**C多线程**

XueTr中的线程入口是函数的地址，主线程入口是main函数地址

一个进程至少有一个线程

一个进程相当与一个公司，线程相当于员工，多个线程相当于多个线程

多个员工一起工作比一个员工工作的效率高，因此多线程比单线程效率高

主线程结束，子线程全部退出

要学会调试，线程，并行堆栈，并行监视

第二个参数设置为0，就设置栈的大小默认为1MB

\_beginthread(runmsg, 0, NULL);

\_beginthread(runmsg, 0, &i);//i不断变化，多个线程同时读取会出错

\_beginthread(runmsg, 0, &a[i]);//必须传入固定的线程参数

\_endthread();//结束当前线程

多线程切割规则：

能整除：线程查询长度都相同

不能整除：大多数线程的查询长度相同，最后一个线程的查询长度小于前面的相等的长度

大数据往往不能被整除

面试题:

//不用求模运算符求出余数

void main()

{

//100 % 7 = 14 \*7 + 2 余数为2

//100 - (100/7) \* 7 = 2;

}

同步和异步：

system("calc");//同步，一个一个的，必须要等待上一个结束

system("start calc");//异步，并行，不需要等待上一个结束

//beginthread返回值是线程的句柄，可以操作某个线程

句柄（HANDLE），资源的标识。

操作系统要管理和操作这些资源，都是通过句柄来找到对应的资源。

按资源的类型，又可将句柄细分成

图标句柄（HICON），

光标句柄（HCURSOR），

窗口句柄（HWND），

应用程序实例句柄（HINSTANCE）等等各种类型的句柄。

操作系统给每一个窗口指定的一个唯一的标识号即窗口句柄

HANDLE hd = \_beginthread(runmsg, 0, &a[i]);//默认是异步模式

WaitForSingleObject(hd, INFINITE);//设置为同步模式

设置随机数种子：

#include <time.h>

time\_t ts;

unsigned int data = time(&ts);//获取时间转化为无符号

srand(data);//设置随机数

p[i] = rand() % 1000;

卡内存用堆，卡CPU用栈

360的栈默认是1MB—利用堆栈溢出来攻击360

木马的dll放在qq里面最安全，因为用户一般都会用qq