command模式：利用多态统一对外接口

public **interface Command** {

   void execute();

}

public class **CommandA implements Command** {

   public void execute() {

       System.out.println("命令A的逻辑");

  }

}

public class **CommandB implements Command** {

   public void execute() {

       System.out.println("命令B的逻辑");

  }

}

public class **Invoker** {

   private **Command** command;

   public Invoker(Command command) {

       this.command = command;

  }

   public void execute() {

       command.execute();

  }

}

命令模式是很常用的，一般就是说，你可能会执行一些操作，这些操作都实现一个接口，但是有不同的命令实现，然后命令的执行类就是同一个，你需要执行哪个命令就封装那个命令后发送给执行类即可

我之前在龙果的电商缓存架构课程里，就用了这个命令模式，读请求和写请求，封装成command，然后发送给一个异步处理线程，他不管是什么命令，执行即可，逻辑都封装在命令的内部了

其实命令模式非常适合跟工厂方法模式结合起来使用，就是使用多个工厂来制造出不同的命令类来，将命令的构造放在工厂方法中

853769620