|  |  |
| --- | --- |
| 1. 重载new和delete 2. Operator new接口和operator delete接口 3. Malloc函数与free函数 | Operator new  Operator delete  Void \*operator new(size\_t);  Void \*operator delete(void\*) noexcept;  Void \*operator new(size\_t,void\*);//不可重载定义的版本 |
| 1. 重载new 和Delete   原本的步骤：new string(“a ”)  一：new表达式调用一个名为operator new(operator new[])的标准库函数。分配空间。二：编译器运行相应的构造函数以构造这些对象，并为其传入初始值。三：完成上两步，返回一个指向该对象的指针  Delete: delete sp;  一：对所指的对象或元素执行对应的析构函数  二：调用operator delete（operator delete[]）的标准库函数释放内存空间   1. 是指在类中或者当前文件中（类作用域或者当前作用域）重载new或delete运算符来实现自定义的new与delete操作   2) 担负起控制动态内存分配的职责，可以指定是哪个版本是自定义的（类中还是全局的）还是标准库的，用::表示全局的,在全局找不到自定义的，就找标准库的。  2. Operator new接口和operator delete接口   1. 具有8个重载版本，4个可能抛出bad\_alloc异常，4个不会 2. 可以自定义这8个版本来对分配空间和销毁空间 3. Nothrow\_t类似定义在new头文件中的一个struct，delete不允许抛出异常 4. 隐式是static的，因为operator new分配内存后分配对象，operator delete在销毁对象后回收内存，所以占整个作用域 5. Malloc函数与free函数 6. 控制具体分配内存，是调用函数 7. Malloc接受一个size\_t的字节数，分配内存并返回指向这个内存的指针，如果返回为0代表失败，free接受一个void\*由malloc返回的指针的副本   总:   1. 重载new和delete是用operator new 与 operator delete接口调用malloc与free函数来执行分配内存的过程 2. 但是不是代替了new与delete运算符，只是控制了分配内存的过程，还有创建对象与销毁对象步骤由运算符new 与delete执行 | |