

Computer Networks Lab 2

1. Write a TCP concurrent client server program where server accepts integer array from client and sorts it and returns it to the client along with process id.

Server Side program file called “q1server.c”:

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

int cmpfunc(const void *a, const void *b){
    return (*(int *)a - *(int *)b);
}

int main(){
    int sd, nd, len, n;
    struct sockaddr_in seraddress, cliaddr;
    int arr[20];
    int arr_size = 0;

    sd = socket(AF_INET, SOCK_STREAM, 0);

    seraddress.sin_family = AF_INET;

    seraddress.sin_addr.s_addr = INADDR_ANY;

    seraddress.sin_port = htons(10200);

    bind(sd, (struct sockaddr *)&seraddress, sizeof(seraddress));
```

```

listen(sd, 5);

len = sizeof(cliaddr);

while (1){
    nd = accept(sd, (struct sockaddr *)&cliaddr, &len);
    printf("Connected to client");

    if (fork() == 0){
        close(sd);
        int pid = getpid();
        n = read(nd, &arr_size, sizeof(int));
        n = read(nd, arr, arr_size * sizeof(int));

        //Sort
        qsort(arr, arr_size, sizeof(int), cmpfunc);

        n = write(nd, &pid, sizeof(int));
        n = write(nd, arr, arr_size * sizeof(int));

        getchar();

        close(nd);
    }
}
}

```

Client Side program file called “q1client.c”:

```

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

int main(){

```

```

int sd, len, n;
struct sockaddr_in seraddress, cliaddr;
int arr[20];
int arr_size, pid;

sd = socket(AF_INET, SOCK_STREAM, 0);
seraddress.sin_family = AF_INET;
seraddress.sin_addr.s_addr = INADDR_ANY;
seraddress.sin_port = htons(10200);

len = sizeof(seraddress);
connect(sd, (struct sockaddr *)&seraddress, len);

printf("Enter number of elements: \n");
scanf("%d", &arr_size);
printf("Enter elements: \n");

for (int i = 0; i < arr_size; i++){
    scanf("%d", &arr[i]);
}

n = write(sd, &arr_size, sizeof(int));
n = write(sd, arr, arr_size * sizeof(int));
n = read(sd, &pid, sizeof(int));
n = read(sd, arr, arr_size * sizeof(int));

printf("\nSorted array: ");
for (int i = 0; i < arr_size; i++){
    printf("%d ", arr[i]);
}

printf("\nProcess ID: %d\n", pid);

getchar();
}

```

We first compile and execute the server side program in a terminal, after which we compile and run the client side program. The desired output is as shown below:

Output:

Server side terminal:

```
Student@project-lab: ~/Desktop/AyushGoyalCNLab/Lab_2
File Edit View Search Terminal Help
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ gcc q1server.c -o q1server
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ ./q1server
Connected to client
```

Client side terminal:

```
Student@project-lab: ~/Desktop/AyushGoyalCNLab/Lab_2
File Edit View Search Terminal Help
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ gcc q1client.c -o q1client
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ ./q1client
Enter number of elements:
5
Enter elements:
23 12 45 21 55

Sorted array: 12 21 23 45 55
Process ID: 5547
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$
```

2. 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 then receives integers and operator. The server will performs the operation on integers and sends result back to the client which is displayed on the client screen. Then both the processes terminate.

Server Side program file called “q2server.c”:

```
#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 operator)
{
    switch(operator)
    {
        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
        close(clientfd); // parent process
}
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 Side program file called “q2client.c”:

```
#include <sys/types.h>
#include <sys/socket.h>
#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("I am a client and my name is Ayush Goyal.\n");
    int a, b;
    char c;
    printf("Enter the expression as you would on a Calculator: \n");
    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 res;
    read(sockfd, (int *)&res, sizeof(int));
    printf("%d %c %d = %d\n", a, c, b, res);
    printf("Client Closing!\n");
}

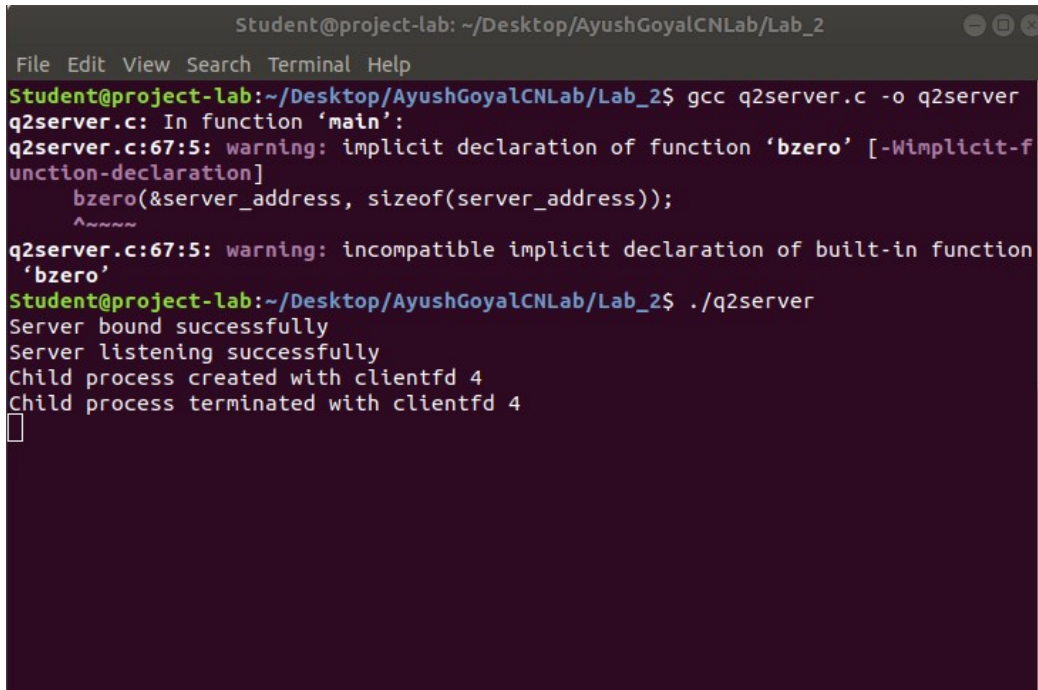
int main(int argc, char const *argv[])
{
    int sockfd;
    int len;
    struct sockaddr_in server_address;
    int result;
    char ch;
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    bzero(&server_address, sizeof(server_address));
    server_address.sin_family = AF_INET;
    server_address.sin_port = 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 Occured!\n");
        exit(0);
    }
}
```

```
    clifunc(sockfd);  
    close(sockfd);  
}
```

We first compile and execute the server side program in a terminal, after which we compile and run the client side program. The desired output is as shown below:

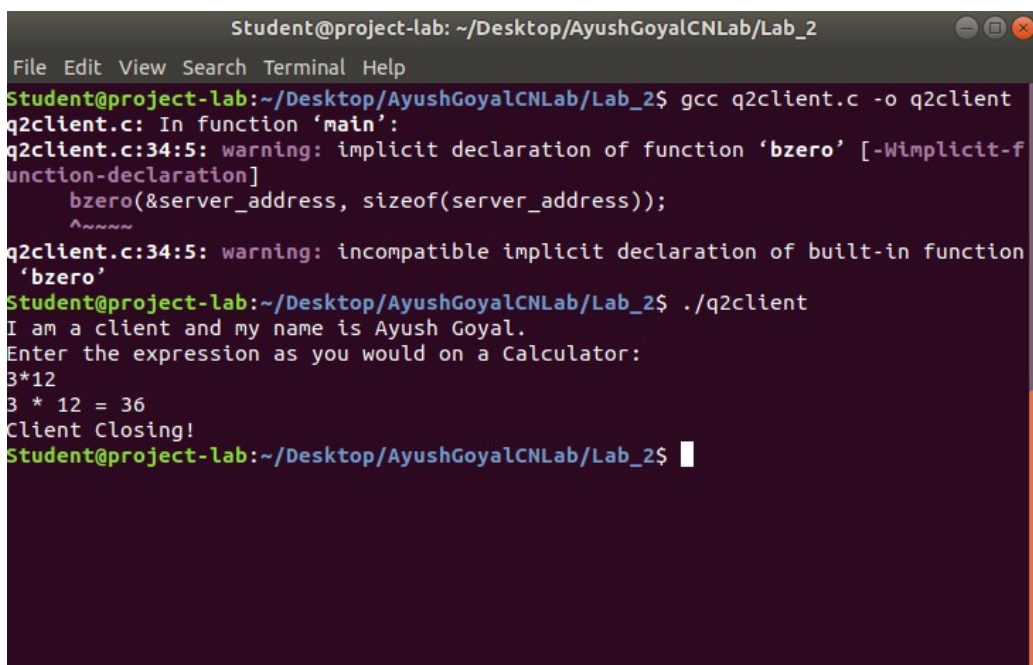
Output:

Server side terminal:



```
Student@project-lab: ~/Desktop/AyushGoyalCNLab/Lab_2  
File Edit View Search Terminal Help  
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ gcc q2server.c -o q2server  
q2server.c: In function 'main':  
q2server.c:67:5: warning: implicit declaration of function 'bzero' [-Wimplicit-f  
unction-declaration]  
    bzero(&server_address, sizeof(server_address));  
    ^~~~~~  
q2server.c:67:5: warning: incompatible implicit declaration of built-in function  
'bzero'  
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ ./q2server  
Server bound successfully  
Server listening successfully  
Child process created with clientfd 4  
Child process terminated with clientfd 4  
█
```

Client side terminal:



```
Student@project-lab: ~/Desktop/AyushGoyalCNLab/Lab_2  
File Edit View Search Terminal Help  
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ gcc q2client.c -o q2client  
q2client.c: In function 'main':  
q2client.c:34:5: warning: implicit declaration of function 'bzero' [-Wimplicit-f  
unction-declaration]  
    bzero(&server_address, sizeof(server_address));  
    ^~~~~~  
q2client.c:34:5: warning: incompatible implicit declaration of built-in function  
'bzero'  
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ ./q2client  
I am a client and my name is Ayush Goyal.  
Enter the expression as you would on a Calculator:  
3*12  
3 * 12 = 36  
Client Closing!  
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ █
```


3. Implement simple TCP daytime server using fork.

Server Side program file called “q3server.c”:

```
#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 5000
#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 Side program file called “q3client.c”:

```
#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 5000
#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 (;;)
    {
        printf("I am a client. My name is Ayush Goyal.\n");
        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!\n");
            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\n");
            continue;
        }
        else
            printf("Recieved time from Server IP:%s and Port:%d is %s\n", client_ip, client_port, buff);
    }
}
```

```

    }
}

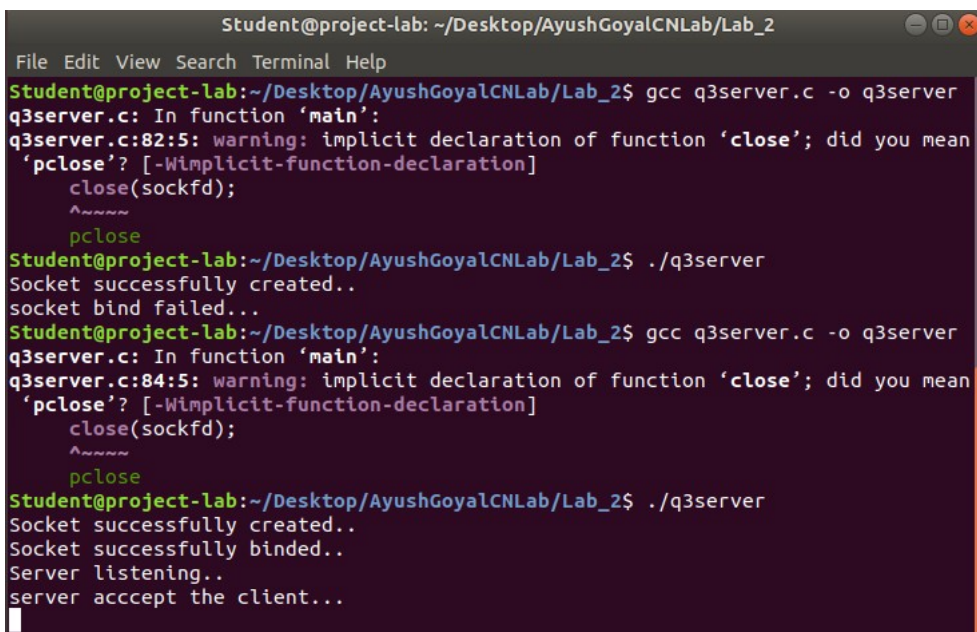
int main(){
    int sockfd, connfd;
    struct sockaddr_in servaddr, cli;
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1){
        printf("socket creation failed...\n");
        exit(0);
    }
    else
        printf("Socket successfully created..\n");
    bzero(&servaddr, sizeof(servaddr));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

```

We first compile and execute the server side program in a terminal, after which we compile and run the client side program. The desired output is as shown below:

Output:

Server side terminal:



```

Student@project-lab: ~/Desktop/AyushGoyalCNLab/Lab_2
File Edit View Search Terminal Help
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ gcc q3server.c -o q3server
q3server.c: In function 'main':
q3server.c:82:5: warning: implicit declaration of function 'close'; did you mean
'pclose'? [-Wimplicit-function-declaration]
    close(sockfd);
    ^
pclose
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ ./q3server
Socket successfully created..
socket bind failed...
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ gcc q3server.c -o q3server
q3server.c: In function 'main':
q3server.c:84:5: warning: implicit declaration of function 'close'; did you mean
'pclose'? [-Wimplicit-function-declaration]
    close(sockfd);
    ^
pclose
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ ./q3server
Socket successfully created..
Socket successfully binded..
Server listening..
server accept the client...

```

Client side terminal:

```
Student@project-lab: ~/Desktop/AyushGoyalCNLab/Lab_2
File Edit View Search Terminal Help
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ gcc q3client.c -o q3client
q3client.c: In function 'main':
q3client.c:70:5: warning: implicit declaration of function 'close'; did you mean
'pclose'? [-Wimplicit-function-declaration]
    close(sockfd);
    ^~~~~
    pclose
Student@project-lab:~/Desktop/AyushGoyalCNLab/Lab_2$ ./q3client
Socket successfully created..
connected to the server..
I am a client. My name is Ayush Goyal.
Enter the String : time
Recieved time from Server IP:0.0.0.0 and Port:5000 is Thu Oct 21 18:06:23 2021

I am a client. My name is Ayush Goyal.
Enter the String : 
```

THE END