Matt Wright

Period 5

Elevens Lab Questions

1) No Questions

2) A) A deck contains one or more cards, but cards can exist without a deck

B) 6

C) String[] suits = new String[4];

Suits[0] = “Spades”; suits[1] = “Hearts”; suits[2] = “Diamonds”; suits[3] = “Clubs”;

String[] ranks = new String[13];

ranks[0] = “One”; ranks [1] = “Two”; ranks [2] = “Three”; ranks [3] = “Four”;

ranks[4] = “Five”; ranks [5] = “Six”; ranks [6] = “Seven”; ranks [7] = “Eight”;

ranks[0] = “Nine”; ranks [8] = “Jack”; ranks [9] = “Queen”; ranks [10] = “King”;

ranks[12] = “Ace”;

int[] pointValues = new int[10];

for(int i = 0; i<13; i++){pointValues[i] = i;}

D)Yes, they’ll get assigned values respectively

3) A) public static String flip(){

Random generator = new Random();

int determineFlip = generator.nextInt(3);

//Tails if zero, heads if one or two

if(determineFlip == 0){return “Tails”;}

else(){return “Heads”;}

}

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

boolean[] answer = new boolean[array1.length];

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

for(int num; array2){

if(array1[i] == num){

Boolean[i] = true;

Break;

}

}

if (boolean[i] = false){return false}

}

return true;

}

C) It’ll start at the rear, generate progressive random numbers (1, 2, 3, 4) and swap them as the loop iterates backwards through half of the array.

5) Buggy1: Constructor or Method (write method name): deal()

Describe a Possible Code Error: Not decrementing the size after successfully dealing a card.

Buggy2: Constructor or Method (write method name): testOneCard()

Describe a Possible Code Error: Bad boolean operator comparison

Buggy3: Constructor or Method (write method name): shuffle()

Describe a Possible Code Error: Does not iterate through shuffled deck to assign to original deck

Buggy4: Constructor or Method (write method name): Constructor

Describe a Possible Code Error: Bounds error when setting size

6) A) 5♠ 4♥ 2♦

5♠ 6♣

4♥ 6♣

4♥ 2♦ 5♣

2♦ 6♣ 2♠ A♠

6♣ 5♣

A♠ J♥

A♠ K♦

B) False, you can have an empty deck, three cards on the board, and lose that round. If you were to win, it would have to be true, because you cannot have three number cards add up to eleven

C) If you know the values of the cards being dealt, yes. That is, if you carefully think about the cards in the deck and ones that have already been played, you can make choices accordingly, such as opting not to take that Ace and King when you know there will be a Queen and Jack to come.

7) A) Deck, ArrayList<Card>, int[] selected

B) create int total

Have two for loops iterate through the deck of cards, starting from the beginning, then starting from 1 when the inner loop finishes. Stop when total == 11 or total exceeds 11 or when total does not reach 11 by the end of the loop

C) Yes

D) 1) contructor

newGame()

2) anotherPlayIsPossible()

isLegal()

3) 0 1 3 6 7

4) String cards = "";

for (int k = 0; k < cIndexes.size(); k++) {cards += k + ": " + cards[k] + "\n"; }

return cards;

5) isLegal() because it checks whether or not it contains a sum of 11 or a JKQ, which require the indexes of the cards

8) A) Deal(), has a card, has a deck

The complete implementation of how the game is one is different. So checking for it will also be different

B) In the ElevensBoard constructor, pass in the values to the super constructor. Have a method that returns these values.

C) Yes because each implementation covers what that specific board needs to be covered.

9) A) The private instance variable cards is the length of the number of cards on the table. Each subclass passes in a different length of Cards, therefore, no need to be abstract.

B) Each board handles dealing the cards and removing them in their own special way according to their rules.

C) Yes because the GUI need not know how it is implemented, it just needs it to be guaranteed to be implemented, as guaranteed by the implementation of the interface. No because it strips away a lot of the code reuse that would be gained from the abstract board class.