汇编语言的准备知识-给初次接触汇编者4 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/138/2021_2022__E6_B1_87_E7_BC_96_E8_AF_AD_E8_c98_138262.htm 高级语言程序的汇编解析 在高级语言中，如C和PASCAL等等，我们不再直接对硬件资源进行操作，而是面向于问题的解决，这主要体现在数据抽象化和程序的结构化。例如我们用变量名来存取数据，而不再关心这个数据究竟在内存的什么地方。这样，对硬件资源的使用方式完全交给了编译器去处理。不过，一些基本的规则还是存在的，而且大多数编译器都遵循一些规范，这使得我们在阅读反汇编代码的时候日子好过一点。这里主要讲讲汇编代码中一些和高级语言对应的地方。 1. 普通变量。通常声明的变量是存放在内存中的。编译器把变量名和一个内存地址联系起来（这里要注意的是，所谓的“ 确定的地址 ”是对编译器而言在编译阶段算出的一个临时的地址。在连接成可执行文件并加载到内存中执行的时候要进行重定位等一系列调整，才生成一个实时的内存地址，不过这并不影响程序的逻辑，所以先不必太在意这些细节，只要知道所有的函数名字和变量名字都对应一个内存的地址就行了），所以变量名在汇编代码中就表现为一个有效地址，就是放在方括号中的操作数。例如，在C文件中声明： int my_age. 这个整型的变量就存在一个特定的内存位置。语句 my_age= 32. 在反汇编代码中可能表现为： mov word ptr [007E85DA], 20 所以在方括号中的有效地址对应的是变量名。又如： char my_name[11] = "lianzi2000". 这样的说明也确定了一个地址，对应于my_name. 假设地址是007E85DC,则内存中[007E85DC]=

’l’,[007E85DD]=’i’, etc. 对my_name的访问也就是对这地址处的数据访问。 指针变量其本身也同样对应一个地址,因为它本身也是一个变量。如： char *your_name. 这时也确定变量"your_name"对应一个内存地址，假设为007E85F0. 语句your_name=my_name.很可能表现为： mov [007E85F0], 007E85DC .your_name的内容是my_name的地址。 2. 寄存器变量 在C和C 中允许说明寄存器变量。register int i. 指明i是寄存器存放的整型变量。通常，编译器都把寄存器变量放在esi和edi中。寄存器是在cpu内部的结构，对它的访问要比内存快得多，所以把频繁使用的变量放在寄存器中可以提高程序执行速度。 3. 数组 不管是多少维的数组，在内存中总是把所有的元素都连续存放，所以在内存中总是一维的。例如，int i_array[2][3]. 在内存确定了一个地址，从该地址开始的12个字节用来存贮该数组的元素。所以变量名i_array对应着该数组的起始地址，也即是指向数组的第一个元素。存放的顺序一般是i_array[0][0],[0][1],[0][2],[1][0],[1][1],[1][2] 即最右边的下标变化最快。当需要访问某个元素时，程序就会从多维索引值换算成一维索引，如访问i_array[1][1],换算成内存中的一维索引值就是1*3 1=4.这种换算可能在编译的时候就可以确定，也可能要到运行时才可以确定。无论如何，如果我们把i_array对应的地址装入一个通用寄存器作为基址，则对数组元素的访问就是一个计算有效地址的问题： . i_array[1][1]=0x16 lea ebx,xxxxxxxx .i_array 对应的地址装入ebx mov edx,04 .访问i_array[1][1]，编译时就已经确定 mov word ptr [ebx edx*2], 16 . 当然，取决于不同的编译器和程序上下文，具体实现可能不同，但这种基本的形式是确定的。从这里

也可以看到比例因子的作用（还记得比例因子的取值为1，2，4或8吗？），因为在目前的系统中简单变量总是占据1,2,4或者8个字节的长度，所以比例因子的存在为在内存中的查表操作提供了极大方便。 4. 结构和对象 结构和对象的成员在内存中也都连续存放，但有时为了在字边界或双字边界对齐，可能有些微调整，所以要确定对象的大小应该用sizeof操作符而不应该把成员的大小相加来计算。当我们声明一个结构变量或初始化一个对象时，这个结构变量和对象的名字也对应一个内存地址。举例说明： struct tag_info_struct { int age. int sex. float height. float weight. } marry. 变量marry就对应一个内存地址。在这个地址开始，有足够多的字节(sizeof(marry))容纳所有的成员。每一个成员则对应一个相对于这个地址的偏移量。这里假设此结构中所有的成员都连续存放，则age的相对地址为0，sex为2, height 为4,weight为8。 . marry.sex=0. lea ebx,xxxxxxxx .marry 对应的内存地址 mov word ptr [ebx 2], 0 ..... 对象的情况基本相同。注意成员函数具体的实现在代码段中，在对象中存放的是一个指向该函数的指针。