A1-2:

1. A deck is the group of cards, while cards are the individual items within a deck.
2. 6
3. String[] suits = {“Hearts”, “Spades”, “Clubs”, “Diamonds”};

int[] values = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13};

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

1. No

A3-4:

1. public static String flip()  
    {  
    Random rnd = new Random();  
    String side;  
    int n = rnd.nextInt(3);  
    if(n==0 | n==1)  
    {  
    side = "heads";  
    }  
    else  
    {  
    side = "tails";  
    }  
    return side;  
   }
2. public static void arePermutations(int[]a, int[] b)  
    {  
    boolean[] perm = new boolean[a.length];  
    if(a.length == b.length)  
    {  
    for(int i = 0; i < a.length; i++)  
    {  
    perm[i] = false;  
    for(int j = 0; j < b.length; j++)  
    {  
    if(a[i] == b[j])  
    {  
    perm[i] = true;  
    }  
    }  
    }  
    boolean isPerm = true;  
    for(int i=0; i < perm.length;i++)  
    {  
    if(perm[i] != true)  
    {  
    isPerm = false;  
    break;  
    }  
    }  
    if(isPerm == true)  
    {  
    System.*out*.println("They are permutations.");  
    }  
    else  
    {  
    System.*out*.println("There are not permutations.");  
    }  
    }  
    }
3. The Shuffle method is completely random, so you cannot tell when an array of {1,2,3,4} will reverse into {4,3,2,1}.

A6:

1. 5 and 6, Ace and Jack, King and Ace.
2. No because the player removes any triplet consisting of a Jack, Queen, or King.
3. Yes, it does involve strategy because when multiple plays are available you can play your cards in a right way to maximize pairs.

A7:

1. A deck of cards is needed. BOARD\_SIZE, RANKS, SUITS, POINT\_VALUES are the variables.
2. First there would be a while loop running when an int value i not being equal to 9. This loop would necessitate the use of rand.NextInt(cards.size()) to generate a card to deal. This card would then be removed from the ArrayList, while, i would increase by 1 (i++) and cards.size() would decrease by 1 due to the removal of a card. Then an if statement would determine if the player's selected cards equaled eleven, if they did then the if-then would break going back to the while loop. The same thing would happen if the player selected a J, Q, & K. All of this repeating until the deck of cards is emptied.
3. The ElevensBoard.java starter code has all the prerequisites except for the actual code within certain methods
4. A. dealMyCards() is called in the ElevensBoard constructor as well as the newGame() method

B. isLegal(List<Integer> selectedCards) and anotherPlayIsPossible() should call the containsPairSum11() and containsJQK() methods

C. The returned list would be: {J,6,2,A,4} or {0,1,3,6,7}

D. public static void printCards(ElevensBoard board) { List<Integer> cIndexes = board.cardIndexes(); for(int i = 0; i < cIndexes.size(); i++) { System.out.println(board.cards[i]); } }

E. anotherPlayIsPossible() needs to call the containsPairSum11() and containsJQK() methods to determine whether another play is possible

A8:

1. Elevens, Thirteens, and Tens are all played with the goal of reaching a certain point value. The difference is the amount that you are reaching within the game.
2. The ElevensBoard creates a ElevensBoard object which is modelled after a Board object and gives that object the values to play Elevens.
3. 3. both isLegal and anotherPlayIsPossible are abstract methods meaning these methods cover the differences between all three games, so if another class was written for Thirteens or Blackjack then it could implement these methods.

A9:

1. Size varies between games and thus wouldn't be abstract since it's a variable.
2. The Board interface already has the abstract methods, for selecting the cards to be removed or replaced, to implement.
3. If Board was an interface it would lose its constructor and Elevens wouldn’t be able to run.