```
import java.util.*;

// Enum for card suits
enum Suit { SPADE, CLUB, HEART, DIAMOND }

// Enum for card ranks
enum Rank { ACE, TWO, THREE, FOUR, FIVE, SIX, SEVEN, EIGHT, NINE, TEN, JACK, QUEEN, KING
}

// Card class representing a single card
class Card {
    private Suit suit;
    private Rank rank;

    public Card(Suit suit, Rank rank) {
        this.suit = suit;
        this.rank = rank;
    }

    public Suit getSuit() {
        return suit;
    }

    public Rank getRank() {
        return rank;
    }

    @Override
    public String toString() {
        return rank + " of " + suit;
    }
}

// Custom comparator for comparing cards based on color, suit, and rank
class CardComparator implements Comparator<Card>
{
    // Helper method to determine color value
    private int getColorOrder(Suit suit )
    {
        return (suit == Suit.HEART || suit == Suit.DIAMOND) ? 0 : 1 ; // use ternary operator for comparison
    }
    @Override
    public int compare(Card card1, Card card2)
    {
        int colorComparison = getColorOrder(card1.getSuit()) - getColorOrder(card2.getSuit());
        if (colorComparison != 0)
         return colorComparison;

        int suitComparison = card1.getSuit().compareTo(card2.getSuit());
        if (suitComparison != 0)
        return suitComparison;

        return card1.getRank().compareTo(card2.getRank());
    }
}
```

```
// Deck class to manage the deck of cards
class Deck {
    private List<Card> cards;

    public Deck() {
        cards = new ArrayList<>();
        // Initialize deck with 52 cards
        for (Suit suit : Suit.values()) {
            for (Rank rank : Rank.values()) {
                cards.add(new Card(suit, rank));
            }
        }
    }

    // Method to draw n random cards from the deck
    public List<Card> drawRandomCards(int n) {
        if (n > cards.size()) {
            throw new IllegalArgumentException("Not enough cards in the deck");
        }
        Collections.shuffle(cards); // Shuffle the deck
        List<Card> drawnCards = new ArrayList<>();
        for (int i = 0; i < n; i++) {
            drawnCards.add(cards.remove(0));
        }
        return drawnCards;
    }
}

public class card_suffel_game {
    public static void main(String[] args) {
        Deck deck = new Deck();
        List<Card> drawnCards = deck.drawRandomCards(20);

        // Print the drawn cards before sorting
        System.out.println("Drawn cards before sorting:**********************************\n");
        for (Card card : drawnCards) {
            System.out.println(card);
        }

        // Sort the drawn cards using custom comparator
        Collections.sort(drawnCards, new CardComparator());

        // Print the drawn cards after sorting
        System.out.println("\nDrawn cards after sorting:**********************************\n");
        for (Card card : drawnCards) {
            System.out.println(card);
        }
    }
}
```