Fin Malone Elevens Questions!!

Activity 2:

1. A deck doesn’t exist without the card class so a deck, so it is a Composition of cards.
2. 6 cards.
3. Ranks are strings of 1,2,3,4,5,6,7,8,9,10,jack,queen,king,ace.

Suits are strings of spades, hearts, diamonds, clubs.

PointValues has ints of 1,2,3,4,5,6,7,8,9,10,10,10,10,11.

1. No because the deck will be shuffled anyways.

Activity 3:

1. public static String flip(){

int value = (int)(Math.random() \* 3) + 1;

if(value == 1){

return “heads”;

}

if(value == 2){

return “heads”;

}

if(value == 1){

return “tails”;

}

}

1. public static Boolean arePermutations(int[] array1,int[] array2){

int count = 0;

for(int i = 0; i < array1.length; i++){

for(int j = 0; j < array2.length; j++){

if(array1[i] == array2[j]){

count++;

}

}

}

if(count == array1.length){

return true;

}

else{

return false;

}

}

1. The shuffler would have to assign the last ones to the first ones respectively, reversing the order.

Activity 5:

Buggy 1: isEmpty tests whether or not a deck is empty. It probably does this by checking the cards that are left in the deck. If the deck is empty but it is returning false it probably means that the cards left aren’t being recorded somewhere in the class. The number of cards in the deck needs to be updated.

Buggy 2: If the size is 0 and it should be one. There is probably no place in the constructor where it assigns the amount of cards to the size variable.

Buggy 3: The shuffle method does not work because the arrangement of the shuffled deck and the original deck are the same. The algorithm isn’t being applied correctly to the shuffle method.

Buggy 4: The deck contains no cards, which means that somewhere in the constructor the cards aren’t being assigned to the deck.

Activity 6:

1. You can play the 6 and either one of the two 5’s. Those are the only options.
2. Yes, because the number of cards is an uneven number and since the deck is dealt in two’s or threes. The only way that there could be three more cards would be if they were face cards.
3. It doesn’t involve strategy. It won’t matter what play you choose because no matter what, a random set of cards will be selected to place them.

Activity 7:

1. deck, all the cards and their point values, suits, and ranks, and the area to put the cards.
2. Shuffle the deck. Deal the cards in the nine spots. If there is a move that adds up to eleven (two number cards or the three face cards) make the move. If there are more than one, choose one and make the move. Replace those cards with the cards on the top of the deck. Repeat until the game is lost or won.
3. Yes the code has all that is necessary to play the game, aside from the unimplemented code.
4. In the constructor and the newGame() method.
5. These methods should be called in the isLegal() method and the anotherPlayIsPossible() method.
6. 0: J hearts, 1:6 clubs, 2: 2 spades, 3: A hearts, 4: 4 hearts
7. For(int i = 0; i < cards.length; i ++){

If(cards[i] != null){

System.out.println(cards[i].toString());

}

}

1. anotherPlayIsPossible() method because it doesn’t want to have the same play twice.

Activity 8:

1. The Tens and Thirteens games are very similar to the Elevens game. You must add up to ten, thirteen, or eleven in pairs. There are only minor rule changes due to the different number of cards and point values. In thirteen’s, kings are removed singly, however, in tens, kings, queens, jacks, and tens are removed in four of the same rank, making it a more difficult game. The difference in cards and goals of the game makes the methods different.
2. The ElevensBoard class extends the Board class, so you can use the “super” constructor of the Board class for both classes and their instance variables.
3. The differences between the games are covered with the abstract Board class because the methods not implemented can be implemented in the subclasses (like the elevens example in the starter code).

Activity 9:

1. The size of the board is a private instance variable in the subclass of Board that is entered as a parameter in the constructor.
2. Those methods can be used in all three games without effecting the outcome of the program to the point where an error occurs.
3. In an interface I believe that you cannot implement code, therefore the code for all the methods used would be implemented in the ElevensBoard class, which would change the design work and class design, and it would not be polymorphic.