**第一次作业：**

1、编写C程序读取下列文件中的内容，计算张军同学的高考总分并指出最高分科目

文件名为a.txt,其中的内容为：

语文:90,英语:89,政治:78,生物:78,物理:92,化学:87,数学:76

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

int main(void)

{

FILE \*fp;

int ninzu = 0;

char buf[100] = {0};

char name[10][10] = {0};

int score[10];

int i = 0;

if((fp = fopen("f:\\TXT.txt","r")) == NULL)

printf("wrong\n");

else

{

fread(buf,sizeof(buf)-1,1,fp);

char \*p = buf;

char \*p1 = p;

char \*p2 = p;

char buf1[10] = {0};

while(p1!=NULL)

{

p1 = strstr(p,",");

p2 = strstr(p,":");

memcpy(name[i],p,p2-p);

if(p1!=NULL)

memcpy(buf1,p2+1,p1-p2);

else

strcpy(buf1,p2+1);

score[i++] = atoi(buf1);

p = p1 +1;

}

}

fclose(fp);

int all = 0;

int maxIndex;

int maxScore = 0;

for(int j=0;j<i;j++)

{

all += score[j];

if(score[j]>maxScore)

{

maxScore = score[j];

maxIndex = j;

}

}

printf("总分:%d, 最高分为:%s:%d\n",all,name[maxIndex],score[maxIndex]);

return(0);

}

2、编写C程序提取下列字符串中的目录名，文件名和后缀名

c:\windows\path\networks.docx

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int main()

{

char s[]="c:\\windows\\path\\networks.docx";

char \*p=s;

char \*p1=s;

char buf[1024]={0};

while(p1=NULL)

{

p1=strstr(p,"\\");

if(p1!=NULL)

p=p1+1;

}

memcpy(buf,s,p-s-1);

printf("path=%s\n",buf);

p1=strstr(p,".");

memset(buf,0,sizeof(buf));

memcpy(buf,p,p1-p);

printf("filename=%s\n",buf);

memset(buf,0,sizeof(buf));

strcpy(buf,p1+1);

printf("fileExt=%s\n",buf);

return 0;

}

3、编写C程序实现将字符串表示的IP地址转换为32位整数表示的IP地址。

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int main()

{

char ip[]="192.168.0.100";

char \*p=ip;

char \*p1=p;

unsigned int uip=0;

int i=0;

char buf[4];

while(p1=NULL)

{

memset(buf,0,sizeof(buf));

p1=strstr(p,".");

if(p1=NULL)

strcpy(buf,p);

else

memcpy(buf,p,p1-p);

uip+=atoi(buf)<<(24-i\*8);

p=p1+1;

i++;

}

printf("uip=%x\n",uip);

return 0;

}

4、编写C程序实现将整数表示的IP地址转换为字符串表示的IP地址。

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int main()

{

unsigned int uip=0xc0a80064;

char buf[17]={0};

printf("ip=%d.%d.%d.%d\n",uip>>24,(uip>>16)&0xff,(uip>>8)&0xff,uip&0xff);

return 0;

}

**第二次作业：**

1. 编写C函数mystrcpy实现strcpy的功能，不得使用strcpy函数。

#include <stdio.h>

#include <stdlib.h>

//char \*strcpy(char\* dest, const char \*src);

char \*mystrcpy(char \*d, const char \*s);

int main() {

char \*si="Hello World!";

char \*sc=NULL;

char \*tp=NULL;

int i=0;

if(si!=NULL) {

printf("String to copy:\n%s\n",si);

tp=si;

//获得源字符串长度

while(\*tp!='\0') {

++i;

tp++;

}

//动态创建一个等长的数组

sc=(char \*)malloc((i+1)\*sizeof(char \*));

mystrcpy(sc,si);

printf("An1:String has copied:\n%s\n",sc);

printf("An2:String has copied:\n%s\n",mystrcpy(sc,si));

}else{

printf("Sorry,the string is empty.");

}

if(si!=NULL){

si=NULL;

free(si);

}

if(sc!=NULL){

sc=NULL;

free(sc);

}

return 0;

}

char \*mystrcpy(char \*d, const char \*s) {

char \*tmp=NULL;

//循环赋值到目的字符串

tmp=d;

while(\*s!='\0'){

\*d=\*s;

d++;

s++;

}

\*d='\0';

//返回复制的字符串首地址

return tmp;

}

1. 编写C程序将任意图像文件的内容用16进制显示在文本文件中。

#include <stdio.h>

#include <stdlib.h>

void main()

{

FILE \*fp1,\*fp2;

int i=0;

int len=0;

short \*buff=NULL;

char c=' ';

if((fp1=fopen("pic.jpg","rb"))==NULL) {

printf("Can't open the pic file! \n");

exit(1);

}else if((fp2=fopen("hex.txt","wb"))==NULL) {

printf("Can't create the txt file! \n");

exit(1);

}else{

fseek(fp1,0,SEEK\_END);

len=ftell(fp1);

fseek(fp1,0,SEEK\_SET);

buff=(short \*)malloc(len);

fread(buff,len,1,fp1);

for (i=0;i<len/2;i++) {

fprintf(fp2,"%04X%c",buff[i],c);

}

}

printf("转换输出成功!\n");

fclose(fp1);

fclose(fp2);

}

1. 编写C程序显示当前电脑的系统时间。

#include <stdio.h>

#include <windows.h>

#include <time.h>

int main() {

time\_t timep;

struct tm \*p;

while(1) {

time (&timep);

p=gmtime(&timep);

printf("当前系统时间: %d年%d月%d日 %d时%d分%d秒",1900+p->tm\_year,1+p->tm\_mon,p->tm\_mday,8+p->tm\_hour,p->tm\_min,p->tm\_sec);

Sleep(1000);

system("cls");

}

return 0;

}

1. 编写C程序完成文件复制功能。

#include<stdio.h>

int main(int argc, char \*argv[]) {

int c;

FILE \*fp1, \*fp2;

if(argc==1) {

printf("本程序用于复制文件\n");

return 1;

}

fp1 = fopen(argv[1], "rb");

if(fp1==NULL) {

printf("打开源文件 %s 错误",argv[1]);

return -1;

}

fp2 = fopen(argv[2], "wb");

if(fp2==NULL) {

printf("复制文件 %s 错误",argv[2]);

return -1;

}

while((c=fgetc(fp1))!=EOF){

fputc(c, fp2);

}

printf("复制文件 %s 完成\n新文件名 %s\n",argv[1],argv[2]);

fclose(fp1);

fclose(fp2);

return 0;

}

**第三次作业：**

1、编写一个最简单的TCP客户端和服务器程序，实现客户端向服务器发送以下学生信息，

使用STU打包学生信息

typedef struct STU

{

unsigned char age; //年龄

char name[10]; //姓名

char no[10]; //学号

}STU, \*PSTU;

客户端

//只考虑核心功能的实现,且双方已经知道传输的结构体结构

#include <winsock2.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#pragma comment(lib,"ws2\_32.lib")

typedef struct STU

{

unsigned char age;

char name[10];

char no[10];

}STU, \*PSTU;

int main()

{

STU mystu;

mystu.age='9';

strcpy(mystu.name,"Jack");

strcpy(mystu.no,"123456");

printf("STU结构体信息如下:\n姓名:%s\n年龄:%c\n学号:%s\n\n",mystu.name,mystu.age,mystu.no);

char stubuf[100]={0};

strcat(stubuf,mystu.name);

strcat(stubuf,"#");

char strage[2]={0};

strage[0]=mystu.age;

strcat(stubuf,strage);

strcat(stubuf,"#");

strcat(stubuf,mystu.no);

printf("STU处理后如下:%s\n\n",stubuf);

WSADATA wsaData;

struct sockaddr\_in servaddr;

WSAStartup(0x202, &wsaData);

SOCKET ConnectSocket = socket(AF\_INET,SOCK\_STREAM,IPPROTO\_TCP);

memset(&servaddr,0,sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(7210);

servaddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

char buf[100] = {0};

printf("客户端启动成功...\n");

connect(ConnectSocket,(struct sockaddr \*)&servaddr, sizeof(servaddr));

printf("\n连接服务器成功..\n");

memset(buf,0,sizeof(buf));

strcpy(buf,stubuf);

send(ConnectSocket,buf,strlen(buf),0);

printf("发送数据:%s\n",buf);

memset(buf,0,sizeof(buf));

recv(ConnectSocket,buf,sizeof(buf),0);

printf("接受数据:%s\n",buf);

closesocket(ConnectSocket);

WSACleanup();

printf("断开服务器成功..\n\n");

getchar();

return 0;

}

服务器

//只考虑核心功能的实现,且双方已经知道传输的结构体结构

#include <winsock2.h>

#include <stdio.h>

#include <stdlib.h>

#pragma comment(lib,"ws2\_32.lib")

typedef struct STU

{

unsigned char age;

char name[10];

char no[10];

}STU, \*PSTU;

int main(){

WSADATA wsaData;

struct sockaddr\_in servaddr;

WSAStartup(0x202, &wsaData);

SOCKET ListenSocket = socket(AF\_INET,SOCK\_STREAM,IPPROTO\_TCP);

memset(&servaddr,0,sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(7210);

servaddr.sin\_addr.s\_addr = 0;

bind(ListenSocket, (struct sockaddr \*)&servaddr,sizeof(servaddr));

listen(ListenSocket,1024);

SOCKET ConnectSocket;

char buf[100] = {0};

printf("服务器启动成功...\n");

STU mystu;

memset(buf,0,sizeof(buf));

ConnectSocket = accept(ListenSocket,NULL,NULL);

printf("\n与客户端建立连接..\n");

recv(ConnectSocket,buf,sizeof(buf)-1,0);

printf("接收数据:%s\n",buf);

strcpy(mystu.name,strtok(buf,"#"));

char strage[2]={0};

strcpy(strage,strtok(NULL,"#"));

mystu.age=strage[0];

strcpy(mystu.no,strtok(NULL,"#"));

memset(buf,0,sizeof(buf));

strcpy(buf,"Success!");

send(ConnectSocket,buf,strlen(buf),0);

printf("发送数据:%s\n",buf);

closesocket(ConnectSocket);

printf("与客户端断开连接..\n\n");

closesocket(ListenSocket);

WSACleanup();

printf("接收到的STU结构体信息如下:\n姓名:%s\n年龄:%c\n学号:%s\n\n",mystu.name,mystu.age,mystu.no);

getchar();

return 0;

}

1. 编写一个能够传输文件的TCP客户端和服务器程序。

客户端

//只考虑核心功能的实现,且双方已经知道具体的传输文件名和文件类型,另外仅支持最大4KB文件传输

#include <winsock2.h>

#include <stdio.h>

#include <stdlib.h>

#include <windows.h>

#pragma comment(lib,"ws2\_32.lib")

int main()

{

WSADATA wsaData;

struct sockaddr\_in servaddr;

WSAStartup(0x202, &wsaData);

SOCKET ConnectSocket = socket(AF\_INET,SOCK\_STREAM,IPPROTO\_TCP);

memset(&servaddr,0,sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(7210);

servaddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

char buf[4096] = {0};

printf("客户端启动成功...\n");

connect(ConnectSocket,(struct sockaddr \*)&servaddr, sizeof(servaddr));

printf("\n连接服务器成功..\n");

memset(buf,0,sizeof(buf));

FILE \*fp=fopen("TestSendFile.txt","r");

int i=0;

char c;

while((c=fgetc(fp))!=EOF){

buf[i]=c;

i++;

}

printf("要发送的TestSendFile.txt文件内容为:%s\n",buf);

fclose(fp);

send(ConnectSocket,buf,strlen(buf),0);

printf("发送数据:%s\n",buf);

memset(buf,0,sizeof(buf));

recv(ConnectSocket,buf,sizeof(buf),0);

printf("接受数据:%s\n",buf);

closesocket(ConnectSocket);

WSACleanup();

printf("断开服务器成功..\n\n");

getchar();

return 0;

}

服务器

//只考虑核心功能的实现,且双方已经知道具体的传输文件名和文件类型,另外仅支持最大4KB文件传输

#include <winsock2.h>

#include <stdio.h>

#include <stdlib.h>

#pragma comment(lib,"ws2\_32.lib")

int main(){

WSADATA wsaData;

struct sockaddr\_in servaddr;

WSAStartup(0x202, &wsaData);

SOCKET ListenSocket = socket(AF\_INET,SOCK\_STREAM,IPPROTO\_TCP);

memset(&servaddr,0,sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(7210);

servaddr.sin\_addr.s\_addr = 0;

bind(ListenSocket, (struct sockaddr \*)&servaddr,sizeof(servaddr));

listen(ListenSocket,1024);

SOCKET ConnectSocket;

char buf[100] = {0};

printf("服务器启动成功...\n");

memset(buf,0,sizeof(buf));

ConnectSocket = accept(ListenSocket,NULL,NULL);

printf("\n与客户端建立连接..\n");

recv(ConnectSocket,buf,sizeof(buf)-1,0);

printf("接收数据:%s\n",buf);

FILE \*fp=fopen("TestSendFile.txt","wb");

int i=0;

char c;

while(buf[i]!='\0'){

fputc(buf[i],fp);

i++;

}

fclose(fp);

memset(buf,0,sizeof(buf));

strcpy(buf,"Success!");

send(ConnectSocket,buf,strlen(buf),0);

printf("发送数据:%s\n",buf);

closesocket(ConnectSocket);

printf("与客户端断开连接..\n\n");

closesocket(ListenSocket);

WSACleanup();

printf("TestSendFile.txt已成功接收!\n");

getchar();

return 0;

}

1. 请描述TCP发送所应考虑哪些问题？
2. 请描述TCP接收所应考虑哪些问题？
3. 如何理解TCP的可靠性？
4. 客户端如何检测服务器菪机、应用程序崩溃现象？
5. 服务器绑定IP地址时使用0表示什么意思？
6. 同一台服务器上的端口号能否被多个应用程序同时使用？为什么？

**第四次作业：**

1、UDP适用场合是那些？

2、UDP传输特点有哪些？

3、编写一个简单的UDP服务器和客户端程序。

服务器

#include <winsock2.h>

#include <stdio.h>

#include <stdlib.h>

#pragma comment(lib,"ws2\_32.lib")

int main(){

SOCKET ServerSocket;

WSADATA wsaData;

struct sockaddr\_in servaddr;

WSAStartup(0x202, &wsaData);

memset(&servaddr,0,sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(7210);

servaddr.sin\_addr.s\_addr = 0;

ServerSocket=socket(AF\_INET,SOCK\_DGRAM,0);

const int on=1;

setsockopt(ServerSocket, SOL\_SOCKET, SO\_REUSEADDR, (char \*)&on,sizeof(on));

bind(ServerSocket, (struct sockaddr \*)&servaddr, sizeof(servaddr));

printf("服务器启动成功...\n\n");

struct sockaddr\_in cliaddr;

int addrlen = sizeof( sockaddr\_in );

char buf[100];

memset(buf,0,sizeof(buf));

recvfrom(ServerSocket, buf, 100, 0,(SOCKADDR \*)&cliaddr, &addrlen);

printf("接收数据:%s\n",buf);

sendto(ServerSocket, buf, sizeof(buf), 0, (SOCKADDR \*)&cliaddr, addrlen);

printf("发送数据:%s\n",buf);

closesocket(ServerSocket);

WSACleanup();

printf("与客户端断开连接..\n\n");

getchar();

return 0;

}

客户端

#include <winsock2.h>

#include <stdio.h>

#include <stdlib.h>

#pragma comment(lib,"ws2\_32.lib")

int main(){

SOCKET ClientSocket;

WSADATA wsaData;

struct sockaddr\_in servaddr;

WSAStartup(0x202, &wsaData);

memset(&servaddr,0,sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(7210);

servaddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

ClientSocket=socket(AF\_INET,SOCK\_DGRAM,0);

connect(ClientSocket, (struct sockaddr\*)&servaddr, sizeof(servaddr));

printf("客户端启动成功...\n\n");

char buf[100];

memset(buf,0,sizeof(buf));

strcpy(buf,"This is an UDP packet");

sendto(ClientSocket,buf,strlen(buf), 0, (SOCKADDR \*)&servaddr,sizeof(sockaddr\_in));

printf("发送数据:%s\n", buf);

memset(buf,0,sizeof(buf));

recvfrom(ClientSocket, buf, 100 ,0, NULL, NULL);

printf("接收数据:%s\n", buf);

closesocket(ClientSocket);

WSACleanup();

printf("与服务器断开连接..\n\n");

getchar();

return 0;

}

4、创建一个简单的多线程程序，并简要说明多线程的运行特点。

#include <iostream>

#include <windows.h>

using namespace std;

DWORD WINAPI Fun(LPVOID lpParamter){

Sleep(500);

while(1) {

cout<<"I'm child thread!"<<endl;

Sleep(1000);

}

}

int main() {

HANDLE hThread = CreateThread(NULL, 0, Fun, NULL, 0, NULL);

CloseHandle(hThread);

while(1) {

cout<<"I'm parent thread!"<<endl;

Sleep(1000);

}

return 0;

}

5、UDP的不可靠性体现在哪些方面。

6、如何在C程序调用外部程序，例如调用计算器程序calc，请编写程序。

解法一：

#include <stdio.h>

#include <windows.h>

int main() {

int i=0;

char s[20]={0};

while(s[0]==0){

printf("请输入需要打开的程序或路径(如calc):");

scanf("%s",&s);

if(s[0]==0) {

for(i=0;i<20;i++) s[i]='\0';

}

printf("\n");

}

if(s[0]!=0) system(s);

return 0;

}

解法二：

#include <stdio.h>

#include <windows.h>

int main() {

int i=0;

char s[20]={0};

while(s[0]==0){

printf("请输入需要打开的程序或路径(如calc):");

scanf("%s",&s);

if(s[0]==0) {

for(i=0;i<20;i++) s[i]='\0';

}

printf("\n");

}

if(s[0]!=0) system(s);

return 0;

}

**第五次作业：**

1、原始套接字适用场合是那些？

2、原始套接字的接收特点是什么？

3、什么情况需要我们可以控制原始套接字发送数据包的IP头。

4、什么情况需要我们可以接收所有流经网卡的数据包。

5、创建原始套接字如何设置协议类型？

6、编写一个捕获网卡上所有数据包的程序。

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <WinSock2.h>

#include <WS2tcpip.h>

#include <stdlib.h>

#include <Windows.h>

#include <string.h>

#pragma comment(lib,"Ws2\_32.lib")

#define SOURCE\_PORT 7234

#define MAX\_RECEIVEBYTE 255

#define MAX\_ADDR\_LEN 32

#define SIO\_RCVALL (IOC\_IN|IOC\_VENDOR|1)//定义网卡为混杂模式

typedef struct ip\_hdr//定义IP首部

{

unsigned char h\_verlen;//4位首部长度，4位IP版本号

unsigned char tos;//8位服务类型TOS

unsigned short tatal\_len;//16位总长度

unsigned short ident;//16位标示

unsigned short frag\_and\_flags;//偏移量和3位标志位

unsigned char ttl;//8位生存时间TTL

unsigned char proto;//8位协议（TCP,UDP或其他）

unsigned short checksum;//16位IP首部检验和

unsigned int sourceIP;//32位源IP地址

unsigned int destIP;//32位目的IP地址

}IPHEADER;

typedef struct tsd\_hdr//定义TCP伪首部

{

unsigned long saddr;//源地址

unsigned long daddr;//目的地址

char mbz;

char ptcl;//协议类型

unsigned short tcpl;//TCP长度

}PSDHEADER;

typedef struct tcp\_hdr//定义TCP首部

{

unsigned short sport;//16位源端口

unsigned short dport;//16位目的端口

unsigned int seq;//32位序列号

unsigned int ack;//32位确认号

unsigned char lenres;//4位首部长度/6位保留字

unsigned char flag;//6位标志位

unsigned short win;//16位窗口大小

unsigned short sum;//16位检验和

unsigned short urp;//16位紧急数据偏移量

}TCPHEADER;

typedef struct udp\_hdr//定义UDP首部

{

unsigned short sport;//16位源端口

unsigned short dport;//16位目的端口

unsigned short len;//UDP 长度

unsigned short cksum;//检查和

}UDPHEADER;

typedef struct icmp\_hdr//定义ICMP首部

{

unsigned short sport;

unsigned short dport;

unsigned char type;

unsigned char code;

unsigned short cksum;

unsigned short id;

unsigned short seq;

unsigned long timestamp;

}ICMPHEADER;

int main(int argc, char \*\*argv)

{

SOCKET sock;

WSADATA wsd;

char recvBuf[65535] = { 0 };

char temp[65535] = { 0 };

DWORD dwBytesRet;

int pCount = 0;

unsigned int optval = 1;

unsigned char\* dataip = NULL;

unsigned char\* datatcp = NULL;

unsigned char\* dataudp = NULL;

unsigned char\* dataicmp = NULL;

int lentcp, lenudp, lenicmp, lenip;

char TcpFlag[6] = { 'F', 'S', 'R', 'A', 'U' };//定义TCP标志位

WSAStartup(MAKEWORD(2, 1), &wsd);

if ((sock = socket(AF\_INET, SOCK\_RAW, IPPROTO\_IP)) == SOCKET\_ERROR)//创建一个原始套接字

{

exit(0);

}

char FAR name[MAXBYTE];

gethostname(name, MAXBYTE);

struct hostent FAR\* pHostent;

pHostent = (struct hostent\*)malloc(sizeof(struct hostent));

pHostent = gethostbyname(name);

SOCKADDR\_IN sa;

sa.sin\_family = AF\_INET;

sa.sin\_port = htons(1);//原始套接字没有端口的概念，所以这个值随便设置

memcpy(&sa.sin\_addr,pHostent->h\_addr\_list[0],pHostent->h\_length);//设置本机地址

bind(sock, (SOCKADDR\*)&sa, sizeof(sa));//绑定

if (WSAGetLastError() == 10013)

{

exit(0);

}

//设置网卡为混杂模式，也叫泛听模式。可以侦听经过的所有的包。

WSAIoctl(sock, SIO\_RCVALL, &optval, sizeof(optval), NULL, 0, &dwBytesRet,NULL,NULL);

UDPHEADER \* pUdpheader;//UDP头结构体指针

IPHEADER \* pIpheader;//IP头结构体指针

TCPHEADER \* pTcpheader;//TCP头结构体指针

ICMPHEADER \* pIcmpheader;//ICMP头结构体指针

char szSourceIP[MAX\_ADDR\_LEN], szDestIP[MAX\_ADDR\_LEN];//源IP和目的IP

SOCKADDR\_IN saSource, saDest;//源地址结构体，目的地址结构体

//设置各种头指针

pIpheader = (IPHEADER\*)recvBuf;

pTcpheader = (TCPHEADER\*)(recvBuf + sizeof(IPHEADER));

pUdpheader = (UDPHEADER\*)(recvBuf + sizeof(IPHEADER));

pIcmpheader = (ICMPHEADER\*)(recvBuf + sizeof(IPHEADER));

int iIphLen = sizeof(unsigned long)\*(pIpheader->h\_verlen & 0x0f);

while (1)

{

memset(recvBuf, 0, sizeof(recvBuf));//清空缓冲区

recv(sock, recvBuf, sizeof(recvBuf), 0);//接收包

//获得源地址和目的地址

saSource.sin\_addr.s\_addr = pIpheader->sourceIP;

strncpy(szSourceIP, inet\_ntoa(saSource.sin\_addr), MAX\_ADDR\_LEN);

saDest.sin\_addr.s\_addr = pIpheader->destIP;

strncpy(szDestIP, inet\_ntoa(saDest.sin\_addr), MAX\_ADDR\_LEN);

//计算各种包的长度（只有判断是否是该包后才有意义，先计算出来）

lenip = ntohs(pIpheader->tatal\_len);

lentcp = ntohs(pIpheader->tatal\_len) - (sizeof(IPHEADER) + sizeof(TCPHEADER));

lenudp = ntohs(pIpheader->tatal\_len) - (sizeof(IPHEADER) + sizeof(UDPHEADER));

lenicmp = ntohs(pIpheader->tatal\_len) - (sizeof(IPHEADER) + sizeof(ICMPHEADER));

//判断是否是TCP包

if (pIpheader->proto == IPPROTO\_TCP&&lentcp != 0)

{

pCount++;//计数加一

dataip = (unsigned char \*)recvBuf;

datatcp = (unsigned char \*)recvBuf + sizeof(IPHEADER) + sizeof(TCPHEADER);

system("cls");

printf("\n#################数据包[%i]=%d字节数据#############", pCount,

lentcp);

printf("\n\*\*\*\*\*\*\*\*\*\*IP协议头部\*\*\*\*\*\*\*\*\*\*\*");

printf("\n标示：%i", ntohs(pIpheader->ident));

printf("\n总长度：%i", ntohs(pIpheader->tatal\_len));

printf("\n偏移量：%i", ntohs(pIpheader->frag\_and\_flags));

printf("\n生存时间：%d",pIpheader->ttl);

printf("\n服务类型：%d",pIpheader->tos);

printf("\n协议类型：%d",pIpheader->proto);

printf("\n检验和：%i", ntohs(pIpheader->checksum));

printf("\n源IP：%s", szSourceIP);

printf("\n目的IP：%s", szDestIP);

printf("\n\*\*\*\*\*\*\*\*\*\*TCP协议头部\*\*\*\*\*\*\*\*\*\*\*");

printf("\n源端口：%i", ntohs(pTcpheader->sport));

printf("\n目的端口：%i", ntohs(pTcpheader->dport));

printf("\n序列号：%i", ntohs(pTcpheader->seq));

printf("\n应答号：%i", ntohs(pTcpheader->ack));

printf("\n检验和：%i", ntohs(pTcpheader->sum));

printf("\n标志位：");

unsigned char FlagMask = 1;

int k;

//打印标志位

for (k = 0; k < 6; k++)

{

if ((pTcpheader->flag)&FlagMask)

printf("%c", TcpFlag[k]);

else

printf(" ");

FlagMask = FlagMask << 1;

}

//打印出前100个字节的十六进制数据

printf("\n数据：\n");

for (int i = 0; i < 100; i++)

{

printf("%x", datatcp[i]);

}

}

//+++++++++++++++++++++++++++++

//在这里可以加入其它封包的判断和处理

//+++++++++++++++++++++++++++++

}

}