指针函数与函数指针
函数指针 Char (*p)();　　指针函数 char *p();
static_cast与dynamic_cast
类型强制转换 `Data * myData = static_cast<Data*>(f(4,5));`
new与malloc与calloc
delete[]与free
动态数组
int size=1;int *a=new int[size];　(size必须初始化)
空指针
const常量必须初始化
指针常量和常量指针
指针常量必须初始化 常量指针不需要
this常量指针
引用　　类型 &引用名=目标变量名；　　必须初始化，定义后不能再修改
指针的引用与引用的指针：没有引用的指针
声明 定义 实现 调用
常成员函数无法修改类中数据
cout<<::a; //全局对象被屏蔽后的强行访问
函数重载 不区分只有参数上const区别的两个函数，但能区分常函数和普通函数(见ppt p88，91)
名称压扎技术
extern
模版的强制转换
template每个函数前都要加
类模板和模板类
函数模板和模板函数
namespace
friend添加在public中

构造函数和析构函数
构造带参函数后不能直接创建类对象
调用无参构造函数不加()
对象本体与实体
拷贝构造函数//深拷贝&&浅拷贝
assert
explicit抑制隐形转换
return()=return
运算符重载
如果返回值会被赋值，就必须返回引用，例如=重载
如果返回值是一个局部变量，就一定不能返回引用，例如+重载
如果为了提高效率，参数可以使用引用，
而[]的实际参数经常是一个常量，常量你是无法引用，所以不能用引用作为[]的参数。
静态数据成员与静态成员函数
静态数据成员初始化不能在头文件中
静态数据成员可以成为成员函数的可选参数，而普通数据成员则不可以。
静态成员函数只能访问静态成员变量
继承
虚拟继承 菱形继承 菱形虚拟继承
虚函数和纯虚函数
多态
抽象类与纯虚函数
抽象类不能new
过载(覆盖)(override)和重载(overload)
容器类
volatile
派生类的虚函数参数与基类不同时，会发生override

若想实现重载，可以在实现派生类时，虚函数前加using <基类类名>：<函数名>；
泛化（Generalization）, 实现（Realization），关联（Association)，聚合（Aggregation），组合(Composition)，依赖(Dependency)
extern 可以放在头文件，不分配空间，头文件中不可以放变量的定义。
头文件可以放

1）值在编译时就已知的const变量的定义可放在头文件中，如：const int num=10;

2）类的定义可放在头文件中。

3）inline函数

虚基类