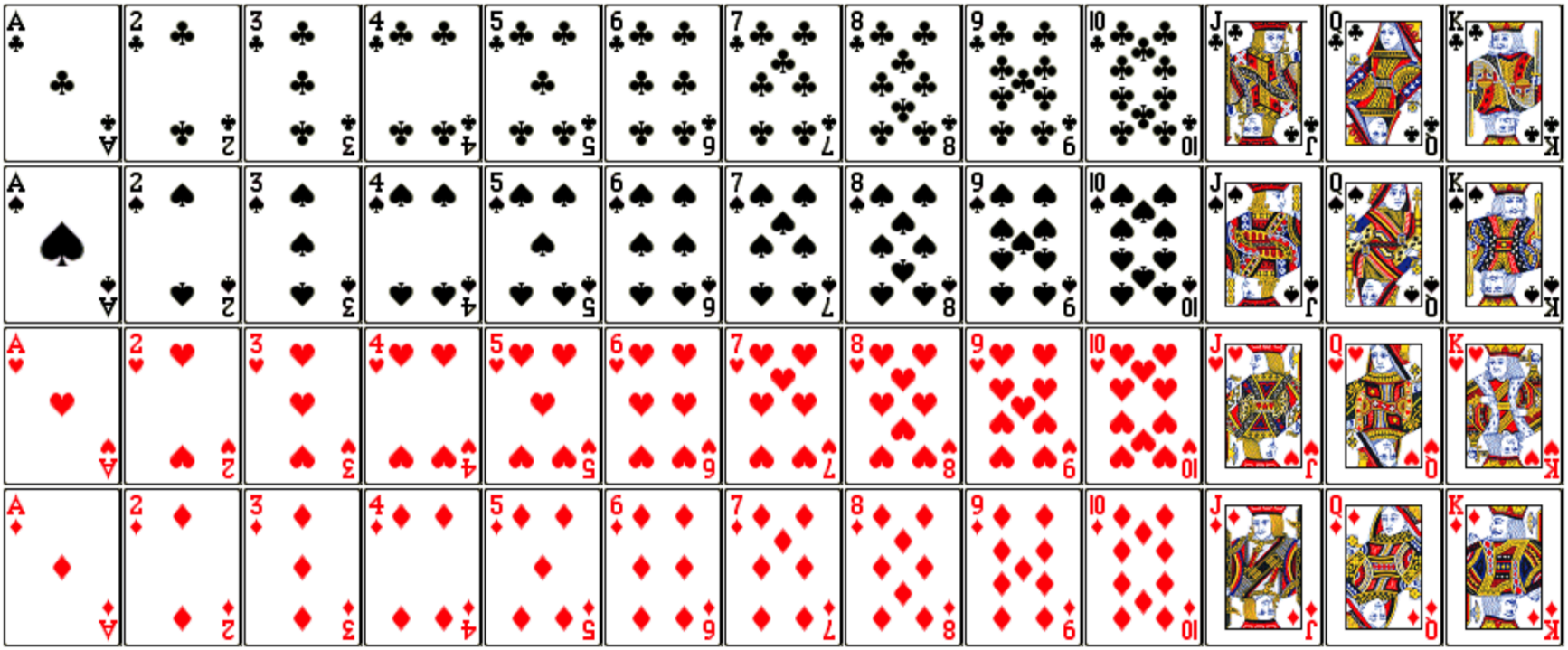
CS 149

Professor: Kevin Molloy
(slides adapted from Alvin Chao)

Model 2 Playing Cards

- Now let's play a different game. In this section, we'll create an array of face cards. (Note: The array is one-dimensional, but the cards are displayed on four lines for convenience.)



Card Questions

1. Implement the following constructor. The class has two attributes: rank and suit. Don't think too hard about it.

```
/**
 * Constructs a face card given its rank and suit.
 * @param rank face value (1 = ace, 11 = jack, 12 = queen, 13 = king)
 * @param suit category ("clubs", "diamonds", "hearts", or "spades")
*/
public Card(int rank, String suit) { }
```

2. In one line of code,declare an array of strings named suits and initialize its contents to the four possible suits as shown above.

Card Questions Contd.

3. Write several lines of code to declare and create an array of 52 Card objects. Use nested for loops to construct Card objects in the order of the Model. Make use of your suits array from the previous question. How you will keep track of the array index?

Card Questions Contd.

4. Describe what the following code does and how it works. (Note: You've come along way this semester, to be able to understand this example!)

```
public static Card[] sort(Card[] deck) {
        if (deck == null) {
           System.err.println("Missing deck!");
           return null;
        }
        Card[] sorted = new Card[deck.length];
        for (Card card : deck) {
           int index = card.position();
           sorted[index] = card;
        }
        return sorted;
}
```

13. Write a static method named inDeck that takes a Card[] representing a

Card Questions Contd.

5. Write a static method named inDeck that takes a Card[] representing a deck of cards and a Card object representing a single card, and that returns true if the card is somewhere in the deck of cards. Be sure to use the equals method of the Card object to make comparisons.

- **Acknowledgements**

Parts of this activity are based on materials developed by Chris Mayfield and Nathan Sprague.

</end>