

電腦網路

LAB:LAB3

報告

班級:資工 2A

學號:01057020

姓名:林佑丞

1.

code:

Server 端:

```
1  #include <winsock2.h>
2  #include <stdio.h>
3  #include <winsock2.h>
4  #include <Ws2tcpip.h>
5  #include <cstdlib>
6  #include <ctime>
7  #include <thread>
8  #include <vector>
9  #include <string>
10 using namespace std;
11
12 int x;
13
14 const wchar_t* GetWC(const char* c) { //將字串從const char* 轉為 const wchar_t*
15     const size_t cSize = strlen(c) + 1;
16     wchar_t* wc = new wchar_t[cSize];
17     size_t convertedChar = 0;
18     mbstowcs_s(&convertedChar, wc, cSize, c, _TRUNCATE);
19     return wc;
20 }
21
22 int main() {
23     WSADATA wsaData;
24
25     if (WSAStartup(MAKEWORD(1, 1), &wsaData) != 0) {
26         fprintf(stderr, "WSAStartup failed.\n");
27         exit(1);
28     }
29
30     SOCKET sListen;
31     SOCKET sConnect;
32
33     sockaddr_in addr;
34     InetPton(AF_INET, GetWC("127.0.0.1"), &addr.sin_addr.s_addr); // 設定 IP
35     addr.sin_family = AF_INET;
36     addr.sin_port = htons(1234); // 設定 port, htons()跟網路位元組順序有關
37
38     sListen = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
39     //AF_INET:IPv4    AF_INET6:IPv6
40     //SOCK_STREAM:TCP  SOCK_DGRAM:UDP
41     //IPPROTO_TCP:TCP  IPPROTO_UDP:UDP
```

```
42 bind(sListen, (SOCKADDR*)&addr, sizeof(SOCKADDR)); //server端綁定位置
43 listen(sListen, 20);
44
45 srand(time(NULL));
46
47 sockaddr_in clientAddr; // client 端位址資訊
48 int clientAddrLen = sizeof(clientAddr);
49 sConnect = accept(sListen, (SOCKADDR*)&clientAddr, &clientAddrLen);
50 if (sConnect != INVALID_SOCKET)
51 {
52     // 有 client 端成功連線過來
53     int x = rand() % 9 + 1;
54     char clientIP[20];
55     inet_ntop(AF_INET, (void*)&clientAddr, clientIP, 20);
56     printf("server: got connection from %s\n", clientIP);
57     printf("answer: %d\n", x);
58     while (true) {
59         char recvbuf[2];
60         recv(sConnect, recvbuf, sizeof(recvbuf), 0);
61         int guess = atoi(recvbuf);
62         printf("Data received: %d\n", guess);
63         if (guess == x) {
64             const char* s = "Bingo!";
65             send(sConnect, s, (int)strlen(s), 0);
66         }
67         else if (guess == 0)
68             break;
69         else {
70             const char* s = "Fail";
71             send(sConnect, s, (int)strlen(s), 0);
72         }
73         memset(recvbuf, 0, sizeof(recvbuf));
74     }
75 }
76
77 closesocket(sConnect);
78 closesocket(sListen);
79
80 WSACleanup();
81
82 return 0;
83
84 }
```

Client 端:

```
1  #include <stdio.h>
2  #include <winsock2.h>
3  #include <ws2tcpip.h>
4
5  const wchar_t* GetWC(const char* c) {
6      const size_t cSize = strlen(c) + 1;
7      wchar_t* wc = new wchar_t[cSize];
8      size_t convertedChar = 0;
9      mbstowcs_s(&convertedChar, wc, cSize, c, _TRUNCATE);
10     return wc;
11 }
12
13 int main() {
14     WSADATA wsaData;
15
16     if (WSAStartup(MAKEWORD(1, 1), &wsaData) != 0) {
17         fprintf(stderr, "WSAStartup failed.\n");
18         exit(1);
19     }
20
21     SOCKET sock;
22
23     sockaddr_in addr;
24     InetPton(AF_INET, GetWC("127.0.0.1"), &addr.sin_addr.s_addr); // 設定 IP
25     addr.sin_family = AF_INET;
26     addr.sin_port = htons(1234); // 設定 port, htons() 跟網路位元組順序有關
27
28     sock = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
29
30     connect(sock, (SOCKADDR*)&addr, sizeof(SOCKADDR));
31
32     char szBuffer[MAXBYTE] = { 0 };
33
34     int guess;
35     int iResult;
36     while (true) {
37         printf("Your guess(1-9): ");
38         scanf_s("%d", &guess);
39
40         char sendbuf[2];
41         sprintf_s(sendbuf, sizeof(sendbuf), "%d", guess);
42
43         iResult = send(sock, sendbuf, (int)strlen(sendbuf), 0);
44         if (iResult == SOCKET_ERROR) {
45             printf("send failed with error: %d\n", WSAGetLastError());
46             closesocket(sock);
47             WSACleanup();
48         }
49         else if (iResult > 0) {
50             recv(sock, szBuffer, MAXBYTE, NULL);
51             printf("Data received: %s\n", szBuffer);
52             if (strcmp(szBuffer, "Bingo!") == 0)
53                 break;
54         }
55         else if (iResult == 0) {
56             printf("Connection closed\n");
57             break;
58         }
59         else {
60             printf("recv failed with error: %d\n", WSAGetLastError());
61             closesocket(sock);
62             WSACleanup();
63             return 1;
64         }
65     }
66
67     closesocket(sock);
68
69     WSACleanup();
70
71     return 0;
72 }
```

執行結果:

Microsoft Visual Studio 範本1 x + -

server: got connection from 2.0.234.180
answer: 9
Data received: 1
Data received: 2
Data received: 3
Data received: 4
Data received: 5
Data received: 6
Data received: 7
Data received: 8
Data received: 9
Data received: 8

C:\Users\ap006\source\repos\server\server\Debug\server.exe (應用程式 480448) 已結束，出現代碼 0。
若要在此視窗停止時自動關閉主控台，請啟用【工具】->【選項】->【解碼停止時，自動關閉主控台】，
並任意離開此視窗。】

Microsoft Visual Studio 範本1 x + -

Your guess(1-9): 1
Data received: Fail
Your guess(1-9):
2
Data received: Fail
Your guess(1-9): 3
Data received: Fail
Your guess(1-9): 4
Data received: Fail
Your guess(1-9): 5
Data received: Fail
Your guess(1-9): 6
Data received: Fail
Your guess(1-9): 7
Data received: Fail
Your guess(1-9): 8
Data received: Fail
Your guess(1-9): 9
Data received: Bingo!

C:\Users\ap006\source\repos\client\client\Debug\client.exe (應用程式 39396) 已結束，出現代碼 0。
若要在此視窗停止時自動關閉主控台，請啟用【工具】->【選項】->【解碼停止時，自動關閉主控台】，
並任意離開此視窗。】

2.

code:

Server 端

```
1  #include <winsock2.h>
2  #include <stdio.h>
3  #include <winsock2.h>
4  #include <Ws2tcpip.h>
5  #include <cstdlib>
6  #include <ctime>
7  #include <thread>
8  #include <vector>
9  #include <string>
10 using namespace std;
11
12 int x;
13
14 const wchar_t* GetWC(const char* c) { //將字串從const char* 轉為 const wchar_t*
15     const size_t cSize = strlen(c) + 1;
16     wchar_t* wc = new wchar_t[cSize];
17     size_t convertedChar = 0;
18     mbstowcs_s(&convertedChar, wc, cSize, c, _TRUNCATE);
19     return wc;
20 }
21
22 void check(SOCKET Connect, string ip) {
23     while (true) {
24         char recvbuf[2];
25         recv(Connect, recvbuf, sizeof(recvbuf), 0);
26         int guess = atoi(recvbuf);
27         printf("%s: %d\n", ip.c_str(), guess);
28         if (guess == x) {
29             const char* s = "Bingo!";
30             send(Connect, s, (int)strlen(s), 0);
31         }
32         else if (guess == 0)
33             break;
34         else {
35             const char* s = "Fail";
36             send(Connect, s, (int)strlen(s), 0);
37         }
38         memset(recvbuf, 0, sizeof(recvbuf));
39     }
40 }
41 //*
```

```

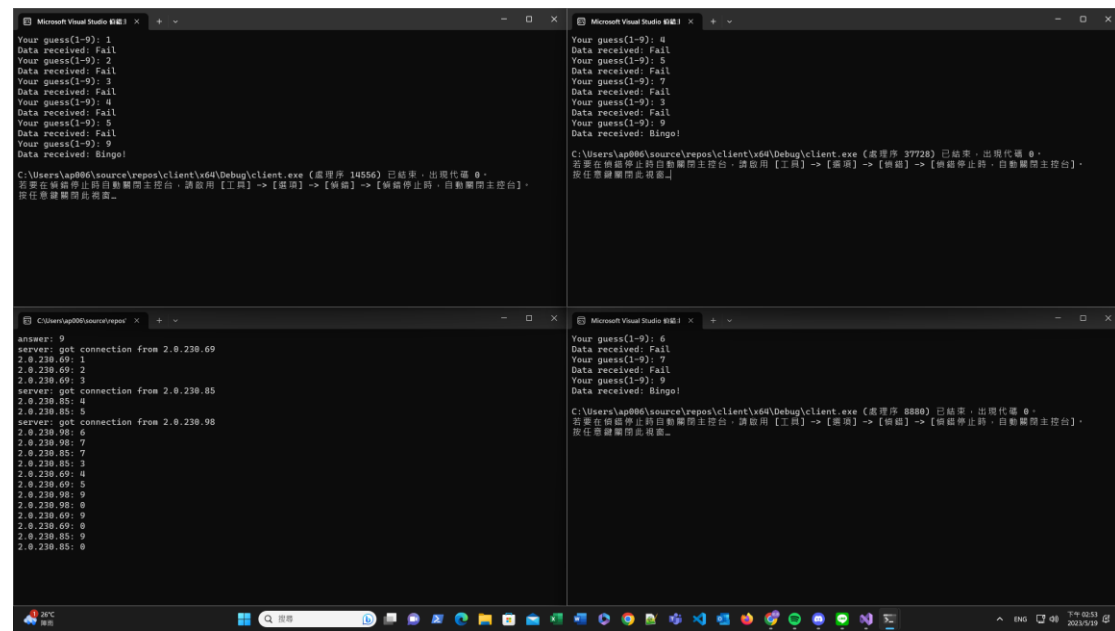
42 int main() {
43     WSADATA wsaData;
44     if (WSAStartup(MAKEWORD(1, 1), &wsaData) != 0) {
45         fprintf(stderr, "WSAStartup failed.\n");
46         exit(1);
47     }
48
49     SOCKET sListen;
50     SOCKET sConnect;
51     sockaddr_in addr;
52     InetPton(AF_INET, GetWC("127.0.0.1"), &addr.sin_addr.s_addr); // 設定 IP
53     addr.sin_family = AF_INET;
54     addr.sin_port = htons(1234); // 設定 port, htons() 跟網路位元組順序有關
55
56     sListen = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
57     // AF_INET: IPv4    AF_INET6: IPv6
58     // SOCK_STREAM: TCP  SOCK_DGRAM: UDP
59     // IPPROTO_TCP: TCP  IPPROTO_UDP: UDP
60     bind(sListen, (SOCKADDR*)&addr, sizeof(SOCKADDR)); // server 端綁定位置
61     listen(sListen, 20);
62
63     srand(time(NULL));
64
65     vector<thread> clients;
66     vector<string> ip;
67     x = rand() % 9 + 1;
68     printf("answer: %d\n", x);
69
70     while (true) {
71         sockaddr_in clientAddr; // client 端位址資訊
72         int clientAddrLen = sizeof(clientAddr);
73         sConnect = accept(sListen, (SOCKADDR*)&clientAddr, &clientAddrLen);
74
75         if (sConnect != INVALID_SOCKET)
76         {
77             // 有 client 端成功連線過來
78             string clientIP;
79             inet_ntop(AF_INET, (void*)&clientAddr, const_cast<char *>(clientIP.c_str()), 20);
80             printf("server: got connection from %s\n", clientIP.c_str());
81             ip.push_back(clientIP);
82
83             thread newConnect(check, sConnect, ip[ip.size() - 1]);
84             newConnect.detach();
85
86         }
87     }
88
89     closesocket(sConnect);
90     closesocket(sListen);
91
92     WSACleanup();
93
94     return 0;
95 }

```

Client 端:

```
1  #include <stdio.h>
2  #include <winsock2.h>
3  #include <ws2tcpip.h>
4
5  const wchar_t* GetWC(const char* c) {
6      const size_t cSize = strlen(c) + 1;
7      wchar_t* wc = new wchar_t[cSize];
8      size_t convertedChar = 0;
9      mbstowcs_s(&convertedChar, wc, cSize, c, _TRUNCATE);
10     return wc;
11 }
12
13 int main() {
14     WSADATA wsaData;
15
16     if (WSAStartup(MAKEWORD(1, 1), &wsaData) != 0) {
17         fprintf(stderr, "WSAStartup failed.\n");
18         exit(1);
19     }
20
21     SOCKET sock;
22
23     sockaddr_in addr;
24     InetPton(AF_INET, GetWC("127.0.0.1"), &addr.sin_addr.s_addr); // 設定 IP
25     addr.sin_family = AF_INET;
26     addr.sin_port = htons(1234); // 設定 port, htons() 跟網路位元組順序有關
27
28     sock = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
29
30     connect(sock, (SOCKADDR*)&addr, sizeof(SOCKADDR));
31
32     char szBuffer[MAXBYTE] = { 0 };
33
34     int guess;
35     int iResult;
36     while (true) {
37         printf("Your guess(1-9): ");
38         scanf_s("%d", &guess);
39
40         char sendbuf[2];
41         sprintf_s(sendbuf, sizeof(sendbuf), "%d", guess);
42
43         iResult = send(sock, sendbuf, (int)strlen(sendbuf), 0);
44         if (iResult == SOCKET_ERROR) {
45             printf("send failed with error: %d\n", WSAGetLastError());
46             closesocket(sock);
47             WSACleanup();
48         }
49         else if (iResult > 0) {
50             recv(sock, szBuffer, MAXBYTE, NULL);
51             printf("Data received: %s\n", szBuffer);
52             if (strcmp(szBuffer, "Bingo!") == 0)
53                 break;
54         }
55         else if (iResult == 0) {
56             printf("Connection closed\n");
57             break;
58         }
59         else {
60             printf("recv failed with error: %d\n", WSAGetLastError());
61             closesocket(sock);
62             WSACleanup();
63             return 1;
64         }
65     }
66
67     closesocket(sock);
68
69     WSACleanup();
70
71     return 0;
72 }
```


執行結果:



The screenshot displays four terminal windows from Microsoft Visual Studio, showing the execution of a program. The windows are arranged in a 2x2 grid.

Top-Left Window: Shows a sequence of guesses (1-9) and "Data received: Fail" messages, followed by "Data received: Bingo!". Below this, a message states: "C:\Users\ap006\source\repos\client\vd4\Debug\client.exe (應用程式 14556) 已結束，出現代碼 0。若要在此處停止時自動關閉主控台，請啟用【工具】->【結束】->【結束】->【偵錯停止時，自動關閉主控台】，按任意鍵關閉此視窗。"

Top-Right Window: Shows a sequence of guesses (4-9) and "Data received: Fail" messages, followed by "Data received: Bingo!". Below this, a message states: "C:\Users\ap006\source\repos\client\vd4\Debug\client.exe (應用程式 37720) 已結束，出現代碼 0。若要在此處停止時自動關閉主控台，請啟用【工具】->【結束】->【結束】->【偵錯停止時，自動關閉主控台】，按任意鍵關閉此視窗。"

Bottom-Left Window: Shows a sequence of "server: got connection from" messages followed by a list of IP addresses and port numbers (e.g., 2.0.238.69: 1, 2.0.238.69: 2, etc.).

Bottom-Right Window: Shows a sequence of guesses (6-9) and "Data received: Fail" messages, followed by "Data received: Bingo!". Below this, a message states: "C:\Users\ap006\source\repos\client\vd4\Debug\client.exe (應用程式 8880) 已結束，出現代碼 0。若要在此處停止時自動關閉主控台，請啟用【工具】->【結束】->【結束】->【偵錯停止時，自動關閉主控台】，按任意鍵關閉此視窗。"

3.

code:

Server 端:

```
1  #include <winsock2.h>
2  #include <stdio.h>
3  #include <winsock2.h>
4  #include <Ws2tcpip.h>
5  #include <cstdlib>
6  #include <thread>
7  #include <vector>
8  #include <string>
9  #include <iostream>
10 using namespace std;
11
12 vector<pair<string, SOCKET> > ip;
13
14 const wchar_t* GetWC(const char* c) {
15     const size_t cSize = strlen(c) + 1;
16     wchar_t* wc = new wchar_t[cSize];
17     size_t convertedChar = 0;
18     mbstowcs_s(&convertedChar, wc, cSize, c, _TRUNCATE);
19     return wc;
20 }
21
22 void check(SOCKET Connect, int idx) {
23     while (true) {
24         char recvbuf[20000] = { '\0' };
25         int now = recv(Connect, recvbuf, sizeof(recvbuf), 0);
26         if (now == 0 || now == -1) break;
27
28         string mes = ip[idx].first + ": " + recvbuf;
29         cout << mes << endl;
30
31         for (int i = 0; i < ip.size(); i++)
32             send(ip[i].second, mes.c_str(), (int)mes.size(), 0);
33         memset(recvbuf, 0, sizeof(recvbuf));
34     }
35 }
36 /*
```

```

37 int main() {
38     WSADATA wsaData;
39     if (WSAStartup(MAKEWORD(1, 1), &wsaData) != 0) {
40         fprintf(stderr, "WSAStartup failed.\n");
41         exit(1);
42     }
43
44     SOCKET sListen;
45     SOCKET sConnect;
46     sockaddr_in addr;
47     InetPton(AF_INET, GetWC("127.0.0.1"), &addr.sin_addr.s_addr); // 設定 IP
48     addr.sin_family = AF_INET;
49     addr.sin_port = htons(1234); // 設定 port, htons() 跟網路位元組順序有關
50
51     sListen = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
52     // AF_INET: IPv4    AF_INET6: IPv6
53     // SOCK_STREAM: TCP  SOCK_DGRAM: UDP
54     // IPPROTO_TCP: TCP  IPPROTO_UDP: UDP
55     bind(sListen, (SOCKADDR*)&addr, sizeof(SOCKADDR)); // server 端綁定位置
56     listen(sListen, 20);
57
58
59     vector<thread> clients;
60
61     while (true) {
62         sockaddr_in clientAddr; // client 端位址資訊
63         int clientAddrLen = sizeof(clientAddr);
64         sConnect = accept(sListen, (SOCKADDR*)&clientAddr, &clientAddrLen);
65
66         if (sConnect != INVALID_SOCKET)
67         {
68             // 有 client 端成功連線過來
69             char clientIP[20];
70             inet_ntop(AF_INET, (void*)&clientAddr, clientIP, 20);
71             printf("server: got connection from %s\n", clientIP);
72             ip.emplace_back(clientIP, sConnect);
73
74             thread newConnect(check, sConnect, ip.size() - 1);
75             newConnect.detach();
76         }
77     }
78
79     closesocket(sConnect);
80     closesocket(sListen);
81
82     WSACleanup();
83
84     return 0;
85 } // */

```

Client 端:

```
1  #include <stdio.h>
2  #include <winsock2.h>
3  #include <Ws2tcpip.h>
4  #include <string>
5  #include <iostream>
6  #include <thread>
7  using namespace std;
8
9  const wchar_t* GetWC(const char* c) {
10     const size_t cSize = strlen(c) + 1;
11     wchar_t* wc = new wchar_t[cSize];
12     size_t convertedChar = 0;
13     mbstowcs_s(&convertedChar, wc, cSize, c, _TRUNCATE);
14     return wc;
15 }
16 void getMes(SOCKET sock) {
17     while (true) {
18         char szBuffer[20000] = { 0 };
19         recv(sock, szBuffer, 20000, NULL);
20         cout << szBuffer << endl;
21     }
22 }
23
24 int main() {
25     WSADATA wsaData;
26
27     if (WSAStartup(MAKEWORD(1, 1), &wsaData) != 0) {
28         fprintf(stderr, "WSAStartup failed.\n");
29         exit(1);
30     }
31
32     SOCKET sock;
33
34     sockaddr_in addr;
35     InetPton(AF_INET, GetWC("127.0.0.1"), &addr.sin_addr.s_addr); // 設定 IP
36     addr.sin_family = AF_INET;
37     addr.sin_port = htons(1234); // 設定 port, htons() 跟網路位元組順序有關
38
39     sock = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
40
41     connect(sock, (SOCKADDR*)&addr, sizeof(SOCKADDR));
42
43     char szBuffer[20000] = { 0 };
44     thread get_mes(getMes, sock);
45     get_mes.detach();
46
47     while (true) {
48         string mes;
49         getline(cin, mes);
50         char* sendbuf = const_cast<char*>(mes.c_str());
51         send(sock, sendbuf, (int)strlen(sendbuf), 0);
52     }
53
54     closesocket(sock);
55
56     WSACleanup();
57
58     return 0;
59 }
60
61 }
```

執行結果:

```
server: got connection from 2.0.192.128
server: got connection from 2.0.192.141
server: got connection from 2.0.192.149
2.0.192.128: hello
2.0.192.141: hi
2.0.192.149: how's your day
2.0.192.128: nice
2.0.192.141: great
2.0.192.149: that's sound pretty good
2.0.192.149: by the way, today is really not my day

hello
2.0.192.128: hello
2.0.192.141: hi
2.0.192.149: hi
2.0.192.149: how's your day
2.0.192.128: nice
2.0.192.141: great
2.0.192.149: that's sound pretty good
2.0.192.149: by the way, today is really not my day

2.0.192.128: hello
hi
2.0.192.141: hi
2.0.192.149: hi
2.0.192.149: how's your day
2.0.192.128: nice
great
2.0.192.141: great
2.0.192.149: that's sound pretty good
2.0.192.149: by the way, today is really not my day

2.0.192.128: hello
2.0.192.141: hi
hi
2.0.192.149: hi
how's your day
2.0.192.149: how's your day
2.0.192.128: nice
2.0.192.141: great
2.0.192.149: that's sound pretty good
2.0.192.149: that's sound pretty good
by the way, today is really not my day
2.0.192.149: by the way, today is really not my day
```