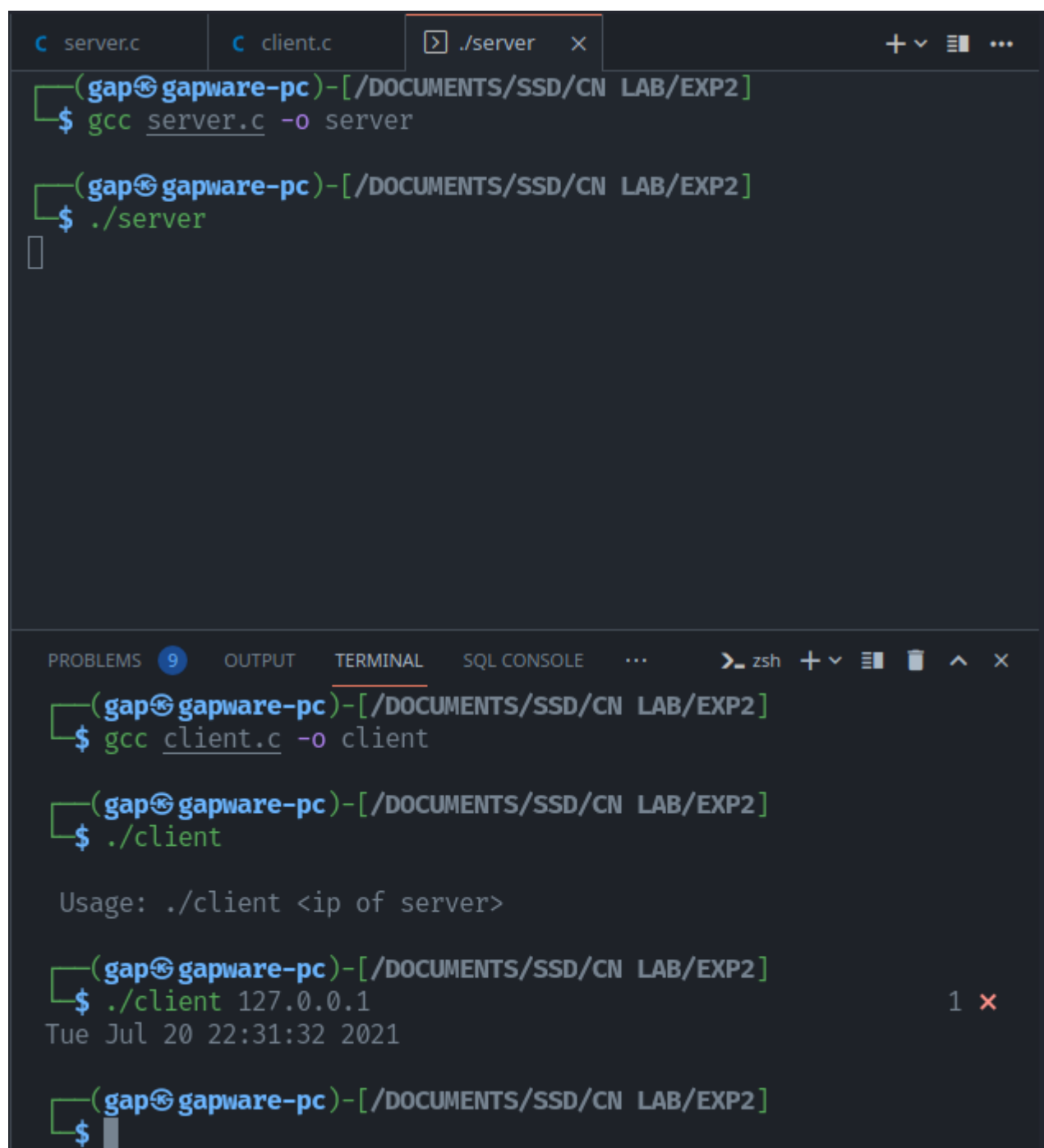
    if (inet_pton(AF_INET, argv[1], &serv_addr.sin_addr) <= 0)
    {
        printf("\n inet_pton error occured\n");
        return 1;
    }

    if (connect(sockfd, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0)
    {
        printf("\n Error : Connect Failed \n");
        return 1;
    }
}
```

```
while ((n = read(sockfd, recvBuff, sizeof(recvBuff) - 1)) > 0)
{
    recvBuff[n] = 0;
    if (fputs(recvBuff, stdout) == EOF)
    {
        printf("\n Error : Fputs error\n");
    }
}

if (n < 0)
{
    printf("\n Read error \n");
}

return 0;
}
```

Output:

```
(gap@gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP2]
$ gcc server.c -o server

(gap@gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP2]
$ ./server

(gap@gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP2]
$ gcc client.c -o client

(gap@gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP2]
$ ./client

Usage: ./client <ip of server>

(gap@gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP2]
$ ./client 127.0.0.1
Tue Jul 20 22:31:32 2021
```

Result:

The required code for client-server communication was written in Visual Studio Code editor and successfully compiled using 2 terminals.