Aviel Resnick

Questions:

1. The size of the board is one of the differences between Elevens and Thirteens. Why is size not an abstract method?

Size is an instance variable, instead of an abstract method because it is an accessor to a private variable within the abstract class Board.

2. Why are there no abstract methods dealing with the selection of the cards to be removed or replaced in the array cards?

The selected cards are determined by the ElevensGUIRunner (User Inputed), and the selections are passed to the abstract class isLegal in Board to determine whether the move is valid.

3. Another way to create “IS-A” relationships is by implementing interfaces. Suppose that instead of creating an abstract Board class, we created the following Board interface, and had

ElevensBoard implement it. Would this new scheme allow the Elevens GUI to call isLegal

and anotherPlayIsPossible polymorphically? Would this alternate design work as well as

the abstract Board class design? Why or why not?

|  |
| --- |
| public interface Board  {  boolean isLegal(List<Integer> selectedCards);  boolean anotherPlayIsPossible();  } |

Implementing Board as an interface rather than an abstract class would still allow calls to isLegal and anotherPlayIsPossible, however the design would not work as well as using an abstract class, because the current “setup” allows similar games to use the same code.