📝

Assignment-4

CODE:

```
//Name : Aniket Singh
//PRN: 21070126013

// Problem: Write a menu-driven Java Program for the following: There are 52 cards in a deck, each of
which belongs to one of four suits and one of 13 ranks. Represent a deck of cards as an array of Obje
cts

import java.util.*;

public class Assignment_4_vector {
    public static void main(String[] args) {
        Deck deck = new Deck();
        deck.createDeck();

    }
}

class Card {
    public static final String[] suits = { "Hearts", "Diamonds", "Clubs", "Spades" };
    public static final String[] ranks = { "Ace", "2", "3", "4", "5", "6", "7", "8", "9", "10", "Jac
k", "Queen",
            "King" };

    private int rank;
    private String suit;

    public Card(int rank, String suit) {
        this.rank = rank;
        this.suit = suit;
    }

    public int getRank() {
        return rank;
    }

    public String getSuit() {
        return suit;
    }

    public String toString() {
        return ranks[rank - 1] + " of " + suit;
    }
}

// deck class
class Deck {
    public void createDeck() {
        Scanner input = new Scanner(System.in);
        Vector<Card> deck = new Vector<Card>(52);

        // populate the deck with cards
        for (int rank = 1; rank <= 13; rank++) {
            for (String suit : Card.suits) {
                Card card = new Card(rank, suit);
                deck.add(card);
            }
        }

        // display the menu
        while (true) {
            System.out.println("***************************\n");
            System.out.println("1. Display the deck of cards");
            System.out.println("2. Shuffle the deck of cards");
            System.out.println("3. Draw a card from the deck");
            System.out.println("4. Empty the deck");
            System.out.println("5. Print a card from the deck");
            System.out.println("6. Compare two cards");
            System.out.println("7. Check if two cards are same");
            System.out.println("8. Find card by rank and suit");
            System.out.println("9. Deal a hand of cards");
            System.out.println("10. Quit");
            // get user choice
            System.out.print("Enter your choice (1-10): ");
            int choice = input.nextInt();

            System.out.println("***************************\n");

            // handle user choice
            switch (choice) {
                case 1:
                    displayDeck(deck);
                    break;
                case 2:
                    shuffleDeck(deck);
                    break;
                case 3:
                    drawCard(deck);
                    break;
                case 4:
                    emptyDeck(deck);
                    break;
                case 5:
                    printCard(deck);
                    break;
                case 6:
                    compareCard(deck);
                    break;
                case 7:
                    sameCard(deck);
                    break;
                case 8:
                    findCard(deck);
                    break;
                case 9:
                    dealCard(deck);
                    break;
                case 10 :
                    System.out.println("Goodbye!");
                    System.exit(0);
                default:
                    System.out.println("Invalid choice. Please try again.");
                    break;
            }
        }
    }

    // display the current state of the deck
    public static void displayDeck(Vector<Card> deck) {
        System.out.println("Deck of Cards:");
        for (Card card : deck) {
            System.out.println(card);
        }
        System.out.println();
    }

    // shuffle the deck
    public static void shuffleDeck(Vector<Card> deck) {
        Collections.shuffle(deck);
        System.out.println("Deck shuffled.");
    }

    // draw a card from the deck
    public static void drawCard(Vector<Card> deck) {
        if (deck.isEmpty()) {
            System.out.println("Deck is empty.");
        } else {
            Card card = deck.remove(0);
            System.out.println("You drew: " + card);
        }
    }

    // empty the deck
    public static void emptyDeck(Vector<Card> deck) {
        deck.clear();
        System.out.println("Deck emptied.");
    }

    // printCard() function take the input position in the deck and print the card
    public static void printCard(Vector<Card> deck) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the position of the card you want to draw: ");
        int position = input.nextInt();

        if (deck.isEmpty()) {
            System.out.println("Deck is empty.");
        } else {
            Card card = deck.get(position);
            System.out.println("You drew: " + card);
        }
    }

    // sameCard() draws 2 random cards and compare their ranks to check if they are
    // same or not
    public static void sameCard(Vector<Card> deck) {
        Random rand = new Random();
        int firstCard = rand.nextInt(52);
        int secondCard = rand.nextInt(52);
        if (deck.isEmpty()) {
            System.out.println("Deck is empty.");
        } else {
            Card card1 = deck.get(firstCard);
            Card card2 = deck.get(secondCard);
            if (card1.getRank() == card2.getRank()) {
                System.out.println("You drew: " + card1 + " and " + card2 + " and they are ranked sam
e.");
            } else {
                System.out.println("You drew: " + card1 + " and " + card2 + " and they are not ranked
same.");
            }
        }
    }

    // compareCard() draws 2 random cards and compare them to get the card of higher
    // rank and if ranks are same then compare their suits.
    public static void compareCard(Vector<Card> deck) {
        Random rand = new Random();
        int firstCard = rand.nextInt(52);
        int secondCard = rand.nextInt(52);

        if (deck.isEmpty()) {
            System.out.println("Deck is empty.");
        }
        else
        {
            Card card1 = deck.get(firstCard);
            Card card2 = deck.get(secondCard);
            if (card1.getRank() > card2.getRank()) {
                System.out.println("You drew: " + card1 + " and " + card2 + " and " + card1 + " is of
higher rank.");
            } else if (card1.getRank() < card2.getRank()) {
                System.out.println("You drew: " + card1 + " and " + card2 + " and " + card2 + " is of
higher rank.");
            } else {
                if (card1.getSuit().equals("Hearts")) {
                    System.out
                            .println("You drew: " + card1 + " and " + card2 + " and " + card1 + " is
 of higher rank.");
                } else if (card2.getSuit().equals("Hearts")) {
                    System.out
                            .println("You drew: " + card1 + " and " + card2 + " and " + card2 + " is
 of higher rank.");
                } else if (card1.getSuit().equals("Diamonds")) {
                    System.out
                            .println("You drew: " + card1 + " and " + card2 + " and " + card1 + " is
 of higher rank.");
                } else if (card2.getSuit().equals("Diamonds")) {
                    System.out
                            .println("You drew: " + card1 + " and " + card2 + " and " + card2 + " is
 of higher rank.");
                } else if (card1.getSuit().equals("Clubs")) {
                    System.out
                            .println("You drew: " + card1 + " and " + card2 + " and " + card1 + " is
 of higher rank.");
                } else if (card2.getSuit().equals("Clubs")) {
                    System.out
                            .println("You drew: " + card1 + " and " + card2 + " and " + card2 + " is
 of higher rank.");
                }
            }
        }
    }

    // sortCard() function sorts the deck of cards in ascending order of rank and if
    // ranks are same then sort them in ascending order of suits.
    public static void sortDeck(Vector<Card> deck) {
        Collections.sort(deck, new Comparator<Card>() {
            @Override
            public int compare(Card card1, Card card2) {
                if (card1.getRank() == card2.getRank()) {
                    return card1.getSuit().compareTo(card2.getSuit());
                } else {
                    return card1.getRank() - card2.getRank();
                }
            }
        });
        System.out.println("Deck of Cards:");
        for (Card card : deck) {
            System.out.println(card);
        }
        System.out.println();
    }

    // findCard() function takes the input rank and suit and search the deck of cards to find the car
d with the given rank and suit. returns position of the card in the deck.
    public static void findCard(Vector<Card> deck) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the rank of the card you want to find: ");
        int rank = input.nextInt();
        System.out.print("Enter the suit (\"Hearts\", \"Diamonds\", \"Clubs\", \"Spades\") of the car
d you want to find: ");
        String suit = input.next();

        if (deck.isEmpty()) {
            System.out.println("Deck is empty.");
        } else {
            for (int i = 0; i < deck.size(); i++) {
                Card card = deck.get(i);
                if (card.getRank() == rank && card.getSuit().equals(suit)) {
                    System.out.println("Card found at position " + i + " in the deck.");
                    break;
                }
            }
        }
    }

    // dealCard() function takes the input number of players and deal the cards to the players.
    public static void dealCard(Vector<Card> deck) {
        shuffleDeck(deck);
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the number of players: ");
        int players = input.nextInt();

        if (deck.isEmpty()) {
            System.out.println("Deck is empty.");
        } else {
            int cardsPerPlayer = deck.size() / players;
            int remainingCards = deck.size() % players;
            int start = 0;
            int end = cardsPerPlayer;
            for (int i = 0; i < players; i++) {
                System.out.println("\nPlayer " + (i + 1) + " cards:");
                for (int j = start; j < end; j++) {
                    System.out.println(deck.get(j));
                }
                start = end;
                end += cardsPerPlayer;
            }
            if (remainingCards > 0) {
                System.out.println("Remaining cards:");
                for (int i = end; i < deck.size(); i++) {
                    System.out.println(deck.get(i));
                }
            }
        }
    }
}
```

OUTPUT:

```
PS C:\Users\ANIKET\Desktop\CODE\Java> cd "c:\Users\ANIKET\Desktop\CODE\Java\" ; if ($?) { javac Assig
nment_4_vector.java } ; if ($?) { java Assignment_4_vector }
***************************

1. Display the deck of cards
2. Shuffle the deck of cards
3. Draw a card from the deck
4. Empty the deck
5. Print a card from the deck
6. Compare two cards
7. Check if two cards are same
8. Find card by rank and suit
9. Deal a hand of cards
10. Quit
Enter your choice (1-10): 1
***************************

Deck of Cards:
Ace of Hearts
Ace of Diamonds
Ace of Clubs
Ace of Spades
2 of Hearts
2 of Diamonds
2 of Clubs
2 of Spades
3 of Hearts
3 of Diamonds
3 of Clubs
3 of Spades
4 of Hearts
4 of Diamonds
4 of Clubs
4 of Spades
```

```
5 of Hearts
5 of Diamonds
5 of Clubs
5 of Spades
6 of Hearts
6 of Diamonds
6 of Clubs
6 of Spades
7 of Hearts
7 of Diamonds
7 of Clubs
7 of Spades
8 of Hearts
8 of Diamonds
8 of Clubs
8 of Spades
9 of Hearts
9 of Diamonds
9 of Clubs
9 of Spades
10 of Hearts
10 of Diamonds
10 of Clubs
10 of Spades
Jack of Hearts
Jack of Diamonds
Jack of Clubs
Jack of Spades
Queen of Hearts
Queen of Diamonds
Queen of Clubs
Queen of Spades
King of Hearts
King of Diamonds
King of Clubs
King of Spades

***************************

1. Display the deck of cards
2. Shuffle the deck of cards
3. Draw a card from the deck
4. Empty the deck
5. Print a card from the deck
6. Compare two cards
7. Check if two cards are same
8. Find card by rank and suit
9. Deal a hand of cards
10. Quit
Enter your choice (1-10): 2
***************************

Deck shuffled.
***************************

1. Display the deck of cards
2. Shuffle the deck of cards
3. Draw a card from the deck
4. Empty the deck
5. Print a card from the deck
```

```
6. Compare two cards
7. Check if two cards are same
8. Find card by rank and suit
9. Deal a hand of cards
10. Quit
Enter your choice (1-10): 3
***************************

You drew: 4 of Hearts
***************************

1. Display the deck of cards
2. Shuffle the deck of cards
3. Draw a card from the deck
4. Empty the deck
5. Print a card from the deck
6. Compare two cards
7. Check if two cards are same
8. Find card by rank and suit
9. Deal a hand of cards
10. Quit
Enter your choice (1-10): 5
***************************

Enter the position of the card you want to draw: 6
You drew: 2 of Spades
***************************

1. Display the deck of cards
2. Shuffle the deck of cards
3. Draw a card from the deck
4. Empty the deck
5. Print a card from the deck
6. Compare two cards
7. Check if two cards are same
8. Find card by rank and suit
9. Deal a hand of cards
10. Quit
Enter your choice (1-10): 7
***************************

You drew: Queen of Clubs and Ace of Diamonds and they are not ranked same.
***************************

1. Display the deck of cards
2. Shuffle the deck of cards
3. Draw a card from the deck
4. Empty the deck
5. Print a card from the deck
6. Compare two cards
7. Check if two cards are same
8. Find card by rank and suit
9. Deal a hand of cards
10. Quit
Enter your choice (1-10): 8
***************************

Enter the rank of the card you want to find: 9
Enter the suit ("Hearts", "Diamonds", "Clubs", "Spades") of the card you want to find: Hearts
Card found at position 49 in the deck.
```

```
***************************

1. Display the deck of cards
2. Shuffle the deck of cards
3. Draw a card from the deck
4. Empty the deck
5. Print a card from the deck
6. Compare two cards
7. Check if two cards are same
8. Find card by rank and suit
9. Deal a hand of cards
10. Quit
Enter your choice (1-10): 4
***************************

Deck emptied.
***************************

1. Display the deck of cards
2. Shuffle the deck of cards
3. Draw a card from the deck
4. Empty the deck
5. Print a card from the deck
6. Compare two cards
7. Check if two cards are same
8. Find card by rank and suit
9. Deal a hand of cards
10. Quit
Enter your choice (1-10): 10
***************************

Goodbye!
PS C:\Users\ANIKET\Desktop\CODE\Java>
```

java_Assignments/Assignment_4 at main · AniketSingh1m/java_Assignments

Contribute to AniketSingh1m/java_Assignments development by creating an account on GitHub.

https://github.com/AniketSingh1m/java_Assignments/tree/main/Assignment_4