Lab8 分析

1. 使用命令模式，让 class GameEgine 通过 class PlayerCommand 控制 class Player，让 Player 完成玩一回合（输入方向让地图移动）、加分的动作。实现了行为请求者与行为实现者之间的解耦，较好地完成了撤销等功能，也使游戏过程的流程步骤更加层次清晰简明，也很易于添加新的 PlayerCommand 来实现新的游戏功能。

2. 就命令模式及其相关方面而言，目前还存在的一些问题：

①由于两个命令的内部属性较多，所需调用的资源较大，其初始化比较繁琐；

②由于需要得知进行具体命令 RoundCommand 过后的结果信息，导致不得不在该类中多写一个 GetRoundinf()来让 GameEngine 获取信息(用数据包 class Roundinf 作返回值)，不知道这样是不是不太好；

③实现进行一回合的 PlayerCommand（class RoundCommand）中需要玩家控制棋盘移动，导致棋盘的指针会作为属性暴露在 class RoundCommand 中，感觉有点不太好。

3. 对于 bouns 功能，并没有运用命令模式，bonus 功能的逻辑是由作为 class Player 的成员函数（void GetBouns()）实现的，该函数会在 Player 加分（即调用 AddScore 函数）的过程中向所有观察者发送信息时，被用于记录 bouns 加分的日志的观察者（class PlayerBounsLog）调用，最终实现 bouns 加分功能。

这样实现起来十分简洁，也不需修改任何已有代码，我认为有一定巧妙之处。但问题在于就算没有开启 log 功能，也会产生用于记录 bouns 加分的日志的观察者（class PlayerBounsLog）以用于实现 bonus 加分功能，所有这样就需要进行 if 条件判断来看是否需要输出日志，并且导致奖励加分功能的最终完成不得不与和该具体观察者捆绑，也违反了单一职责原则，在一定程度上有悖于设计该类的初衷。但就本例而言，鄙以为利远大于弊，因此没有改动。