Rationale

Name: Jiaqi Luo, ID: jiaqiluo

I basically use General Responsibility Assignment Software Patterns (principles) (GRASP) and Observer to implement this design.

1.Observer pattern

In this homework, the Game class is the subject, the GameChangeListener is the observer, the boardPanel is the concrete observer to implement the interface GameChangeListener, which implements actual action.
When the Core part changes, Game will notify the board, something should change, such as the rack , score and so on.

2.GRASP

Controller:

The game class behaves as a controller. It can receive the message from UI, and then execute different command among other classes

High cohesion and low coupling:

When designing, I focus on that if the responsibilities of a given element are strongly related and highly focused. To speak specifically, I break programs into classes, which, I believe, can increase the cohesive properties of a system. I give a class Move, which can be included into class Game. But I break it into 2 classes. Game is just a controller for communicating information. Move is used to store words, and calculate and so on. Game owns move.

Information Expert

I used this principle to determine where to delegate responsibilities. These responsibilities include methods, computed fields, and so on. Such as when determining which class to calculate the score, I decided that Game class to calculate scores, then return score to player, and store score in player. If so, the Player class has no need to access Board class.