WinSockによる
ネットワークプログラミング超入門

○　第２回　ソケット　○

ソケット

ソケットとは、IPアドレスとポート番号を組み合わせたものです。通常、TCP/IPで通信を行うコンピュータは、IPアドレスを持っていますが、複数のコンピュータと同時に通信するために、補助アドレスとして0～65,535の数値が割り当てられたポートを持っています。接続を行なう場合、必ずIPアドレスとポート番号を指定するようになっています。この組み合わせのことをソケットといいます。
アプリケーションは、ソケットに対して関数を呼び出すだけで、通信手順の詳細を気にすることなくデータの送受信を行なうことができます。

ソケットの生成と破棄

ソケットの生成はsocket関数で行います。

```
// ソケット生成
SOCKET    sock = socket(AF_INET, SOCK_STREAM, 0);
```

１つめの引数はアドレスファミリと呼ばれるもので、通常は「AF_INET」を指定します。２つめの引数はソケットの種類で、TCPの場合は「SOCK_STREAM」、UDPの場合は「SOCK_DGRAM」を指定します。３つめの引数は、プロトコルの種類です。

ソケットの破棄は、closesocket関数で行います。

```
// ソケット破棄
closesocket(sock);
```

WinSock関数一覧

socket	ソケットを生成します
listen	ソケットを接続待ち状態にします
accept	待機中のソケットで接続されるまで待ちます
bind	IPアドレスとポート番号をソケットに結びつけます
connect	指定されたアドレスに接続します
recv	ソケットを使用してデータを受信します
send	ソケットを使用してデータを送信します
recvfrom	ソケットを使用して受信するデータの送信元を指定し、データを受信します
sendto	ソケットを使用して送信先を指定し、データを送信します
select	ソケットの状態が条件を満たすまで待ちます
shutdown	ソケットの送受信を無効にします
closesoket	ソケットを破棄します
htonl	long型の整数をネットワークバイトオーダーに変換します
htons	short型の整数をネットワークバイトオーダーに変換します
ntohl	long型の整数をホストバイトオーダーに変換します
ntohs	short型の整数をホストバイトオーダーに変換します
inet_addr	ドット表記のIPアドレス文字列をネットワークバイトオーダー整数に変換します
inet_ntoa	in_addr構造体をIPアドレス文字列に変換します
gethostbyaddr	IPアドレスからホスト名を取得します
gethostbyname	ホスト名からホスト情報を取得します
gethostname	ローカルマシンのホスト名を取得します
getpeername	ソケットの接続先のアドレスを取得します
getprotobyname	指定されたプロトコル名のprotoent構造体を取得します
getprotobynumber	プロトコル番号からprotoent構造体を取得します
getservbyname	サービス名およびプロトコル名からservent構造体を取得します
getservbyport	ポート番号およびプロトコル名からservent構造体を取得します
getsockname	ソケットからそのローカルアドレスが入っているsockaddr構造体を取得します
getsockopt	ソケットのオプションを取得します
setsockopt	ソケットのオプションを設定します
ioctrlsocket	接続、データ送受信時のブロッキングモードを設定します

課　題

ソケットを生成し、他のコンピュータとデータ送受信を行いましょう。

(１)以下のプログラムは、UDPでデータを受信するためのプログラムです。足りない部分を補って完成させましょう。

-receive.cpp-

```
// データ受信
#include <ここは各自考えましょう.h>
#include <iostream>

#pragma comment(lib, "ここは各自考えましょう.lib")

using namespace std;

int main()
{
    // WinSock初期化(バージョン2.2)
    WSAData   wsadat;
    if(??????????(MAKEWORD(2, 2), &wsadat) != 0) {
        cerr << "WinSock初期化失敗" << endl;
        return -1;
    }

    // ソケット生成
    SOCKET   sock = ??????(AF_INET, SOCK_DGRAM, 0);
    if(sock == INVALID_SOCKET) {
        cerr << "ソケット生成失敗" << endl;
        WSACleanup();
        return -1;
    }

    // バインド
    sockaddr_in   addr;
    addr.sin_family           = AF_INET;
    addr.sin_port             = htons(7743);    // 受信ポート
    addr.sin_addr.S_un.S_addr = INADDR_ANY;
    ????(sock, (sockaddr*)&addr, sizeof(addr));

    // データ受信
    char   recv_buff[2048];
    memset(recv_buff, 0, sizeof(recv_buff));
    recv(sock, recv_buff, sizeof(recv_buff), 0);

    cout << "受信データ : " << recv_buff << endl;

    // ソケット破棄
    ??????????(sock);

    // WinSock解放
    ??????????();

    return 0;
}
```

(２)以下のプログラムは、UDPでデータを送信するためのプログラムです。足りない部分を補って完成させましょう。

-send.cpp-

```
// データ送信
#include <ここは各自考えましょう.h>
#include <iostream>

#pragma comment(lib, "ここは各自考えましょう.lib")

using namespace std;

int main()
{
    // WinSock初期化(バージョン2.2)
    WSAData   wsadat;
    if(??????????(MAKEWORD(2, 2), &wsadat) != 0) {
        cerr << "WinSock初期化失敗" << endl;
        return -1;
    }

    // ソケット生成
    SOCKET   sock = ??????(AF_INET, SOCK_DGRAM, 0);
    if(sock == INVALID_SOCKET) {
        cerr << "ソケット生成失敗" << endl;
        WSACleanup();
        return -1;
    }

    // データ送信
    char          send_data[256] = "代詩雫津門夢";              // 送信データ
    sockaddr_in   addr;
    addr.sin_family              = AF_INET;
    addr.sin_port                = htons(7743);                // 送信ポート
    addr.sin_addr.S_un.S_addr    = inet_addr("192.168.0.255");  // 送信アドレス
    sendto(sock, send_data, strlen(send_data), 0, (sockaddr*)&addr, sizeof(addr));

    cout << send_data << "の送信完了" << endl;

    // ソケット破棄
    ??????????(sock);

    // WinSock解放
    ??????????();

    return 0;
}
```

(３)(１)および(２)のプログラムを変更し、何度でもデータを送受信できるようにしましょう。