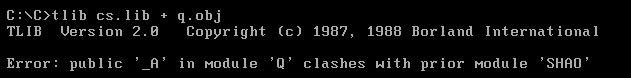
第四章

1. **tlib除了和+、-、\*号配合使用外，后边跟/C和/E的作用是什么呢？**

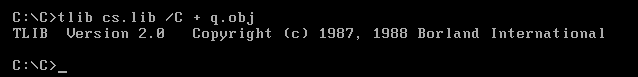
/C大小写敏感,加上-c则区分函数名的大小写。

我在 shao.c 写了一个 a() 函数，生成obj加载到cs.lib文件中

在 q.c写了一个A()函数，生成obj加载到cs.lib,如下：



是我q.c的A()函数与shao.c中的a()函数重了，说明此时不区分大小写，我在上/C继续加载如下：



看到是成功了，即 加上/C对大小写敏感

/E:建立扩展字典。建立扩展字典可以加速大的库文件的连接过程。

1. **tcc –S 和tcc –B区别**

tcc –S 直接使文件以.asm格式输出

tcc –B是先生成.asm ，没有tasm.exe则文件夹中发现了.asm文件，如果有tasm.exe,然后会调用tasm。

**（2）建立自己的库，该如何调用自己的库？**

tlib 建立自己的库，tcc mian.c \*\*.lib 就可以调用自己的库。

**（3）了解一下静态链接库和动态链接库**

静态连接库就是把(lib)文件中用到的函数代码直接链接进目标程序，程序运行的时候不再需要其它的库文件；动态链接就是把调用的函数所在文件模块（DLL）和调用函数在文件中的位置等信息链接进目标程序，程序运行的时候再从DLL中寻找相应函数代码，因此需要相应DLL文件的支持。

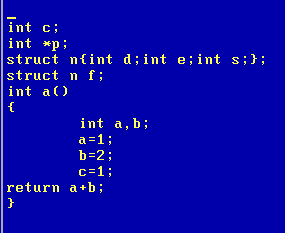
静态链接库与动态链接库都是共享代码的方式，如果采用静态链接库，则无论你愿不愿意，lib 中的指令都全部被直接包含在最终生成的 EXE 文件中了。但是若使用 DLL，该 DLL 不必被包含在最终 EXE 文件中，EXE 文件执行时可以“动态”地引用和卸载这个与 EXE 独立的 DLL 文件。

1. **Lib库文件中的文件是以函数名为关键字还是以文件名为关键字。**

答:通过生成lib.list ，查看可知，是以文件名为关键字。

1. **变量、指针变量、结构体变量能否作为全局变量加入到lib文件中？**

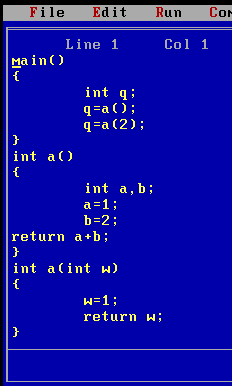
提示：这里的意思就是，将这些变量放在lib文件中，我们在用的时候使用即可，至于怎么使用，利用你们之前研究过的关键字extern。



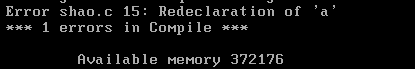


加载成功。

1. **C语言在编译链接过程中对同名函数如何处理？是否可以像其他高级语言一样进行函数重载？**

1

两个函数重名，但是功能不一样，编译链接如下：



1. **为什么还要分那么多\*.lib，如果可以，都集合到同一个lib不行吗？**

可以集成到一个lib 如果那样就比较混乱，思路不清晰，比如数学运算函数放一个库，得maths.lib,图形界面放一个库 得graphics.lib

这样每个库干什么就很清晰了。

第五章

**1、char ch; char \*p, char\*\*pa; 在p = &ch, pa=&p的情况下，\*\*pa=\*\*pa+1和\*p=\*p+1有什么不同？**

**\***\*pa=\*\*pa+1和\*p=\*p+1 结果是相同的，但是\*\*pa 是双重指针，\*pa必须指向指针变量p的内存空间，才能通过\*\*pa才指向变量ch的内存空间。

**2、二维数组b[][]与一维数组a[]中a，a[0], \*a, b, \*b, b[0], b[1], \*\*b都有着什么样的意义？**

二维数组的研究

当你定义char b[3][3]={1,2,3,4,5,6,7,8,9};

发现在内存中是按顺序存放，比如首地址为0194，则0194：1

0195：2

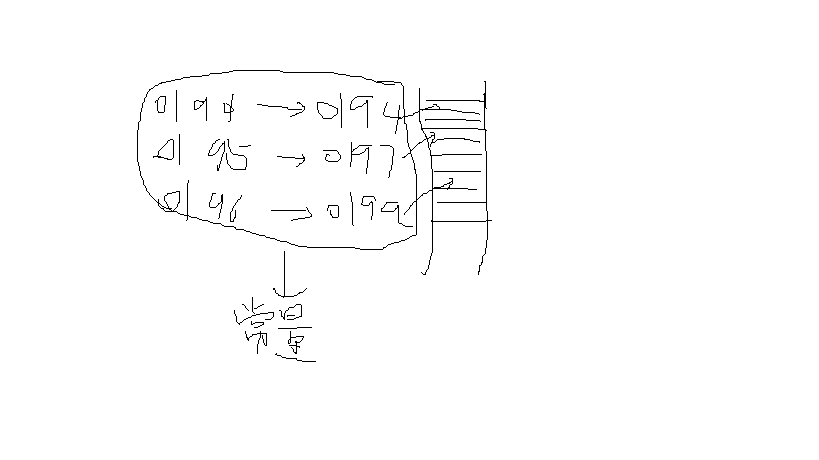
0196：3

0197：4

0198：5

…

1 2 3 4 5 6 7 8 9直接就放到一块内存处，b是地址常量其值为0194，并且指向一个地址常量\*b 也是0194，\*(b+1)是0197，好像是0195指向一个地址常量为0197。我推论觉得这就是常量指向常量，

 如果这时你定义一个char \*\*p; p=b 即0194赋值给变量p。\*p就直接指向了0194的内存空间，就和\*b含义就不一样了。 b与\*b是常量指向常量。

当你定义指针数组 char \*b[]={{1,2,3},{4,5,6},{7,8,9}}; 这样就会报错。

你可以这样定义 char q1[]={1,2,3};

char q2[]={4,5,6};

char q3[]={7,8,9};

char \*b[]={q1,q2,q3};

char \*\*p;

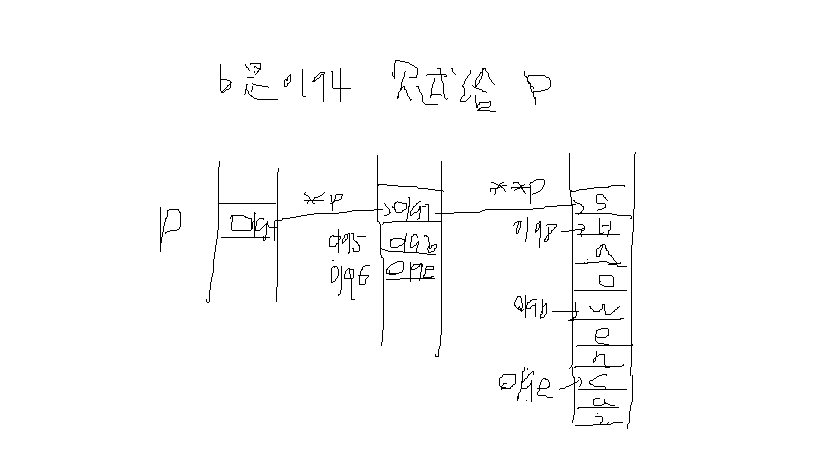
p=b

则p[0][0]就是1，p[0][1]就是2

当我定义指针数组 char \*b[]={“shao”,”wen”,”cai”}; 这样就不错报错，因为字符串都是会给出首地址的。 char \*\*p; p=b;

则 p[0][0]=’s’，p[0][1]=’h’，p[0][2]=’a’，p[0][3]=’o’

经查看内存发现是这样寻址的，如下：



**3、字符串默认以\0结尾，\0是否属于字符串的一部分，在初始化时系统是否将其与字符串放在一起？**

答：数组大小确定，如果字符串与数组大小一致，则不加\0

如果字符串大，则会出现错误

如果字符串小，则空余全部补0.

数组大小不确定，分配给数组的大小为字符串长度+\0。

**4、n=(int)&ch；去掉强制类型转换后，会编译出警告，为什么？从底层观察，强制类型转换时，不是直接将地址放到内存单元里，而是通过寄存器作为中介，为什么？**

总结：&取出来的就是指针，所以取地址符后需要赋值给指针变量，如果赋值给变量，则会出现警告，只有将指针（有效地址）强制转化为int型数据赋值给n这时候编译器才认为正确。通过汇编发现将指针（有效地址）强制转化为int型数据是通过将指针放到AX来实现。

单单& 相当于 offset 或者 lea 就是取有效地址,默认的段为ds段。如果加上（far \*），则&(取地址付)和far\* 共同作用取出段地址和偏移地址。

如 char far \*p; p=&ch，p=(char far \*)0x0100;

1. **C语言中编译器为什么将FOR语句的判断句安排到最后，而在第一次执行时利用JMP直接跳转到那里？**

答：这是汇编程序判断条件所引起的。

1. **数组名、函数名和指针的关系？**

数组名是地址 比如a[]数组的a就是地址，\*a 就是指向这个地址的内存空间，即数组名也可以说是a[0]指针，因为它可以指向这个地址的内存空间，函数名也是地址，就是指向它所在地址的内存空间，调用函数名就跑到函数名这个地址处，执行这个地址处的指令和数据，函数名也可以说是指针，

1. **指针的类型和指针所指的数据的类型有什么区别和关系？**

char型指针就是指向char型的数据

什么型的指针就是指向什么型的数据

1. **C中的指针与汇编中的寻址方式的联系？**

答：c中的数据一般都是通过AX

寻址是通过bx寄存器。