试用Bridge模式完成下列事情：饮料的杯子有大、中、小；行为有：加奶，加糖，啥都不加

Code：

// 接口：添加物  
interface Additive {  
 void add();  
}  
  
// 具体类：加奶  
class AddMilk implements Additive {  
 @Override  
 public void add() {  
 System.*out*.println("加了奶");  
 }  
}  
  
// 具体类：加糖  
class AddSugar implements Additive {  
 @Override  
 public void add() {  
 System.*out*.println("加了糖");  
 }  
}  
  
// 具体类：啥都不加  
class AddNothing implements Additive {  
 @Override  
 public void add() {  
 System.*out*.println("啥都没加");  
 }  
}  
  
// 抽象类：饮料  
abstract class Beverage {  
 protected Additive additive;  
  
 public Beverage(Additive additive) {  
 this.additive = additive;  
 }  
  
 public abstract void serve();  
}  
  
// 具体类：大杯饮料  
class LargeBeverage extends Beverage {  
 public LargeBeverage(Additive additive) {  
 super(additive);  
 }  
  
 @Override  
 public void serve() {  
 System.*out*.print("大杯饮料，");  
 additive.add();  
 }  
}  
  
// 具体类：中杯饮料  
class MediumBeverage extends Beverage {  
 public MediumBeverage(Additive additive) {  
 super(additive);  
 }  
  
 @Override  
 public void serve() {  
 System.*out*.print("中杯饮料，");  
 additive.add();  
 }  
}  
  
// 具体类：小杯饮料  
class SmallBeverage extends Beverage {  
 public SmallBeverage(Additive additive) {  
 super(additive);  
 }  
  
 @Override  
 public void serve() {  
 System.*out*.print("小杯饮料，");  
 additive.add();  
 }  
}  
  
// 主类  
public class Main {  
 public static void main(String[] args) {  
 // 大杯加奶饮料  
 Beverage largeMilkBeverage = new LargeBeverage(new AddMilk());  
 largeMilkBeverage.serve();  
  
 // 中杯加糖饮料  
 Beverage mediumSugarBeverage = new MediumBeverage(new AddSugar());  
 mediumSugarBeverage.serve();  
  
 // 小杯啥都不加饮料  
 Beverage smallNothingBeverage = new SmallBeverage(new AddNothing());  
 smallNothingBeverage.serve();  
 }  
}

运行结果：

