Elevens Lab

Student Answer Sheet - Activity 2

**Questions:**

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

1. Consider the deck initialized with the statements shown in the *Elevens Student Lab Guide*. How many cards does the deck contain?
2. 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 aces 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.

1. Does the order of the **ranks**, **suits**, and **pointValues** arrays matter?

Elevens Lab

Student Answer Sheet - Activity 3

**Questions:**

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**”.
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**.
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}**?

Elevens Lab

Student Answer Sheet - Activity 5

**Record your conclusions below:**

**Buggy 1:**

Constructor or Method (write method name):

Describe a Possible Code Error:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Buggy 2:**

Constructor or Method (write method name):

Describe a Possible Code Error:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Buggy 3:**

Constructor or Method (write method name):

Describe a Possible Code Error:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Buggy 4:**

Constructor or Method (write method name):

Describe a Possible Code Error:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Elevens Lab

Student Answer Sheet - Activity 6

**Questions:**

1. List all possible plays for the board 5♠ 4♥ 2♦ 6♣ A♣ J♥ K♦ 5♣ 2♠
2. If the deck is empty and the board has three chards left, must they be a **J**, **Q**, AND **K**? Why or why not?
3. Does the game involve any strategy? That is, when more than one play is possible, does it matter which one is chosen? Briefly explain your answer.

Elevens Lab

Student Answer Sheet - Activity 7

**Questions:**

1. What items would be necessary if you were playing a game of Elevens at your desk (not on the computer)? List the private instance variables needed for the **ElevensBoard** class.
2. Write an algorithm (pseudocode) that describes the actions necessary to play the Elevens game.
3. Now examine the partially implemented **ElevensBoard.java** file found in the ***Activity Starter Code*** directory. Does the **ElevensBoard** class contain all the state and behavior necessary to play the game?
4. **ElevensBoard.java** contains three helper methods. These helper methods are **private** because they are only called from the **ElevensBoard** class.   
   1. Where is the **dealMyCards** method called in **ElevensBoard**?
   2. Which **public** methods should call the **containsPairSum11** and **containsJQK** methods?
   3. It’s important to understand how the **cardIndexes** method works, and how the list that it returns is used. Suppose that cards contains the elements shown in in question 4 part c of the Elevens Student Lab Guide. Trace the execution of the **cardIndexes** method to determine what list will be returned. Complete the diagram below by filling in the elements of the returned list, and by showing how those values index cards. Note that the returned list may have less than 9 elements.

returned list

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|  |  |  |  |  |  |  |  |  |

* 1. Complete the following **printCards** method to print all of the elements of cards that are indexed by **cIndexes**.

**public static printCards(ElevensBoard board) {**

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

**/\* your code goes here \*/  
  
  
  
  
  
  
  
}**

* 1. Which one of the methods that you identified in question 4b above needs to call the **cardIndexes** method before calling the **containsPairSum11** and **containsJQK** methods? Why?

Elevens Lab

Student Answer Sheet - Activity 8

**Questions:**

1. Discuss the similarities and differences between *Elevens, Thirteens,* and *Tens*,
2. As discussed previously, all of the instance variables are declared in the **Board** class. But it is the **ElevensBoard** class that “knows” the board size, and the ranks, suits, and point values of the cards in the deck. How do the **Board** instance variables get initialized with the **ElevensBoard** values? What is the exact mechanism?

1. Now examine the files **Board.java**, and **ElevensBoard.java**, found in the **Activity8 Starter Code** directory. Identify the **abstract** methods in **Board.java**. See how these methods are implemented in **ElevensBoard**. Do they cover all the differences between ***Elevens***, ***Thirteens***, and ***Tens*** as discussed in question 1? Why or why not?

Elevens Lab

Student Answer Sheet - Activity 9

**Questions:**

1. The size of the board is one of the differences between ***Elevens*** and ***Thirteens***. Why is **size** not an abstract method?
2. Why are there no abstract methods dealing with the selection of the cards to be removed or replaced in the array **cards**?
3. Another way to create “IS-A” relationships is by implementing interfaces. Suppose that instead of creating an **abstract Board** class, we created the following **Board** interface, and had **ElevensBoard** implement it. Would this new scheme allow the Elevens GUI to call **isLegal** and **anotherPlayIsPossible** polymorphically? Would this alternate design work as well as the **abstract Board** class design? Why or why not?

**public interface Board {**

**boolean isLegal(List<Integer> selectedCards);**

**boolean anotherPlayIsPossible();**

**}**

Elevens Lab

Student Answer Sheet - Activity 11

**Question:**

1. Set the **I\_AM\_DEBUGGING** flags to **false** and **GAMES\_TO\_PLAY** to **10**. Run the **ElevensSimulation** program a few times and record the percentage of games won for each run. What is the range of win percentages that you saw? Were the percentages fairly consistent, or did they vary quite a bit?
2. Increase the number of games to play to 100. Are the win percentages more consistent from run to run?

1. Experiment with simulating different numbers of games. How many games do you need to play in order to get consistent results from run to run?

Optional: Repeat the above steps for the Thirteens game.