|  |  |
| --- | --- |
| 1. 使用命名空间成员 2. 命名空间的别名 3. Using声明：扼要概述 4. Using指示 5. Using指示与作用域 6. Using指示示例 7. 头文件与using声明与指示 | Namespace primer = cplusplus\_sdfs;  Name inner = cpsdfs::innername;  Inner::Query q;  Using namespace std::mem;  Using namespace std; |
| 1. 使用命名空间成员 2. 命名空间名字沉长，繁琐，需别名使用 3. 命名空间的别名 4. 用直接用同义词 5. 可以指向一个嵌套的命名空间 6. Using声明：扼要概述 7. 一次只引入命名空间的一个成员 8. 可以出现在全局、局部、命名空间以及类作用域中。在类作用域中，声明语句只能指向基类成员（继承中）P546，15.5节 9. Using 命名空间名字::成员 10. Using指示 11. 所有名字都是可见的 12. 不能出现在类的作用域中 13. 在所在的作用域有效 14. Using namespace 命名空间name 15. Using指示与作用域 16. using指示一般被看做是出现在最近的外层作用域中 17. using声明在声明语句的作用域中 18. Using指示示例 19. Using指示与同名的不会冲突，只需指明哪个就行 20. Using声明与同名的会冲突，必须删其一 21. 头文件与using声明与指示 22. 若在顶层作用域有声明与指示，会将名字到所有包含了该头文件的文件中 23. 头文件应该只负责定义接口部分的名字，而不定义实现部分的名字，所以头文件最多在它的函数或命名空间内使用using声明或指示。 24. 2)的意思是，如果在头文件里在顶层作用域用using声明或指示，别的文件引入这个头文件也会包含这个using声明或指示。   所以最好在cpp文件中实现函数的定义使用using声明与指示，这样就不会出现也包含的问题。  Note：避免using指示，不然命名空间污染，二义性，延后检测，使用该库部分才检测错误  Using指示也并非一无是处，在命名空间本身的实现文件(?)中就可以使用using指示,就是实现对命名空间的成员定义的时候可以，但是得用一个cpp文件 | |