C语言笔记第六章指针和结构类型的关系 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/135/2021_2022_C_E8_AF_AD_E8_A8_80_E7_AC_94_c97_135126.htm 第六章 指针和结构类型的关系 可以声明一个指向结构类型对象的指针。 例十一： struct MyStruct { int a. int b. int c. } MyStruct ss={20,30,40}.//声明了结构对象ss，并把ss的三个成员初始 化为20，30和40。 MyStruct *ptr=amp.ss.//声明了一个指向结构对象ss的指针。但是它的 类型和它指向的类型和ptr是不同的。 请问怎样通过指针ptr来访问ss的三个成员变量？ 答案： ptr->a. ptr->b. ptr->c. 又请问怎样通过指针pstr来访问ss的三个成员变量？ 答案： *pstr；//访问了ss的成员a。 *(pstr 1).//访问了ss的成员b。 *(pstr 2)//访问了ss的成员c。 呵呵，虽然我在我的MSVC 6.0上调式过上述代码，但是要知道，这样使用p str来访问结构成员是不正规的，为了说明为什么不正规，让我们看看怎样通过指 针来访问数组的各个单元： 例十二： int array[3]={35,56,37}. int *pa=array. 通过指针pa访问数组array的三个单元的方法是： *pa.//访问了第0号单元 *(pa 1).//访问了第1号单元 *(pa 2).//访问了第2号单元 从格式上看倒是与通过指针访问结构成员的不正规方法的格式一样。 所有的C/C 编译器在排列数组的单元时，总是把各个数组单元存放在连续的 存储区里，单元和单元之间没有空隙。但在存放结构对象的各个成员时，在某种 编译环境下，可能会需要字对齐或双字对齐或者是别的什么对齐，需要在相邻两 个成员之间加若干个“ 填充字节 ”，这就导致各个成员之间可能会有若干个字节 的空隙。 所以，在例十二中，即使*pstr访问到了结构对

象ss的第一个成员变量a，也 不能保证*(pstr 1)就一定能访问到结构成员b。因为成员a和成员b之间可能会有 若干填充字节，说不定*(pstr 1)就正好访问到了这些填充字节呢。这也证明了指 针的灵活性。要是你的目的就是想看看各个结构成员之间到底有没有填充字节， 嘿，这倒是个不错的方法。 通过指针访问结构成员的正确方法应该是象例十二中使用指针ptr的方法。 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com