本章知识点小结

内容	实例	备注
动态内存分配函数	`void *malloc(unsigned int size);`	函数 malloc()用于分配若干字节的内存空间，返回一个指向该存储区地址的指针；若系统不能提供足够的内存单元，函数将返回空指针 NULL。
动态内存分配函数	`void *calloc(unsigned int num, unsigned int size);`	函数 calloc()用于给若干同一类型的数据项分配连续的存储空间并赋值为 0
动态内存分配函数	`void free(void *p);`	函数 free()的功能是释放向系统动态申请的由指针 p 指向的存储空间
动态内存分配函数	`void *realloc(void *p, unsigned int size);`	该函数的功能是将指针 p 所指向的存储空间的大小改为 size 个字节，函数返回值是新分配的存储空间的首地址，与原来分配的首地址不一定相同。

动态数组	`int *p = NULL;`
	`p=(int*)malloc(n*sizeof(int));`
	或者
	`p=(int*)calloc(m*n,sizeof(int));`
		需指针与动态内存分配函数联用来实现长度可变的动态数组。

void*指针		void*指针是 ANSI C 新标准中增加的一种指针类型，称为通用指针（Generic Pointer），常用来说明其基类型未知的指针，即声明了一个指针变量，但未指定它可以指向哪一种基类型的数据。

本章常见错误小结

常见错误描述	错误类型
没有意识到内存分配会不成功。内存分配未成功就使用它，将会导致非法内存访问错误。在使用内存之前，检查指针是否为空指针，可避免该错误发生	运行时错误
如果内存分配成功，但是尚未初始化就引用它，那么将会导致非法内存访问错误	运行时错误
向系统动态申请了一块内存，使用结束后，忘记了释放内存，造成内存泄漏	运行时错误
释放了内存，但却仍然继续使用它，导致产生“野指针”	运行时错误
没有变量初始化的观念，误以为没有初始化的内存的默认值全为 0	理解错误
误以为指针消亡了，它所指向的内存就一定被自动释放了	理解错误
误以为内存被释放了，指向它的指针就一定消亡了，或者成了空指针 NULL	理解错误