Aviel Resnick

Activity 7

1. The items necessary include a deck of cards and a pair of hands. Variables would include the board size = 9, all the suits, ranks, and point values.
2. 1. Create a new deck of cards
   2. Shuffle the deck
   3. Deal 9 Cards onto a board
   4. Check for a combination, else print game over
   5. Wait for player to input combination
   6. Remove matched cards
   7. Deal new cards
   8. Repeat D-G until no new cards exist
   9. Game is won
3. Based on what I saw, I think it already contains everything necessary (just not the implementation).
4. 1. It is called on lines 59 and 68.
   2. They should be called from the anotherPlayIsPossible and isLegal methods.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 1 | 3 | 6 | 7 |

d.

|  |
| --- |
| public static printCards(ElevensBoard board) {  List<Integer> cIndexes = board.cardIndexes();   for (int x : cIndexes) {  System.out.println(boards.cards[x].toString());  } } |

e. The anotherPlayIsPossible, because it requires a list of indexes to make sure that the cards are present.

Activity 8

1. Elevens, Thirteens, and Tens, all use a (partially) similar board, all have legal and illegal moves, and all games have a target end (to clear the deck). However, each game has a different target value, and varying rules on how to clear cards.
2. The board instance values are instantiated through the use of a constructor method, and values are passed into the constructor.
3. The abstract methods (isLegal & anotherPlayPossible) will not work with all games because of the differences mentioned in number 1. However all games need these methods, which is why they are abstract, with varying implementations.