JAVA ASSIGNMENT ASSIGNMENT-4
DATE:25-03-23

PROBLEM STATEMENT :

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.
Should have methods:
a) createDeck() //Can also add this method as a constructor
b) printDeck()
c) printCard()
d) sameCard() //Card which is from same suit
e) compareCard() //Card having same rank or number
f) findCard() //Search for particular card
g) dealCard() //Print 5 random cards
h) shuffleDeck() //Randomize the deck

-The program should contain different java files.
-Each operation should be a separate function.
-Must use ArrayList or Vector to create arraylist or vector of objects.
- Program should contain at top of Main file in comments : Name, PRN, Batch
- Program should follow all the coding guidelines.
- Program should contain comments for particular block of logic.
- It is recommended to upload Program on your GitHub account.
- Your Card_Deck repository on GitHub should contain a README file describing all functions or methods or definitions.

.

CODE:

```
/*
Problem Statement : 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.
                    Should have methods:
                     a) createDeck() //Can also add this  method as a constructor
                     b) printDeck()
                     c) printCard()
                     d) sameCard() //Card which is from same suit
                     e) compareCard() //Card having same rank or number
                     f) findCard() //Search for particular card
                     g) dealCard() //Print 5 random cards
```

h) shuffleDeck() //Randomize the deck

-The program should contain different java files.
-Each operation should be a separate function.
-Must use ArrayList or Vector to create arraylist or vector of objects.
- Program should contain at top of Main file in comments : Name, PRN, Batch
- Program should follow all the coding guidelines.
- Program should contain comments for particular block of logic.
- It is recommended to upload Program on your GitHub account.
- Your Card_Deck repository on GitHub should contain a README file describing all functions or methods or definitions.

Name - Nisarg Patel
PRN - 21070126060
Batch - AIML A3

LAB ASSIGNMENT - 4
*/

```
package Assignment__4;
import java.util.*;
public class Card_dealer
{
   public static void main(String[] args)
   {
      Deck deck = new Deck();
      deck.createDeck();

   }
}

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

   private int rank;                // rank of the card
   private String suit;             // suit of the card

   public Card(int rank, String suit)         // parameterized constructor
   {
      this.rank = rank;
      this.suit = suit;
   }
```

```
    // getters and setters for rank and suit
    public int getRank()
    {
        return rank;
    }

    public String getSuit()
    {
        return suit;
    }

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

// deck class to create deck of cards
class Deck
{
    // createDeck() function creates the deck of cards
    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();
            System.out.println("Card Dealer Menu");
            System.out.println("**************************\n");
            System.out.println(" 1. Create a deck of cards");
            System.out.println(" 2. Display the deck of cards");
            System.out.println(" 3. Shuffle the deck of cards");
            System.out.println(" 4. Draw a card from the deck");
            System.out.println(" 5. Sorts the deck of cards");
            System.out.println(" 6. Print a card from the deck");
```

```
System.out.println(" 7. Compare two cards");
System.out.println(" 8. Check if two cards are same");
System.out.println(" 9. Find card by rank and suit");
System.out.println("10. Deal a hand of cards");
System.out.println("11. Empty the deck");
System.out.println(" 0. Quit");
System.out.println();
// get user choice
System.out.print("Enter your choice : ");
int choice = input.nextInt();

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

// handle user choice
switch (choice)
{
   case 1:
      createDeck();
      break;
   case 2:
      displayDeck(deck);
      break;
   case 3:
      shuffleDeck(deck);
      break;
   case 4:
      drawCard(deck);
      break;
   case 5:
      sortDeck(deck);
      break;
   case 6:
      printCard(deck);
      break;
   case 7:
      compareCard(deck);
      break;
   case 8:
      sameCard(deck);
      break;
   case 9:
      findCard(deck);
      break;
   case 10:
      dealCard(deck);
      break;
   case 11:
      emptyDeck(deck);
```

```
                break;

            case 0 :
                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:");
    System.out.println();
    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();
    System.out.println("Deck shuffled.");
}

// draw a card from the deck
public static void drawCard(Vector<Card> deck)
{
    System.out.println();
    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();
        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.println();
        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);
        System.out.println();
        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
same.");
            }
            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);

        System.out.println();
        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)
   {
      System.out.println();
      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();
        System.out.println("Deck sorted.");
    }

    // findCard() function takes the input rank and suit and search the deck of cards to find the
card 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.println();
        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 card 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)
    {
        Scanner input = new Scanner(System.in);
        System.out.println();
        System.out.print("Enter the number of players: ");
        int players = input.nextInt();

        System.out.print("Enter the number of cards to be dealt to each player: ");
        int cardsPerPlayer = 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("Player " + (i + 1) + " cards:");
                System.out.println("------------");
                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 :

File - Card_dealer

```
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent
:C:\Program Files\JetBrains\IntelliJ IDEA Community
Edition 2022.3.1\lib\idea_rt.jar=65211:C:\Program Files
\JetBrains\IntelliJ IDEA Community Edition 2022.3.1\bin
" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -
Dsun.stderr.encoding=UTF-8 -classpath E:\SEM-4\out\
production\SEM-4 Assignment__4.Card_dealer

Card Dealer Menu
**************************

 1. Create a deck of cards
 2. Display the deck of cards
 3. Shuffle the deck of cards
 4. Draw a card from the deck
 5. Sorts the deck of cards
 6. Print a card from the deck
 7. Compare two cards
 8. Check if two cards are same
 9. Find card by rank and suit
10. Deal a hand of cards
11. Empty the deck
 0. Quit

Enter your choice : 1
**************************


Card Dealer Menu
**************************

 1. Create a deck of cards
 2. Display the deck of cards
 3. Shuffle the deck of cards
 4. Draw a card from the deck
 5. Sorts the deck of cards
 6. Print a card from the deck
 7. Compare two cards
 8. Check if two cards are same
 9. Find card by rank and suit
10. Deal a hand of cards
```

```
11. Empty the deck
 0. Quit

Enter your choice : 2
**************************

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


Card Dealer Menu
**************************

 1. Create a deck of cards
 2. Display the deck of cards
 3. Shuffle the deck of cards
 4. Draw a card from the deck
 5. Sorts the deck of cards
 6. Print a card from the deck
 7. Compare two cards
 8. Check if two cards are same
 9. Find card by rank and suit
10. Deal a hand of cards
11. Empty the deck
 0. Quit

Enter your choice : 3
**************************
```

```
Deck shuffled.

Card Dealer Menu
**************************

 1. Create a deck of cards
 2. Display the deck of cards
 3. Shuffle the deck of cards
 4. Draw a card from the deck
 5. Sorts the deck of cards
 6. Print a card from the deck
 7. Compare two cards
 8. Check if two cards are same
 9. Find card by rank and suit
10. Deal a hand of cards
11. Empty the deck
 0. Quit

Enter your choice : 4
**************************


You drew: Jack of Spades

Card Dealer Menu
**************************

 1. Create a deck of cards
 2. Display the deck of cards
 3. Shuffle the deck of cards
 4. Draw a card from the deck
 5. Sorts the deck of cards
 6. Print a card from the deck
 7. Compare two cards
 8. Check if two cards are same
 9. Find card by rank and suit
10. Deal a hand of cards
11. Empty the deck
 0. Quit

Enter your choice : 5
```

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



Deck sorted.

Card Dealer Menu
**************************

 1. Create a deck of cards
 2. Display the deck of cards
 3. Shuffle the deck of cards
 4. Draw a card from the deck
 5. Sorts the deck of cards
 6. Print a card from the deck
 7. Compare two cards
 8. Check if two cards are same
 9. Find card by rank and suit
10. Deal a hand of cards
11. Empty the deck
 0. Quit

Enter your choice : 6
**************************


Enter the position of the card you want to draw: 32
You drew: 9 of Clubs

Card Dealer Menu
**************************

 1. Create a deck of cards
 2. Display the deck of cards
 3. Shuffle the deck of cards
 4. Draw a card from the deck
 5. Sorts the deck of cards
 6. Print a card from the deck
 7. Compare two cards
 8. Check if two cards are same
 9. Find card by rank and suit
```

```
10. Deal a hand of cards
11. Empty the deck
 0. Quit

Enter your choice : 7
**************************


You drew: Queen of Diamonds and 4 of Diamonds and Queen
 of Diamonds is of higher rank.

Card Dealer Menu
**************************

 1. Create a deck of cards
 2. Display the deck of cards
 3. Shuffle the deck of cards
 4. Draw a card from the deck
 5. Sorts the deck of cards
 6. Print a card from the deck
 7. Compare two cards
 8. Check if two cards are same
 9. Find card by rank and suit
10. Deal a hand of cards
11. Empty the deck
 0. Quit

Enter your choice : 8
**************************


You drew: 10 of Diamonds and 6 of Diamonds and they are
 not ranked same.

Card Dealer Menu
**************************

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

```
 5. Sorts the deck of cards
 6. Print a card from the deck
 7. Compare two cards
 8. Check if two cards are same
 9. Find card by rank and suit
10. Deal a hand of cards
11. Empty the deck
 0. Quit

Enter your choice : 9
**************************


Enter the rank of the card you want to find: 34
Enter the suit ("Hearts", "Diamonds", "Clubs", "Spades
") of the card you want to find: Hearts

Card Dealer Menu
**************************

 1. Create a deck of cards
 2. Display the deck of cards
 3. Shuffle the deck of cards
 4. Draw a card from the deck
 5. Sorts the deck of cards
 6. Print a card from the deck
 7. Compare two cards
 8. Check if two cards are same
 9. Find card by rank and suit
10. Deal a hand of cards
11. Empty the deck
 0. Quit

Enter your choice : 10
**************************


Enter the number of players: 2
Enter the number of cards to be dealt to each player: 3
Player 1 cards:
------------
```

```
Ace of Clubs
Ace of Diamonds
Ace of Hearts
Player 2 cards:
------------
Ace of Spades
2 of Clubs
2 of Diamonds
Remaining cards:
3 of Diamonds
3 of Hearts
3 of Spades
4 of Clubs
4 of Diamonds
4 of Hearts
4 of Spades
5 of Clubs
5 of Diamonds
5 of Hearts
5 of Spades
6 of Clubs
6 of Diamonds
6 of Hearts
6 of Spades
7 of Clubs
7 of Diamonds
7 of Hearts
7 of Spades
8 of Clubs
8 of Diamonds
8 of Hearts
8 of Spades
9 of Clubs
9 of Diamonds
9 of Hearts
9 of Spades
10 of Clubs
10 of Diamonds
10 of Hearts
10 of Spades
Jack of Clubs
```

```
Jack of Diamonds
Jack of Hearts
Queen of Clubs
Queen of Diamonds
Queen of Hearts
Queen of Spades
King of Clubs
King of Diamonds
King of Hearts
King of Spades

Card Dealer Menu
**************************

 1. Create a deck of cards
 2. Display the deck of cards
 3. Shuffle the deck of cards
 4. Draw a card from the deck
 5. Sorts the deck of cards
 6. Print a card from the deck
 7. Compare two cards
 8. Check if two cards are same
 9. Find card by rank and suit
10. Deal a hand of cards
11. Empty the deck
 0. Quit

Enter your choice : 11
**************************


Deck emptied.

Card Dealer Menu
**************************

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

File - Card_dealer

```
 6. Print a card from the deck
 7. Compare two cards
 8. Check if two cards are same
 9. Find card by rank and suit
10. Deal a hand of cards
11. Empty the deck
 0. Quit

Enter your choice : 0
**************************

Goodbye!

Process finished with exit code 0
```


GitHub Repository: https://github.com/SnakeEyes1308/Java-Assignment-