Activity 2

1. Composition; a deck “owns” cards and cannot exist independently without them
2. 6
3. String[] ranks = {“two”, “three”, “four”, “five”, “six”, “seven”, “eight”, “nine”, “ten”, “jack”, “queen”, “king”, “ace};

String[] suits = {“spades”, “hearts”, “diamonds”, “clubs”};

String[] values = {2, 3, 4, 5, 6, 7, 8, 9, 10, 10, 10, 10, 11};

1. The order of the elements in ranks must match the order of elements in pointValues, but other than that it can be in any order

Activity 3

1. import java.util.Random;

public static String flip()

{

int[] side = {1, 1, 2}

random rand = new Random();

for (int i = 0; i < 3; i++)

{

int j = rand.nextInt(3);

if (side[j] == 1)

{

return “heads”;

}

else

{

return “tails”;

}

}

}

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

{

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

{

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

{

if (array1[i] != array2[j];

{

return false;

}

}

}

return true;

}

1. 1, 3

Activity 5

1. Method: public Boolean isEmpty()

Possible Code Error: method could return if size!= 0 instead of size == 0, which would return false if the deck was not empty

1. Method: public void shuffle() / public int size()

Possible Code Error: method could have forgotten to reset size of deck to represent entire new deck after shuffling, so that public int size() returns wrong size

1. Method: public Boolean isEmpty()

Possible Code Error:

1. Method: public Boolean isEmpty()

Possible Code Error:

Activity 6

1. 5 spades, 6 clubs

5 clubs, 6 clubs

1. Yes, because every card numbered ace – 10 can be paired and removed from the board, so if there is an odd number of cards left it cannot be any of those cards. The rest of the deck can only be removed in trios of jacks, queens, and kings, so if there are three cards left it can only be one jack, one queen, and one king.
2. There’s no strategy, because the cards that replace the ones you remove are random, so you cannot know which card pair you choose is best.

Activity 7

1. Board, undealt deck, cards
2. While number of cards in undealt deck < 0

If board doesn’t have 9 cards

Deal necessary number of cards from undealt deck to get 9 cards on board

Check if any cards can be removed

If no, print “You Lose”, end game

If board has 9 cards

If sum of point values of cards == 9

Remove cards from boards, adjust size of undealt deck

Else if there is one jack, one king, one queen

Remove cards from boards, adjust size of undealt deck

1. ElevensBoard needs to depend on cards and deck
   1. It is called at the beginning of the game to deal cards to the board to start the game
   2. Public boolean isLegal(List<Integer> selectedCards) and public boolean anotherPlayIsPossible()
   3. 0 1 2 3 4 5 6 7 8

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 3 | 6 | 7 | Null | Null | Null | null |

1. public static printCards(ElevensBoard board) {

List<Integer> cIndexes = board.cardIndexes();