

```

1 // Homework Number: hw10
2 // Name: Shu Hwai Teoh
3 // ECN Login: teoh0
4 // Due Date: Thursday 4/09/2020 at 4:29PM
5
6 /*
7 / This is a server socket program that echos recieved messages
8 / from the client.c program. where 8000 is the port you want your server to monitor.
9 */
10
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include <errno.h>
14 #include <string.h>
15 #include <sys/types.h>
16 #include <netinet/in.h>
17 #include <sys/socket.h>
18 #include <sys/wait.h>
19 #include <arpa/inet.h>
20 #include <unistd.h>
21
22 #define MAX_PENDING 10 /* maximun # of pending for connection */
23 #define MAX_DATA_SIZE 5
24
25 int DataPrint(char *recvBuff, int numBytes);
26 char* clientComm(int clntSockfd, int * senderBuffSize_addr, int * optlen_addr);
27
28 int main(int argc, char *argv[])
29 {
30     if (argc < 2) {
31         fprintf(stderr, "ERROR, no port provided\n");
32         exit(1);
33     }
34     int PORT = atoi(argv[1]);
35
36
37     int senderBuffSize;
38     int servSockfd, clntSockfd;
39     struct sockaddr_in sevrAddr;
40     struct sockaddr_in clntAddr;
41     int clntLen;
42     socklen_t optlen = sizeof senderBuffSize;
43
44     /* make socket */
45     if ((servSockfd = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
46         perror("sock failed");
47         exit(1);
48     }
49
50     /* set IP address and port */
51     sevrAddr.sin_family = AF_INET;
52     sevrAddr.sin_port = htons(PORT);
53     sevrAddr.sin_addr.s_addr = INADDR_ANY;
54     bzero(&(sevrAddr.sin_zero), 8);
55
56     if (bind(servSockfd, (struct sockaddr *)&sevrAddr,
57             sizeof(struct sockaddr)) == -1) {
58         perror("bind failed");
59         exit(1);
60     }
61
62     if (listen(servSockfd, MAX_PENDING) == -1) {
63         perror("listen failed");
64         exit(1);
65     }
66
67     while(1) {
68         clntLen = sizeof(struct sockaddr_in);
69         if ((clntSockfd = accept(servSockfd, (struct sockaddr *)&clntAddr,
70                                &clntLen)) == -1) {
71             perror("accept failed");
72             exit(1);
73         }

```

```

73
74     printf("Connected from %s\n", inet_ntoa(clntAddr.sin_addr));
75
76     if (send(clntSockfd, "Connected!!!\n", strlen("Connected!!!\n"), 0) == -1) {
77         perror("send failed");
78         close(clntSockfd);
79         exit(1);
80     }
81
82     /* repeat for one client service */
83     while(1) {
84         free(clientComm(clntSockfd, &senderBuffSize, &optlen));
85     }
86
87     close(clntSockfd);
88     exit(1);
89 }
90 }
91
92 char * clientComm(int clntSockfd, int * senderBuffSize_addr, int * optlen_addr){
93     char *recvBuff; /* recv data buffer */
94     int numBytes = 0;
95     char str[MAX_DATA_SIZE] = "less";
96     /* recv data from the client */
97     getsockopt(clntSockfd, SOL_SOCKET, SO_SNDBUF, senderBuffSize_addr, optlen_addr);
98     /* check sender buffer size */
99     recvBuff = malloc((*senderBuffSize_addr) * sizeof (char));
100
101     if ((numBytes = recv(clntSockfd, recvBuff, *senderBuffSize_addr, 0)) == -1) {
102         perror("recv failed");
103         exit(1);
104     }
105
106     // if the number of received byte is larger than MAX_DATA_SIZE
107     // avoid using strcpy for variable str.
108     while(numBytes >= MAX_DATA_SIZE){
109         fprintf(stderr, "Too many characters for input will cause buffer overflow.\n");
110
111         //send "less" to let client input less characters
112         if (send(clntSockfd, str, strlen(str), 0) == -1) {
113             perror("send failed");
114             close(clntSockfd);
115             exit(1);
116         }
117
118         if ((numBytes = recv(clntSockfd, recvBuff, *senderBuffSize_addr, 0)) == -1) {
119             perror("recv failed");
120             exit(1);
121         }
122     }
123
124     recvBuff[numBytes] = '\0';
125     if(DataPrint(recvBuff, numBytes)){
126         fprintf(stderr, "ERROR, no way to print out\n");
127         exit(1);
128     }
129
130     strcpy(str, recvBuff);
131
132     /* send data to the client */
133     if (send(clntSockfd, str, strlen(str), 0) == -1) {
134         perror("send failed");
135         close(clntSockfd);
136         exit(1);
137     }
138
139     return recvBuff;
140 }
141
142 void secretFunction(){
143     printf("You weren't supposed to get here!\n");
144     exit(1);

```

```
145     }
146
147     int DataPrint(char *recvBuff, int numBytes) {
148         printf("RECEIVED: %s", recvBuff);
149         printf("RECEIVED BYTES: %d\n\n", numBytes);
150         return(0);
151     }
152
```