Assignment: Card class

Introduction to Software Engineering CSCI 2360

Objective

Learn how to implement conditional logic and boolean methods in a Python class by adding trump card functionality to the existing `Card` class.

Overview

You will modify the `Card` class in `card.py` to implement two methods that determine whether a card is a trump card or not. This assignment will help you understand:

- Method implementation with optional parameters
- Boolean logic and conditional statements
- Understanding game rules and translating them into code
- Working with class attributes and instance variables

Background: Understanding Trump Cards

In many card games (like Pitch), one suit is designated as the "trump suit" for a particular hand. Trump cards have special properties:

- Cards of the trump suit beat cards of any other suit
- Jokers are always considered trump cards
- Certain Jacks become "off Jacks" (trump cards from a different suit)
- The trump suit can change between hands

Task Description

The `Card` class already exists in `card.py` with basic functionality. Your task is to implement two methods that work together to determine trump status:

Method 1: `is_trump(self, suit=None)`

Implement a method that checks if the card is a trump card.

Method Signature:

```
def is_trump(self, suit=None):
    """
    Checks if the card is a trump card.

    Args:
        suit (str, optional): The suit to check against.
```

```
            If not provided, the trump suit of the card is used.

    Returns:
        bool: True if the card is a trump card, False otherwise.
    """
```

Logic Requirements:

1. If no suit parameter is provided, use the card's `trump_suit` attribute
2. If neither parameter nor attribute is set, return `False`
3. Cards of the trump suit are trump cards
4. Jokers are always trump cards (suit is 'Joker')
5. Handle "off Jacks" - Jacks of certain suits become trump when other suits are trump:
    - Jack of Clubs is trump when Spades is trump
    - Jack of Spades is trump when Clubs is trump
    - Jack of Diamonds is trump when Hearts is trump
    - Jack of Hearts is trump when Diamonds is trump

Method 2: `is_nontrump(self, suit=None)`

Implement a method that checks if the card is a non-trump card.

Method Signature:

```
def is_nontrump(self, suit=None):
    """
    Checks if the card is a non-trump card.

    Args:
        suit (str, optional): The suit to check against.
            If not provided, the trump suit of the card is used.

    Returns:
        bool: True if the card is a non-trump card, False otherwise.
    """
```

Logic Requirements:

- This method should return the opposite of `is_trump()`
- Use the same parameter handling as `is_trump()`

Implementation Hints

For `is_trump()`:

1. Start by handling the case where `suit` is `None`
2. Check if the card is a Joker first (simplest case)
3. Check if the card's suit matches the trump suit
4. Handle the off Jack cases using the card's `base_symbol()` method
5. Use tuple comparisons for off Jack logic

For `is_nontrump()`:

1. This can be implemented very simply using the `is_trump()` method
2. Consider the DRY (Don't Repeat Yourself) principle

Available Class Attributes and Methods:

- `self.suit`: The card's suit ('Spades', 'Hearts', 'Diamonds', 'Clubs', 'Joker')
- `self.trump_suit`: The currently set trump suit (or None)
- `self.base_symbol(name)`: Returns the base symbol for a card name
- `self.name`: The full name of the card
- `self.symbol`: The current symbol of the card

Test Cases to Consider

Your implementation should handle these scenarios:

1. **Regular trump cards**: Ace of Spades when Spades is trump → True
2. **Non-trump cards**: King of Hearts when Spades is trump → False
3. **Jokers**: Any joker with any trump suit → True
4. **Off Jacks**: Jack of Clubs when Spades is trump → True
5. **Regular Jacks**: Jack of Spades when Spades is trump → True
6. **No trump suit set**: Any card with no trump suit → False

Example Usage

```
# Create some cards
ace_spades = Card('Ace', 'Spades')
jack_clubs = Card('Jack', 'Clubs')
big_joker = Card('Big', 'Joker')

# Test with Spades as trump
print(ace_spades.is_trump('Spades'))    # Should print True
print(jack_clubs.is_trump('Spades'))    # Should print True (off Jack)
print(big_joker.is_trump('Spades'))     # Should print True (Joker)

print(ace_spades.is_nontrump('Hearts')) # Should print True
print(jack_clubs.is_nontrump('Hearts')) # Should print True
```

Grading Rubric

See Autograder results for distribution of points.

Submission Requirements

1. Complete the implementation of both methods in `card.py`.
2. Test your implementation with the provided test cases (use `python -m`).
3. Ensure your code follows Python style conventions (consider using `pylint`).
4. Push your modified `card.py` file to GitHub.

Due Dates

The due date is specified on Blackboard.

Good luck! Remember to test your code thoroughly and ask questions if you need clarification on the trump card rules.