Activity 2

1. A deck contains an amount of cards.
2. 6
3. Ranks = {“two”,”three”,”four”,”five”,”six”,”seven”,”eight”,”nine”,”ten”,”king”,”queen”,”jack”,”ace”}

Suits = {“clubs”,” diamonds”, “hearts”,” spades”}

pointValues = {2,3,4,5,6,7,8,9,10,10,10,10,11}

1. Yes, the index values of ranks corresponds to pointValues.

Activity 3

1. Public static Boolean flip()

{

//returning true is heads

Return Math.random() > .33;

}

1. Public static Boolean arePermutations(int[] one, int[] two)

{

Int equals = 0;

For(int i = 0;i<one.length;i++)

{

For(int j = 0; j<onelength;j++)

{

If(one[i] == one[j])

Equals++;

}}

Return equals == one.length;

} 0,1,1

Activity 5

1. isEmpty

I think they might be returning true when the size equals one instead of zero (typo).

1. getSize

They probably forgot to initialize the size variable in the constructor after adding the cards.

1. Shuffle

They are setting card at position r back to r and the card at position k back to k (typo).

1. Deal

There is probably a problem with isEmpty(size might not be initialized correctly; same problem as Buggy 1). Because the deck thinks it is empty and it returns null.

Activity 6

1. 5 spades and 6 clubs, 5 clubs and 6 clubs,
2. If all the cards have been matched and there are 3 cards left they have to matched as a group (since it is an odd number and won’t work with pairs). Jack, Queen, and King is the only group of three that is matchable.
3. No, it is all random chance.

Activity 7

1. Deck, and cards on board

Deck, array of cards, and board size

1. Deal Cards, Select Cards, Replace Cards, Deal More Cards, Restart Game
2. Yes
3. A. newGame()

B. isLegal() and anotherPlayIsPossible()

C. {0,1,3,6,7}

d.for(int val: cIndexes)

{

System.out.println(board.cardAt(val);

}

e.anotherPlayIsPossible() because it needs to examine **all** the cards on the board for possible plays

Activity 8

1. They have different possibilities for legal moves. They have different board sizes. Everything else is the same
2. It uses the super keyword to call Board’s constructor inside of Elevens Board constructor.
3. Elevens board calculates if two cards add up to eleven or three cards are Jack, King, and Queen.

Another play possible checks every set of two and three cards with isLegal. Yes other boards have their own conditions for isLegal and they have different implementations for anotherPlayPossible.

Activity 9

1. Size is not an abstract method because the method is the same for each game. It is based on the size of the array Cards[] which is what is different between games (11 or 13).
2. There is no need for abstract methods because board size does not matter when removing or selecting cards. The implementation is the same (how each game uses the method is different).
3. Yes, you could call isLegal polymorphically since any class that implements the interface must have its own implementation of that method. This would not work as well as the abstract class because each game has instance variables and methods that all games should have so they are implemented in the abstract class (Ex. Cards[]). You cannot have instance variables and non-abstract methods in interfaces.