Shivashriganesh Mahato

Activity 9 Questions

1. The **size** method is simply an accessor for the length of the array holding the board cards. It isn’t involved in the determining of the size of board, that is done through the constructor. It is simply a way for the client class to easily access the size of the board at a given moment.
2. The selection of cards to be removed or replaced differs completely from game to game. Having a common abstract method wouldn’t work because not all games should have the ability to check if a pair of cards add up to 11, for example. So there cannot be an abstract method defined in the parent class that accounts for the removal/replacement combinations from each game.
3. An abstract class essentially inherits the functionality in interfaces that allows polymorphism. Therefore, for the **isLegal** and **anotherPlayIsPossible** methods, nothing would change. The Elevens GUI would be able to call them polymorphically because each individual board would implement these methods. However, this alternate design wouldn’t work as well because there are functionalities that are shared throughout each board (such as dealing, getting a card in the board, replacing the selected cards, etc.). These mechanisms are implemented in the current abstract Board class, and thus they don’t have to be repeatedly implemented in each child Board. With an interface instead this couldn’t happen and code reuse would not be utilized, leading to a lot of repetitions in the child Boards.