```java
import java.util.ArrayList;
import java.util.Random;
import java.util.concurrent.ThreadLocalRandom;

/**
 * Simulates a deck of 52 playing cards.
 */
public class Deck {
    private static final int NUMBER_OF_CARDS=52;
    private static final int NUMBER_OF_SUITS=4;
    private static final int CARDS_IN_SUIT=13;
    private static final int FIRSTCARDINDEXINDECK = 0
;

    private ArrayList<Card> theCards;
    private boolean shuffled;

    /**
     * Constructs a new ordered deck of playing cards
     */

    public Deck()
    {
        theCards = new ArrayList<Card>(
NUMBER_OF_CARDS);
        shuffled= false;
        addCardsToDeck();

    }


    /**
     * Add a standard 52 cards to a deck
     */
    private void addCardsToDeck() {
        for (String suit: Card.SUITS){
            for (int rank = Card.MINRANK; rank <=
Card.MAXRANK; rank++){
                theCards.add(new Card(rank, suit));
            }
        }
    }


    /**
     * Deals out next card in deck; returns null if no cards left
     *
     * @return next card in deck or null if deck empty
     */
    public Card deal() {
        Card cardToDeal;

        if (!shuffled && !isEmpty()) {
            cardToDeal = theCards.get(0);
            theCards.remove(cardToDeal);
            return cardToDeal;
        }

        else if (shuffled && !isEmpty())  {
            int randomCardIndex = ThreadLocalRandom.current().nextInt(FIRSTCARDINDEXINDECK, size());
            cardToDeal = theCards.get(randomCardIndex);
            theCards.remove(cardToDeal);
            return cardToDeal;
        }
        else { return null; }

    }

    /** determines if deck has any cards left in it
     *
     * @return true if Deck empty; else false
     */
    public boolean isEmpty() {
        return size() == 0;
    }

    /**
     * Shuffles the cards
     */
    public void shuffle()
    {
        shuffled = true;
    }

    /** Returns number of undealt cards left in the deck
     *
     * @return number of undealt cards in the deck
     */
    public int size()
    {
        return theCards.size();
    }

    /**
     * Reset the deck by gathering up all dealt cards.
     * Postcondition: Deck contains all cards and is NOT shuffled
     */
    public void gather()
    {
        theCards.clear();
        addCardsToDeck();
        shuffled = false;

    }

    /**
     *  DEBUGGING METHOD: prints out stats of the given list of cards, that is, what was dealt.
     *  Prints the remaining number of cards of each suit and of each rank.
     *
     *  @param cardList list of cards that are (were) in the deck
     * @hidden
     */
    public void printStats(ArrayList<Card> cardList)
```

```java
    {
        int Hcount=0;
        int Dcount=0;
        int Scount=0;
        int Ccount=0;
        int[] ranks = new int[CARDS_IN_SUIT];
        int size=cardList.size();
        for (int i=0; i<size; i++)
        {
            int val = cardList.get(i).getRank();
            String suit = cardList.get(i).getSuit();
            if (suit.equals("clubs"))
                Ccount++;
            else if (suit.equals("diamonds"))
                Dcount++;
            else if (suit.equals("spades"))
                Scount++;
            else if (suit.equals("hearts"))
                Hcount++;
            ranks[val-2]++;  // deck RANKS run from 2-14 so need to subtract 2
        }
        System.out.println("***PRINTING DECK STATS ***");
        System.out.println("# clubs: " + Ccount);
        System.out.println("# diamonds: " + Dcount);
        System.out.println("# hearts: " + Hcount);
        System.out.println("# spades: " + Scount);

        System.out.print("Card:\t");
        for (int j = 2; j< Card.RANKS.length; j++) {
        System.out.print(Card.RANKS[j]+"\t");
        }
        System.out.println();
        System.out.print("Qty:\t");
        for (int j=0; j<ranks.length; j++) {
        System.out.print(ranks[j] + "\t");
        if (j>8) {  // indices 9-12 are Jack thru Ace
            System.out.print("\t");
            }
        }
        System.out.println("\n");
    }

    /**
     *  DEBUGGING METHOD: prints out stats of this Deck object
     *  Prints the remaining number of cards of each suit and of each rank.
     *
     * @hidden
     */
    public void printStats() {
        printStats(theCards);
    }


}

```