```
1  //*****************************************************************************
2  // ip_client.cc TCP/IP socket client
3  // Author:      M. Thaler
4  // Date:        7.7.99
5  // Changes:     tha, 5/2008
6  //              using getaddrinfo -> IPV4 and IPV6 compatible
7  //*****************************************************************************
8
9  //*****************************************************************************
10 // system includes
11 //*****************************************************************************
12
13 #include <stdio.h>
14 #include <stdlib.h>
15 #include <string.h>
16 #include <netinet/in.h>
17 #include <sys/types.h>
18 #include <sys/socket.h>
19 #include <unistd.h>
20 #include <netdb.h>
21
22 #include <iostream>
23 using namespace std;
24
25 //*****************************************************************************
26 // local includes
27 //*****************************************************************************
28
29 #include "lsocks.h"       // PORT_NUMBER, BUF_SIZE
30
31 //*****************************************************************************
32 // Function: main(), parameter: hostname or IP address in dot format
33 //*****************************************************************************
34
35 int main(int argc, char *argv[]) {
36
37   int    sfd, cfd, addrlen, j, sysRet;
38   char   stringPort[8], buf[BUF_SIZE];
39     struct addrinfo hints, *aiList, *aiPtr = NULL;
40
41   if (argc < 2)  {
42     cout << "Need hostname or IP address" << endl;
43     exit(-1);
44   }
45
46   cout << endl << "IP client" << endl;
47
48     sprintf(stringPort, "%d", PORT_NUMBER);
49
50     memset(&hints, '\0', sizeof(hints));
51     hints.ai_flags    = AI_CANONNAME;
52     hints.ai_family   = AF_UNSPEC;
53     hints.ai_socktype = SOCK_STREAM;
54
55     sysRet = getaddrinfo(argv[1], stringPort, &hints, &aiList);
56     if (sysRet != 0) {
57         printf("error getting network address %s\n", gai_strerror(sysRet));
58         return(-1);
59     }
60
61     aiPtr = aiList;                              // search through list
62     while (aiPtr != 0) {
```

```
63          sfd = socket(aiPtr->ai_family, aiPtr->ai_socktype, aiPtr->ai_protocol);
64          if (sfd >= 0) {
65              cout << "Socket created" << endl;
66              cout << "Connected to port #" << stringPort << endl;
67              cout << "Connecting to server ..." << endl;
68              sysRet = connect(sfd, aiPtr->ai_addr, aiPtr->ai_addrlen);
69              if (sysRet == 0)
70                  break;                          // connect successful
71              else
72                  close(sfd);
73          }
74          aiPtr = aiPtr->ai_next;
75      }
76      freeaddrinfo(aiList);
77      if (aiPtr == NULL) {
78          printf("could not connect to %s\n", argv[1]);
79          exit(-1);
80      }
81
82      cout << "... connection established" << endl;
83
84    cout << "? " ;
85    while (fgets(buf, BUF_SIZE, stdin) > 0) {
86      write(sfd, buf, strlen(buf));
87      if (buf[0] == '!') break;       // terminate on '!'
88      if (buf[0] == '@') break;       // terminate on '@'
89      cout << "? " ;
90    }
91    close(sfd);
92    cout << "Connection terminted" << endl;
93  }
```