1. Explain in your own words the relationship between a deck and a card.

A deck is an object that stores numerous card objects.

2. Consider the deck initialized with the statements below. How many cards does the deck contain?

String[] ranks = {"jack", "queen", "king"};

String[] suits = {"blue", "red"};

int[] pointValues = {11, 12, 13};

Deck d = new Deck(ranks, suits, pointValues);

6 cards.

3. The game of Twenty-One is played with a deck of 52 cards. Ranks run from ace (highest) down to 2 (lowest). Suits are spades, hearts, diamonds, and clubs as in many other games. A face card has point value 10; an ace has point value 11; point values for 2, ..., 10 are 2, ..., 10, respectively. Specify the contents of the ranks, suits, and pointValues arrays so that the statement Deck d = new Deck(ranks, suits, pointValues); initializes a deck for a Twenty-One game.

String[] ranks = {“2”, “3”, “4”, “ 5”, “6”, “7”, “8”, “9”, “10”, “Jack”, “Queen”, “King“, “Ace”};

String [] suits = {“Spades”, “Hearts”, “Clubs”, “Diamonds”};

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

4. Does the order of elements of the ranks, suits, and pointValues arrays matter?

The order of ranks and point values need to be in the same order so the point values

match the rank, but the order of the others do not matter.

1. Write a static method named flip that simulates a flip of a weighted coin by returning either "heads" or "tails" each time it is called. The coin is twice as likely to turn up heads as tails. Thus, flip should return "heads" about twice as often as it returns "tails."

public String void flip(){

int chooser = (int)((Math.random() \* 2) + 1);

if(chooser > 2) return “Heads”;

return “Tails”;

}

2. Write a static method named arePermutations that, given two int arrays of the same length but with no duplicate elements, returns true if one array is a permutation of the other (i.e., the arrays differ only in how their contents are arranged). Otherwise, it should return false.

public static boolean arePermutations(int[] a, int[] b){  
 for(int i = 0; i < a.length; i++){  
 boolean test1 = false;  
 for(int k = 0; k < b.length; k++){  
 if(a[i] == b[k]){  
 test1 = true;  
 }  
 }  
 if(test1){  
 return true;  
 }  
 }  
 return false;  
}

3. Suppose that the initial contents of the values array in Shuffler.java are {1, 2, 3, 4}. For what sequence of random integers would the efficient selection shuffle change values to contain {4, 3, 2, 1}?

1, 2, 2, 2