C 实践：用多态机制来做设计计算机等级考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/645/2021_2022_C___E5_AE_9E_E8_B7_B5_EF_c97_645609.htm 多态polymorphism是指具有多种形态的情况，它能根据单一的标记关联不同的行为。多态是面向对象程序设计的基础。在面向对象程序设计中的多态是一种运行时的多态。C 中有两种多态，称为动多态（运行时多态）和静多态（编译时多态），而静多态主要通过模板来实现，宏也是实现静多态的一种途径。其实在做软件设计时静多态的威力也是非常强大的，只不过我们经常对它疏忽了而已。 动多态的设计思想：对于相关的对象类型，确定它们之间的一个共同功能集，然后在基类中，把这些共同的功能声明为多个公共的虚函数接口。各个子类重写这些虚函数，以完成具体的功能。客户端的代码（操作函数）通过指向基类的引用或指针来操作这些对象，对虚函数的调用会自动绑定到你实际提供的子类对象上去。 下面以几何对象的设计为例。对各种几何对象如圆、矩形、直线等，都有一些共同的操作，比如画出几何对象，有重心等，我们把这些接口抽象成虚函数放在所谓的抽象基类GeoObj中，具体的几何对象类则继承这个抽象基类。如下： view plaincopy to clipboardprint? //dynahier.hpp：几何类的定义 #ifndef __GEOOBJ_H__ #define __GEOOBJ_H__ #include "coord.hpp" class GeoObj{ //几何对象的公共抽象基类 public: virtual void draw() const=0. //画出几何对象 virtual Coord center_of_gravity() const=0. //返回几何对象的重心 //... }. class Circle : public GeoObj{ //具体的几何对象类：圆 public: virtual void draw()

const. virtual Coord center_of_gravity() const. //... }. class Line : public GeoObj{ //直线类 public: virtual void draw() const. virtual Coord center_of_gravity() const. //... }. //... #endif //dynahier.hpp ：几何类的定义 #ifndef __GEOOBJ_H__ #define __GEOOBJ_H__ #include "coord.hpp" class GeoObj{ //几何对象的公共抽象基类 public: virtual void draw() const=0. //画出几何对象 virtual Coord center_of_gravity() const=0. //返回几何对象的重心 //... }. class Circle : public GeoObj{ //具体的几何对象类：圆 public: virtual void draw() const. virtual Coord center_of_gravity() const. //... }. class Line : public GeoObj{ //直线类 public: virtual void draw() const. virtual Coord center_of_gravity() const. //... }. //... #endif 客户端的使用如下： view plaincopy to clipboardprint? //dynapoly.cpp：客户端代码 #include "dynahier.hpp" #include  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