Rationale

Name: Jiaqi Luo, ID: jiaqiluo

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

1.Template method pattern

In this object-oriented programming, first I built a class that provides the basic steps of an algorithm design. These steps are implemented using abstract methods. Such as *Square* abstract class. It shares some basic methods and fields for all squares. Later on, subclasses can change the abstract methods to implement real actions. Such the double letter, it will give 2*value for each title. Thus the general algorithm is saved in one place but the concrete steps may be changed by the subclasses.

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.

3. Strategy pattern

To enables an algorithm's behavior to be selected at runtime, I built an interface SpecialTile, which different specific special tile will implement it. Also it is also convenient when I want to extend an extra special tile.