Server Code

The following describes the steps of a server accepting connections from a client:

1. Create a TCP socket:

```
listensockfd = socket (AF_INET, SOCK_STREAM, 0);
```

2. Get information about host running server:

```
gethostname(hostname, 32);
hostptr = gethostbyname(hostname);
```

3. Fill in destination address structure:

```
memset((void *) &servaddr, 0, (size_t)sizeof(servaddr));
servaddr.sin_family = (short)(AF_INET);
memcpy((void *)& servaddr.sin_addr,
       (void *) hostptr->h_addr, hostptr->h_length);
servaddr.sin_port = htons((u_short)8000);
```

4. Bind socket locally:

```
bind(listensockfd, (struct sockaddr *) &servaddr,
         (socklen_t)sizeof(servaddr));
```

5. Listen on socket:

```
listen (listensockfd, MAX_NUM_LISTENER_ALLOWED);
```

6. Accept incoming connections:

```
for (;;) {
     connsockfd = accept (listensockfd, NULL, NULL);
     // receive and send
    …
    close (connsockfd);
}
```

Client Code

In a nutshell, client code must perform the following steps to communicate with a server:

1. Create a TCP socket:
```
sockfd= socket (AF_INET, SOCK_STREAM, 0);
```
2. Get information about destination host:
```
hostptr = gethostbyname(name);
```
3. Fill in destination address structure:
```
memset((void *) &dest, 0, (size_t)sizeof(dest)); // or bzero()
dest.sin_family = (short)(AF_INET);
memcpy((void *)&dest.sin_addr,
       (void *)hostptr->h_addr, hostptr->h_length);
dest.sin_port = htons((u_short)8000);
```
4. Connect to server:
```
connect(sockfd, (struct sockaddr *)&dest, sizeof(dest));
```
5. Send data to destination:
```
write(sockfd, buf, strlen(buf)+1);
```
6. Read data from destination:
```
bzero(buffer, 256);     // instead of memset
read(sockfd, buffer, 255);
```
7. Close socket:
```
close (sockfd);
```