****

**JAVA进阶编程第一次实验报告**



**学 院 智算学部**

**专 业 软件工程**

**年 级 2018**

**姓 名 王申宇**

**2020年 3 月 8 日**

# JAVA进阶编程第一次实验报告

1. 需求分析（描述具体需求）

“某计算机组装公司主要销售各类组装计算机，计算机一般由CPU、内存、主板、硬盘等组件构成。”

**需要自己设计出一个计算机，该计算机包括四大组成部分：CPU、内存、主板、硬盘，每一部分有自己的属性，每个部分又有自己的品牌。**

1. 概要设计（简单描述设计思路，配合UML图）

**对于计算机的设计，由于计算机由各种部件组成，故采用 Abstract Factory 设计模式，将“抽象零件”组装成“抽象产品”。**

**（由于采用设计模式思想， 故不需要使用接口，只使用抽象类即可达成目的）**

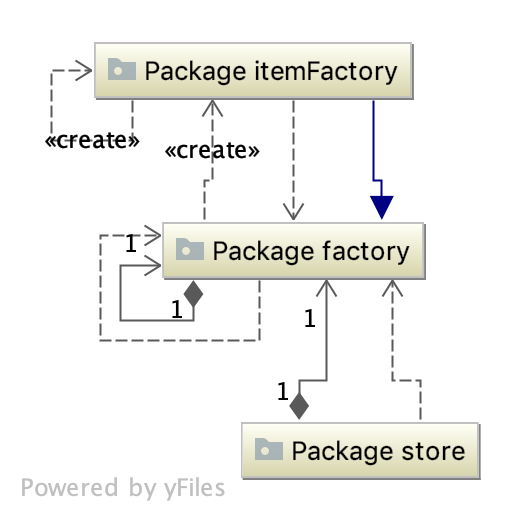
程序分为三个包：

com.shenyu.factory 包含抽象工厂、各种抽象部件、抽象产品(电脑)

com.shenyu.itemFactory 包含具体工厂、具体部件、具体产品

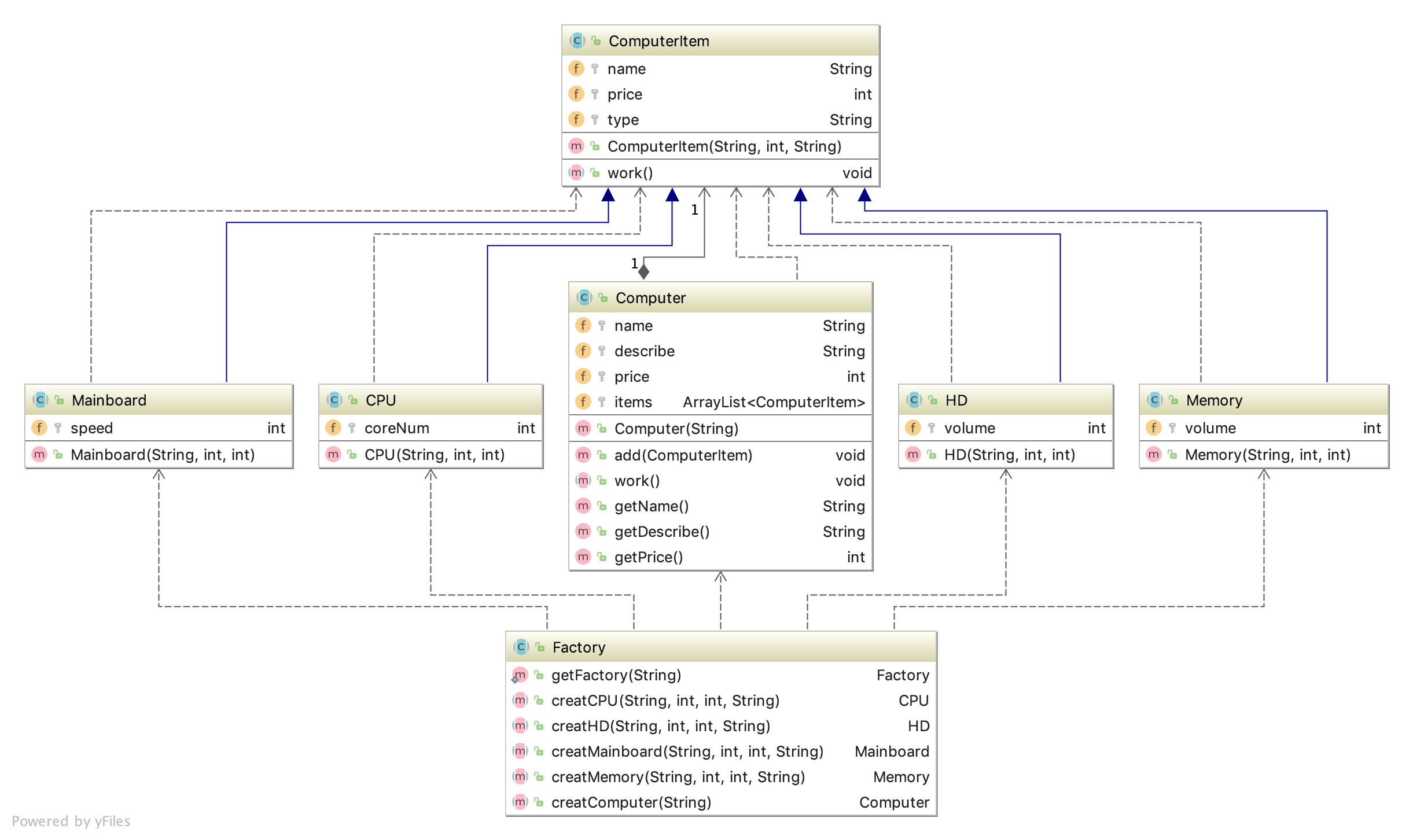
为了可复用性和降低耦合，该设计模式严格采用里氏替换原则，利用抽象零件组装成抽象产品，进而使用该产品。

附UML图：(不清楚 见文件)

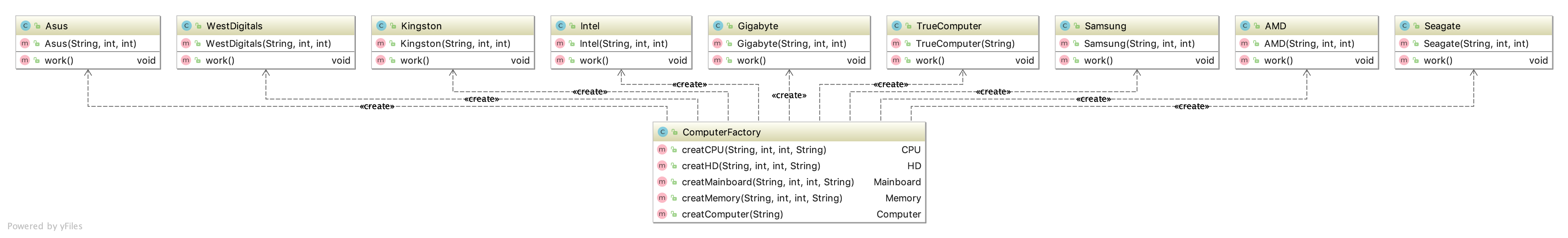




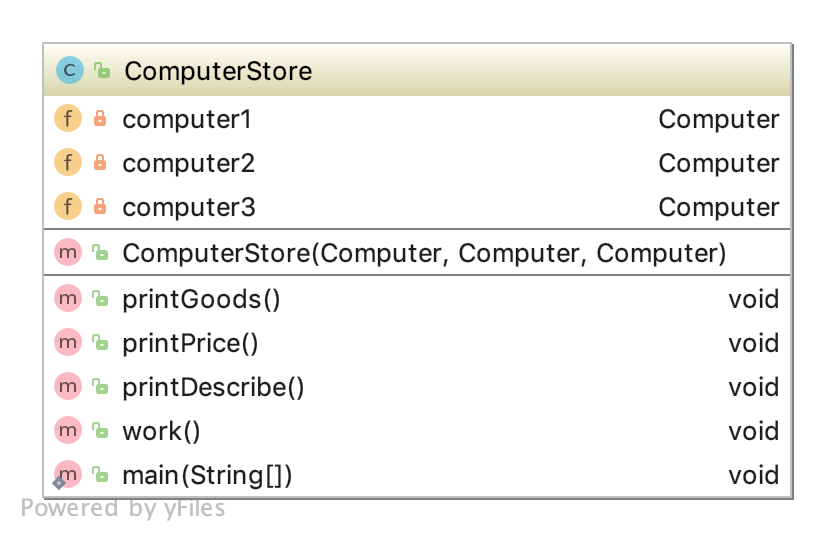
com.shenyu.factory包



com.shan.itemFactory包



com.shan.store包



1. 详细设计（详细描述具体如何实现，附代码及说明）

**factory包下：**

该包存放抽象类：

抽象零件：ComputerItem类 该类使得具体零件可以被抽象替代

四个计算机零件：CPU Memory HD Mainboard，拥有各自的属性

计算机抽象类：Computer类 抽象出的计算机类，可以使代码低耦合，可复用性强

抽象工厂类：Factory类 该抽象工厂可以创造一个具体工厂，从而用具体工厂来生产计算机及其部件

**itemFactory包下：**

该包为各种具体类：

具体工厂： ComputerFactory 继承自Factory类，用于生产计算机及其部件

具体零件：各种品牌类 分别继承自各自的抽象部件，分别有各自的工作模式，用于生成具体零件对象

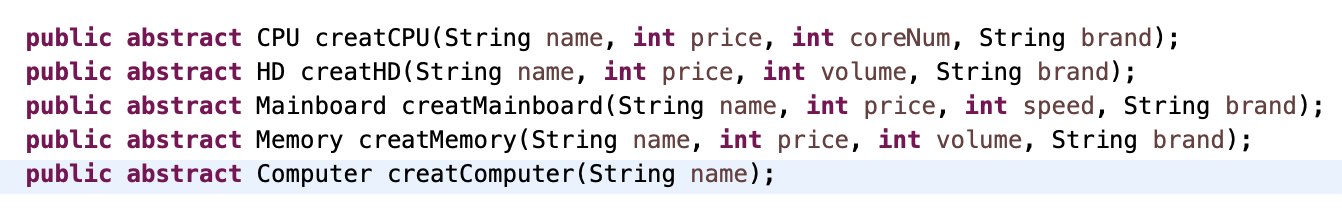
具体计算机： TrueComputer类 继承自Computer类，实现了抽象类的方法，用处生成具体计算机对象

**Store包下：**

该包存放商店 其中包含main函数：

商店：用来贩卖计算机，商店同时还有各种方法展示计算机。其中的main函数也同时承担了“使用工厂生成计算机”的任务

**其中重要的Factory抽象类中：**

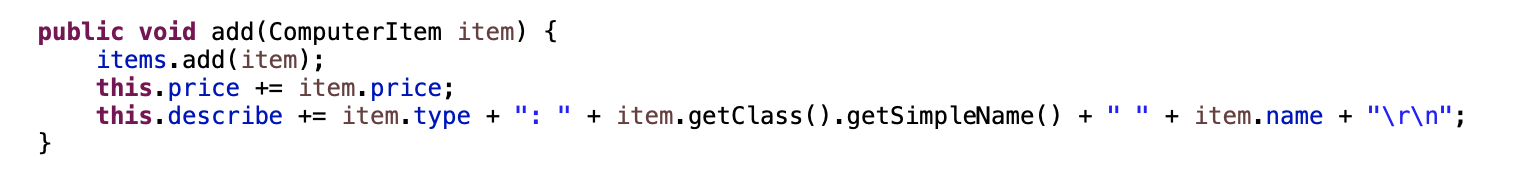


以上五个抽象函数用于产生具体对象，至于怎么实现，交给具体子类实现即可，充分体现了里氏替换原则的优点。同时，即使以后增加更多品牌，也不用更改此处代码，具有强大的可复用性。

**其中Computer抽象类中：**

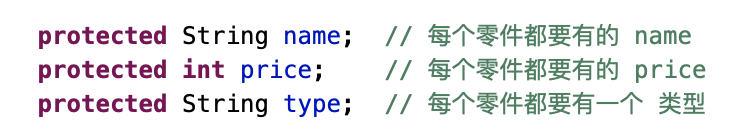


有一个ArrayList属性，用于存放ComputerItem (计算机部件)



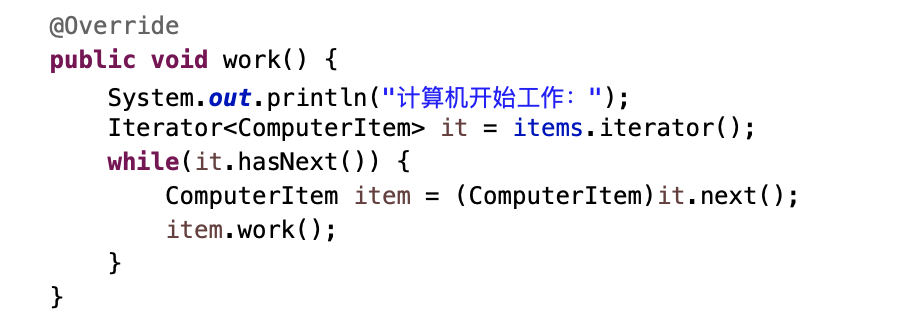
利用add函数，组装部件。高内聚 低耦合

**其中ComputerItem类中：**



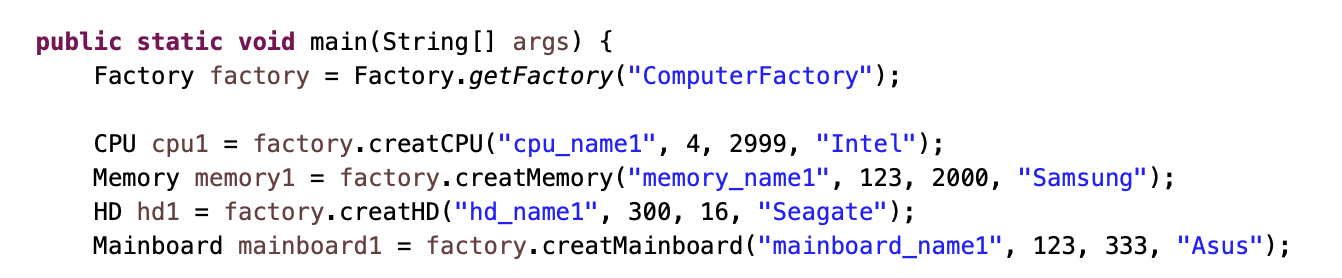
存放了部件的公共属性以及提供了抽象函数work(),具体部件怎么工作交给具体子类实现。

**其中TrueComputer类中：**

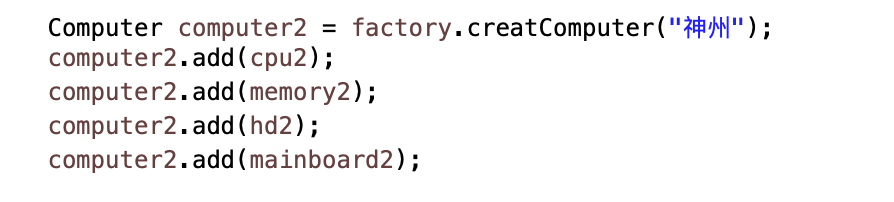


重写的work函数，利用迭代器迭代其中的部件，让每个部件工作。

**其中的ComputerStore函数中：**



Main函数通过生成一个工厂，再利用工厂来生成各种实例。完美。



再通过add函数直接组装。

1. 调试分析（在实验过程中遇到的问题以及如何解决）

遇到的问题：

1. 有时里氏替换原则把自己搞蒙了…通过修改类型解决
2. 设计模式的选择

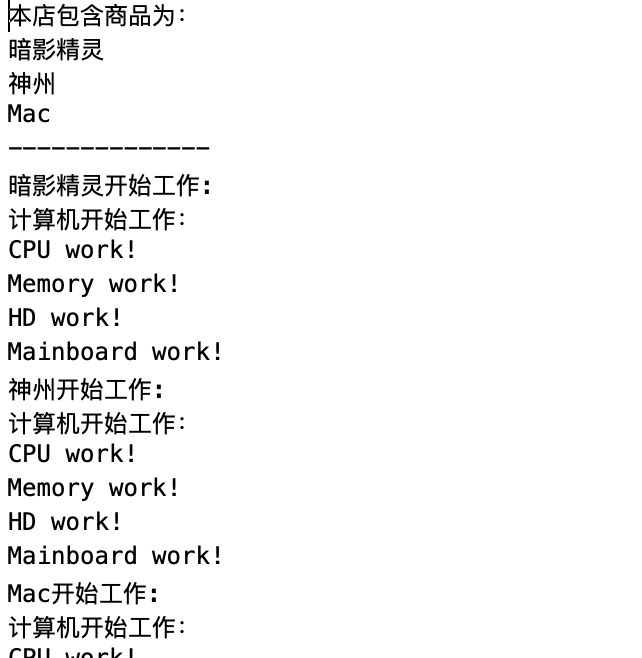
一开始本打算用 Factory 打完代码后觉得不合适…下次应该先设计UML的

1. 测试结果（描述输入和输出）

本实验main函数中提供了一个ComputerStore实例，其中包括三个Computer实例。

同时对每个Computer 进行展示描述、价格、工作

部分测试结果：



1. 总结

该设计模式的使用，可以极大地提高代码可复用性及鲁棒性。比如，如果新添加一个CPU品牌，只需实现该品牌的具体类即可，不需修改抽象类和其他具体类。

同时，也使得生成一个计算机变得容易。定义与生成分隔开，是十分合理的。

我们并不关心每个零件的具体实现，而是只是关心API，仅仅使用API就能使各个组件连接起来，组成一个计算机。

1. 执行方法

在JVM环境中运行

Main函数在ComputerStore类中，不需要参数