Server Code

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

1. Create a UDP socket:

```
listensockfd = socket (AF_INET, SOCK_DGRAM, 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)60000);
```

4. Bind socket locally:

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

5. Process incoming messages:

```
char buffer[BUFFER_SIZE];
int bytesRcvd;

struct sockaddr_in clientAddr;
socklen_t clientAddrLen = sizeof(clientAddr);

for (;;) {
     bytesRcvd = recvfrom(listensockfd, buffer, BUFFER_SIZE, 0,
             (struct sockaddr *)&clientAddr, clientAddrLen);
     // process request

     sendto(listensockfd, buffer, BUFFER_SIZE, 0,
         (struct sockaddr *)&clientAddr, clientAddrLen);
}
close (listensockfd);
```

Client Code

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

1. Create a UDP socket:

```
sockfd= socket (AF_INET, SOCK_DGRAM, 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)60000);
```

4. Send message to server:

```
sendto(sockfd, buffer, BUFFER_SIZE,
       (struct sockaddr *)&dest, (socklent_t)sizeof(dest));
```

5. Receive message from server:

```
bzero(buffer, BUFFER_SIZE);     // instead of memset
recvfrom(sockfd, buffer, BUFFER_SIZE, 0, NULL, NULL);
```

6. Close socket:

```
close (sockfd);
```