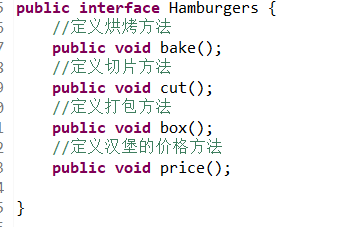
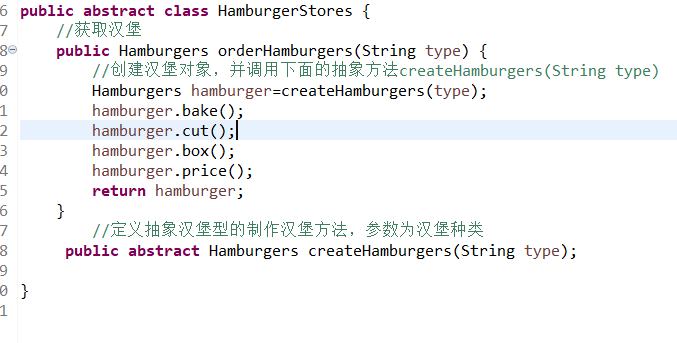
一.思路:

步骤1.创建汉堡的接口,并声明它的四个方法。



步骤2:创建三个不同种类的汉堡类实现汉堡接口。

步骤3:定义一个抽象汉堡店,其中有一个抽象方法createHamburgers()，我们在其他具体的汉堡店类中再来实现它。这么做的目的是，因为制作一个汉堡，都需要bake,cut,box，price等，所以我们把这些属性定义在这个抽象类里面，只是汉堡店的种类有差别罢了，所以我们抽象类HamburgerStores里面的属性可以通过子类来接收不同的值。



接着定义具体的汉堡店继承抽象化汉堡店，然后在creatHamburger里面用equal进行比较，相同就返回相对应汉堡类型。



接着编写不同地区不同口味的汉堡类,这几个类都是实现汉堡类接口。



步骤4：最后编写test测试类,通过调用具体工厂类的方法，从而创建不同具体产品类的实例,并调用这些产品即汉堡的四个方法(实例化汉堡店,为汉堡店添加汉堡对象)



1. 总结

刚开始我是从简单工厂模式开始学习的,发现只需要弄factory，store，hamburgers，test这四个角色,且这些都没有用到接口或者是抽象类,跟平时的逻辑很像,简单工厂其实不是一个设计模式，反而比较像是一种编程习惯,比较容易就可以写出来,但是正因为这样发现它存在很多缺点：

1、由于工厂类集中了所有产品创建逻辑，一旦不能正常工作，整个系统都要受到影响。

2、使用简单工厂模式将会增加系统中类的个数，在一定程序上增加了系统的复杂度和理解难度。

3、系统扩展困难，一旦添加新产品就不得不修改工厂逻辑，在产品类型较多时，有可能造成工厂逻辑过于复杂，不利于系统的扩展和维护。

所以决定进一步深化学习,发现工厂方法用来创建客户所需要的产品，同时隐藏了哪种具体产品类将被实例化的细节，用户只需要要关注工厂，不需要关注创建的细节！从客户端代码就可以看出！只知道对应的工厂就好！ 创建对象的细节完全封装在具体的工厂内部，而且有了抽象的工厂类，所有的具体工厂都继承了自己的父类！完美的体现了多态性！其实还想进行第三个的学习但是第三个看了好几遍后还是没什么思路,不是很明白他和工厂方法模式的区别,看得有的乱。

**㈠** 遇到的问题:

1. 一些基本概念掌握的不是很好，接口和抽象类有点混淆，分不清楚区别.因此我上网搜索了一下他们之间的区别，结果总结如下：①接口只能包含抽象方法，抽象类可以包含普通方法。②接口只能定义静态常量属性，抽象类既可以定义普通属性，也可以定义静态常量属性。③接口不包含构造方法，抽象类里可以包含构造方法。
2. 在最后运行main函数的时候运行界面没有出现任何东西，我以为出错了，想了半天考虑改一些地方，但又感觉没有错了，却不知道其实是自己在实例化方法时没有赋予它任何动作，这个看起来简单，但是有时候想得多了，就会小细节的地方卡住，我问一个同学才醒过来的。
3. 关于注释，我之前写代码有些地方就会写一些注释，因为很害怕自己过段时间再看这个代码竟不知道自己当时是怎么做出来，这次题目既然提到了，就特意上网搜了一些注释的使用规范，网址如下：https://wk.baidu.com/view/58fa3fab5022aaea998f0fc4 还有我发现我不会给类名这一行写注释，但是搜索到的也没有明确说这个，我看披萨这个例子有用／\* \*／注释，所以我就也用了。

**㈡**、收获：这次考核的收获还是不小的,从一道题当中我们可以学习到很多东西，这也是我们为什么要经常练习的原因，像从这次的题目中，我巩固了接口和抽象类的区别，还有一些其他的代码规范问题，还有进一步了解了注释相关的很多问题。让我对java的基础有了进一步的重视,基础很重要，而且细节也很重要，也让我的自学能力有了进一步的提高,还有设计模式是我以前没接触过的，通过本次考核题目的学习，丰富了我的知识面。