Problem Statement: Crazy 8 Card Game (OOP Approach)

Objective

Design and implement the **Crazy 8 Card Game** using Object-Oriented Programming (OOP) principles. The program should include multiple **classes, objects, methods, attributes, and return types**, ensuring proper encapsulation, abstraction, and polymorphism.

Class Design Overview

Class Name	Description
`Card`	Represents a single playing card (Suit, Rank)
`Deck`	Represents a deck of 52 shuffled playing cards
`Hand`	Represents the collection of cards held by a player
`Player`	Represents a player in the game
`Game`	Controls the game flow, players, deck, and rules
`Crazy8Game`	Main driver class to **test all methods**

Class Breakdown

1. `Card` Class

- Represents an individual card with a **suit** and **rank**.
- Determines if one card matches another (either by suit or rank).

Attributes

Attribute	Type	Description
`suit`	`String`	Represents the suit (`Hearts` , `Diamonds` , `Clubs` , `Spades`)
`rank`	`String`	Represents the rank (`2-10, J, Q, K, A, 8`)

Methods

Method	Return Type	Description
`Card(String suit, String rank)`	Constructor	Initializes a card with suit and rank
`getSuit()`	`String`	Returns the suit of the card
`getRank()`	`String`	Returns the rank of the card
`matches(Card other)`	`boolean`	Checks if the card matches another card (by suit or rank)
`toString()`	`String`	Returns the string representation of the card

2. `Deck` Class

- Represents a deck of **52 cards**.
- Manages **shuffling** and **drawing**.

Attributes

Attribute	Type	Description
`cards`	`ArrayList<Card>`	Stores all the cards in the deck

Methods

Method	Return Type	Description
`Deck()`	Constructor	Initializes and shuffles a deck of 52 cards
`shuffle()`	`void`	Shuffles the deck
`drawCard()`	`Card`	Draws and returns the top card from the deck
`isEmpty()`	`boolean`	Checks if the deck is empty

3. `Hand` Class

- Represents a **collection of cards** held by a player.

Attributes

Attribute	Type	Description
`cards`	`ArrayList<Card>`	Stores the cards in the player's hand

Methods

Method	Return Type	Description
`Hand()`	Constructor	Initializes an empty hand
`addCard(Card card)`	`void`	Adds a card to the hand
`removeCard(Card card)`	`boolean`	Removes a card from the hand
`playCard(Card topCard)`	`Card`	Returns the first playable card from the hand
`hasCards()`	`boolean`	Checks if the hand has cards left

4. `Player` Class

- Represents an **individual player**.

Attributes

Attribute	Type	Description
`name`	`String`	Stores the player’s name
`hand`	`Hand`	Stores the player's hand
`isHuman`	`boolean`	Determines if the player is human

Methods

Method	Return Type	Description
`Player(String name, boolean isHuman)`	Constructor	Initializes a player with a name and an empty hand
`drawCard(Deck deck)`	`void`	Draws a card from the deck
`playTurn(Card topCard, Deck deck)`	`Card`	Plays a turn, returning the played card or drawing if no match
`hasWon()`	`boolean`	Checks if the player has won the game

5. `Game` Class

- Manages **game logic, deck, players, and rules**.

Attributes

Attribute	Type	Description
`deck`	`Deck`	Stores the deck of cards
`players`	`ArrayList<Player>`	Stores the list of players
`topCard`	`Card`	Represents the top card on the discard pile
`currentPlayerIndex`	`int`	Keeps track of the current player’s turn

Methods

Method	Return Type	Description
`Game(int numPlayers)`	Constructor	Initializes the game with players and deck
`startGame()`	`void`	Deals initial cards and starts the game
`playGame()`	`void`	Runs the game loop

6. `Crazy8Game` (Main Class)

- **Tests every method of all classes rigorously**.

```
public class Crazy8Game {
    public static void main(String[] args) {
        // Testing Card class
        Card c1 = new Card("Hearts", "8");
        Card c2 = new Card("Diamonds", "8");
        Card c3 = new Card("Hearts", "10");
        System.out.println("Card Test: " + c1 + " matches " + c2 + "? " + c1.matches
        System.out.println("Card Test: " + c1 + " matches " + c3 + "? " + c1.matches

        // Testing Deck class
        Deck deck = new Deck();
        System.out.println("Deck shuffled and first card drawn: " + deck.drawCard())

        // Testing Hand class
        Hand hand = new Hand();
        hand.addCard(new Card("Spades", "A"));
        hand.addCard(new Card("Clubs", "5"));
        System.out.println("Hand size after adding cards: " + hand.getCards().size()
        // Testing Player class
        Player player = new Player("Alice", true);
        player.drawCard(deck);
        System.out.println(player.getName() + " has " + player.getHand().getCards().

        // Testing Game logic
        Game game = new Game(2);
        game.playGame();
    }
}
```

Expected Output (Sample)

```
Card Test: 8 of Hearts matches 8 of Diamonds? true
Card Test: 8 of Hearts matches 10 of Hearts? true
Deck shuffled and first card drawn: K of Spades
Hand size after adding cards: 2
Alice has 6 cards.
Starting Card: 3 of Diamonds
Player 1's Turn. Top Card: 3 of Diamonds
Player 1 played: 3 of Hearts
Player 2's Turn. Top Card: 3 of Hearts
Player 2 had to draw a card.
...
🏆 Player 1 wins the game! 🎉
```