Description of Yahtzee Game:

Yahtzee is a popular dice game that involves both luck and strategy. It is typically played with five six-sided dice and a scorecard. The objective of the game is to score points by rolling various combinations of dice.

Game Setup:

1. **Players:** Yahtzee can be played solo or with multiple players. Each player takes turns rolling the dice and scoring points.
2. **Dice:** The game uses five standard six-sided dice, numbered 1 through 6.
3. **Scorecard:** A scorecard is used to track points earned for different combinations of dice rolls. The scorecard contains various categories, each with its own scoring rules.

Gameplay:

1. **Rolling the Dice:**
    - On their turn, a player rolls all five dice.
    - After the initial roll, the player has the option to reroll any or all of the dice up to two more times.
    - After each roll, the player can choose which dice to keep and which to reroll.
2. **Scoring:**
    - Once the player has completed their rolls, they must choose a category on the scorecard to score their dice roll.
    - The player can only score each category once per game.
    - The score for each category is determined by the specific combination of dice rolled.
    - Categories include ones, twos, threes, fours, fives, sixes, three of a kind, four of a kind, full house, small straight, large straight, chance, and Yahtzee.
3. **Winning:**
    - The game consists of 13 rounds, with each player taking one turn per round.
    - After 13 rounds, the player with the highest total score wins.
    - In case of a tie, additional tiebreaker rules may be used.

Strategy:

- Players must carefully consider which scoring category to choose for each roll based on the current dice combination and their overall game strategy.
- It often involves weighing the potential score for different categories against the likelihood of achieving certain dice combinations.

Objective:

- The primary goal is to maximize the total score by strategically selecting scoring categories and rolling combinations that yield the highest possible scores.

Test Cases for different categories and their scores:

Test Case 1: Ones Category

- **Dice:** [1, 2, 3, 4, 5]
- **Expected Output:** Scored 1 points for Ones.

Test Case 2: Twos Category

- **Dice:** [2, 2, 3, 4, 5]
- **Expected Output:** Scored 4 points for Twos.

Test Case 3: Three of a Kind Category (Valid)

- **Dice:** [3, 3, 3, 4, 5]
- **Expected Output:** Scored 18 points for Three of a Kind.

Test Case 4: Three of a Kind Category (Invalid)

- **Dice:** [1, 2, 3, 4, 5]
- **Expected Output:** No valid combination for Three of a Kind.

Test Case 5: Full House Category (Valid)

- **Dice:** [2, 2, 3, 3, 3]
- **Expected Output:** Scored 25 points for Full House.

Test Case 6: Full House Category (Invalid)

- **Dice:** [1, 2, 3, 4, 5]
- **Expected Output:** No valid combination for Full House.

Test Case 7: Small Straight Category (Valid)

- **Dice:** [1, 2, 3, 4, 6]
- **Expected Output:** Scored 30 points for Small Straight.

Test Case 8: Small Straight Category (Invalid)

- **Dice:** [1, 2, 3, 3, 6]
- **Expected Output:** No valid combination for Small Straight.

Test Case 9: Chance Category

- **Dice:** [1, 2, 3, 4, 5]
- **Expected Output:** Scored 15 points for Chance.

Test Case 10: Yahtzee Category (Valid)

- **Dice:** [6, 6, 6, 6, 6]
- **Expected Output:** Scored 50 points for Yahtzee.

Test Case 11: Yahtzee Category (Invalid)

- **Dice:** [1, 2, 3, 4, 5]
- **Expected Output:** No valid combination for Yahtzee.

Code:

```
import random



class YahtzeeGame:

    def __init__(self):

        self.dice = [0] * 5

        self.roll_count = 0

        self.score_card = {}



    def roll_dice(self, keep=[]):

        self.dice = [random.randint(1, 6) if i not in keep else
self.dice[i] for i in range(5)]

        self.roll_count += 1



    def display_dice(self):

        print("Dice:", self.dice)



    def display_score_card(self):

        print("Score Card:")

        for category, score in self.score_card.items():

            print(f"{category}: {score}")



    def score_ones(self):
```

```
        return sum(die for die in self.dice if die == 1)


    def score_twos(self):
        return sum(die for die in self.dice if die == 2)


    def score_threes(self):
        return sum(die for die in self.dice if die == 3)


    def score_fours(self):
        return sum(die for die in self.dice if die == 4)


    def score_fives(self):
        return sum(die for die in self.dice if die == 5)


    def score_sixes(self):
        return sum(die for die in self.dice if die == 6)


    def score_three_of_a_kind(self):
        for die in set(self.dice):
            if self.dice.count(die) >= 3:
                return sum(self.dice)
        return 0


    def score_four_of_a_kind(self):
        for die in set(self.dice):
```

```
            if self.dice.count(die) >= 4:
                return sum(self.dice)
        return 0

    def score_full_house(self):
        counts = [self.dice.count(die) for die in set(self.dice)]
        if 2 in counts and 3 in counts:
            return 25
        return 0

    def score_small_straight(self):
        sorted_dice = sorted(set(self.dice))
        if len(sorted_dice) >= 4 and (sorted_dice[-1] - sorted_dice[0])
>= 3:
            return 30
        return 0

    def score_large_straight(self):
        sorted_dice = sorted(set(self.dice))
        if len(sorted_dice) == 5 and (sorted_dice[-1] - sorted_dice[0])
== 4:
            return 40
        return 0

    def score_chance(self):
        return sum(self.dice)
```

```
    def score_yahtzee(self):
        if len(set(self.dice)) == 1:
            return 50
        return 0


    def score(self, category):
        if category not in self.score_card:
            if category == "Ones":
                self.score_card[category] = self.score_ones()
            elif category == "Twos":
                self.score_card[category] = self.score_twos()
            elif category == "Threes":
                self.score_card[category] = self.score_threes()
            elif category == "Fours":
                self.score_card[category] = self.score_fours()
            elif category == "Fives":
                self.score_card[category] = self.score_fives()
            elif category == "Sixes":
                self.score_card[category] = self.score_sixes()
            elif category == "Three of a Kind":
                self.score_card[category] =
self.score_three_of_a_kind()
            elif category == "Four of a Kind":
                self.score_card[category] = self.score_four_of_a_kind()
```

```
            elif category == "Full House":
                self.score_card[category] = self.score_full_house()
            elif category == "Small Straight":
                self.score_card[category] = self.score_small_straight()
            elif category == "Large Straight":
                self.score_card[category] = self.score_large_straight()
            elif category == "Chance":
                self.score_card[category] = self.score_chance()
            elif category == "Yahtzee":
                self.score_card[category] = self.score_yahtzee()
            else:
                print("Invalid category!")
                return
            print(f"Scored {self.score_card[category]} points for {category}.")
        else:
            print(f"{category} has already been scored.")

    def play_round(self):
        print(f"Round {self.roll_count + 1}:")
        self.roll_dice()
        self.display_dice()
        for i in range(2):
            reroll_indices = input("Enter indices of dice to re-roll (comma-separated, leave blank to keep current dice): ")
            if reroll_indices:
```

```
                reroll_indices = [int(index) for index in
reroll_indices.split(",")]

                self.roll_dice(keep=reroll_indices)

                self.display_dice()

            else:

                break

        self.display_score_card()

        category = input("Enter category to score: ")

        self.score(category)




def main():

    yahtzee = YahtzeeGame()

    print("Welcome to Yahtzee!")

    while yahtzee.roll_count < 13:

        input("Press Enter to roll the dice...")

        yahtzee.play_round()

    print("Game Over!")

    yahtzee.display_score_card()




if __name__ == "__main__":

    main()
```

Code Overview:

1. **Initialization:** The `YahtzeeGame` class is initialized with attributes for dice, roll count, and score card.
2. **Rolling Dice:** The `roll_dice()` method simulates rolling five dice. The player can optionally specify which dice to keep.
3. **Displaying Dice:** The `display_dice()` method prints the current state of the dice.
4. **Displaying Score Card:** The `display_score_card()` method prints the current scores in the score card.
5. **Scoring Categories:** Methods are defined to score each Yahtzee category, such as ones, twos, three of a kind, etc.
6. **Scoring:** The `score()` method calculates and updates the score for the specified category.
7. **Playing a Round:** The `play_round()` method simulates a round of Yahtzee, allowing the player to roll the dice, reroll selected dice, score a category, and view the score card.
8. **Main Function:** The `main()` function initiates the Yahtzee game, runs rounds until all rounds are completed, and displays the final score card.

Testing Strategies:

1. **Category-Wise Testing:**
    - Individual Categories: Test each scoring category individually to ensure correct scoring.
    - Combination Categories: Test categories like three of a kind, four of a kind, full house, etc., with different combinations of dice to ensure accurate scoring.
2. **Dice Combination Testing:**
    - Test various combinations of dice rolls to verify scoring under different scenarios.
    - Include edge cases such as all dice showing the same number (potential Yahtzee) and different numbers (mixed combinations).
3. **Boundary Testing:**
    - Test boundary conditions such as the minimum and maximum values for dice rolls and the maximum number of rounds allowed.
4. **Reroll Testing:**
    - Test rerolling functionality by rerolling different sets of dice and ensuring that the correct dice are being rerolled and scored.
5. **Interactive Gameplay Testing:**
    - Manually play multiple rounds of the game, including rolling, rerolling, and scoring, to validate the overall gameplay experience.
    - Ensure that the game terminates correctly after 13 rounds and displays the final score card accurately.

Example Test Cases:

1. **Category-Wise Testing:**
    - Test scoring for "Ones" category with different combinations of dice.

    - Test scoring for "Three of a Kind" category with various combinations, including both valid and invalid cases.
2. **Dice Combination Testing:**
    - Roll five dice with all showing the same number and verify scoring for "Yahtzee" category.
    - Roll five dice with different numbers and verify scoring for "Small Straight" and "Large Straight" categories.
3. **Boundary Testing:**
    - Roll five dice with all showing the same number (maximum score for a single category).
    - Test scoring for "Chance" category with minimum and maximum values for dice.
4. **Reroll Testing:**
    - Reroll specific dice to achieve different combinations and verify scoring for appropriate categories.
5. **Interactive Gameplay Testing:**
    - Play multiple rounds of the game, manually inputting dice rolls and category scores, and verify the final score card matches expectations.

Conclusion:

By systematically testing the Yahtzee game code using the outlined strategies and example test cases, we can ensure its correctness, robustness, and adherence to game rules. Continuous testing and refinement may be necessary to address any issues or edge cases that arise during testing.