## 模板方法（Template Method）

### Intent

定义算法框架，并将一些步骤的实现延迟到子类。

通过模板方法，子类可以重新定义算法的某些步骤，而不用改变算法的结构。

### Class Diagram

### Implementation

冲咖啡和冲茶都有类似的流程，但是某些步骤会有点不一样，要求复用那些相同步骤的代码。

public abstract class CaffeineBeverage {

 final void prepareRecipe() {
 boilWater();
 brew();
 pourInCup();
 addCondiments();
 }

 abstract void brew();

 abstract void addCondiments();

 void boilWater() {
 System.out.println("boilWater");
 }

 void pourInCup() {
 System.out.println("pourInCup");
 }
}

public class Coffee extends CaffeineBeverage {
 @Override
 void brew() {
 System.out.println("Coffee.brew");
 }

 @Override
 void addCondiments() {
 System.out.println("Coffee.addCondiments");
 }
}

public class Tea extends CaffeineBeverage {
 @Override
 void brew() {
 System.out.println("Tea.brew");
 }

 @Override
 void addCondiments() {
 System.out.println("Tea.addCondiments");
 }
}

public class Client {
 public static void main(String[] args) {
 CaffeineBeverage caffeineBeverage = new Coffee();
 caffeineBeverage.prepareRecipe();
 System.out.println("-----------");
 caffeineBeverage = new Tea();
 caffeineBeverage.prepareRecipe();
 }
}

boilWater
Coffee.brew
pourInCup
Coffee.addCondiments
-----------
boilWater
Tea.brew
pourInCup
Tea.addCondiments

### JDK

* java.util.Collections#sort()
* java.io.InputStream#skip()
* java.io.InputStream#read()
* java.util.AbstractList#indexOf()