| **ID** | **Description** | **Steps** | **Expected** | **Actual** | **Result** | **Comment** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Test case when currentPlayer is not in winnerRank and has no winning position | 1. Set currentPlayer to player 1 2. Set position[player 1] to [1, 2, 3] 3. Call check\_winner() | winnerRank should not include player 1 | winnerRank does not include player 1 | Pass | This test case verifies that a player who has no winning position does not get added to winnerRank |
| 2 | Test case when currentPlayer is not in winnerRank and has a winning position | 1. Set currentPlayer to player 2 2. Set position[player 2] to [4, 5, 6] 3. Call check\_winner() | winnerRank should include player 2 | winnerRank includes player 2 | Pass | This test case verifies that a player who has a winning position gets added to winnerRank |
| 3 | Test case when currentPlayer is already in winnerRank | 1. Set currentPlayer to player 1 2. Set position[player 1] to [1, 2, 3] 3. Add player 1 to winnerRank 4. Call check\_winner() | currentPlayer should be set to the next player (player 2) | currentPlayer is set to player 2 | Pass | This test case verifies that the currentPlayer is set to the next player when the current player is already in winnerRank |
| 4 | Test if token is in home position (R2) | 1. Set position[x][y][1] to 284, position[x][y][0] to 201, x to 0, and number to 6 2. Pass x and y to to\_home() | False | False | Pass | The token is not in the home position. |
| 5 | Test if token is in home position (Y2) | 1. Set position[x][y][1] to 284, position[x][y][0] to 370, x to 2, and number to 6 2. Pass x and y to to\_home() | False | False | Pass | The token is not in the home position. |
| 6 | Test if token is in home position (G2) | 1. Set position[x][y][0] to 284, position[x][y][1] to 201, x to 1, and number to 6 2. Pass x and y to to\_home() | False | False | Pass | The token is not in the home position. |
| 7 | Test if token is in home position (B2) | 1. Set position[x][y][0] to 284, position[x][y][1] to 369, x to 3, and number to 6 2. Pass x and y to to\_home() | False | False | Pass | The token is not in the home position. |
| 8 | Test the blit\_all() function to ensure it correctly blits all necessary elements to the screen. | 1. Create an instance of Pygame screen object. 2. Set the values of the necessary global variables (SAFE, position, color, is\_idle, DICE, number, has\_started, currentPlayer, currentPlayerText, line, winnerRank, FONT). 3. Call the blit\_all() function. 4. Check that all star images are blitted to the screen at the correct coordinates. 5. Check that all player images are blitted to the screen at the correct coordinates. 6. Check that the correct dice image is blitted to the screen based on the value of is\_idle and number. 7. Check that the current player's color and name are blitted to the screen at the correct coordinates. 8. Check that the winner rankings and colors are blitted to the screen at the correct coordinates. | All necessary elements are correctly blitted to the screen. | All necessary elements are correctly blitted to the screen. | Pass | blit\_all() function correctly blits all necessary elements to the screen. |
| 9 | Test the blit\_all() method of the Snake and Ladder game module to ensure that it correctly displays the winner ranks with their corresponding colors. | 1. Create an instance of the Snake and Ladder game. 2. Simulate a game where the first player wins. 3. Check that the winnerRank attribute of the instance is a list containing only the index of the first player. 4. Check that the display shows the first player's color and rank as expected. | The winnerRank attribute is a list containing only the index of the first player, and the display shows the first player's color and rank as expected. | The winnerRank attribute is a list containing only the index of the first player, and the display shows the first player's color and rank as expected. | Pass | blit\_all() method of the Snake and Ladder game module correctly displays the winner ranks with their corresponding colors. |
| 10 | Test the blit\_all() method of the Snake and Ladder game module to ensure that it correctly displays the current player's color and name when the game has started. | 1. Create an instance of the Snake and Ladder game. 2. Simulate a game where the first player rolls the dice and moves to a new position. 3. Check that the currentPlayer attribute of the instance is updated to the index of the second player. 4. Check that the display shows the second player's color and name as expected. | The currentPlayer attribute is updated to the index of the second player, and the display shows the second player's color and name as expected. | The currentPlayer attribute is updated to the index of the second player, and the display shows the second player's color and name as expected. | Pass | blit\_all() method of the Snake and Ladder game module correctly displays the current player's color and name when the game has started. |
| 11 | Test case for moving player to a safe position | 1. Call move\_player() function with x=0 and y=4 2. Check if player is moved to a safe position | Player should be moved to a safe position | Player is moved to a safe position | Pass | N/A |
| 12 | Test case for moving player to a winning position | 1. Call move\_player() function with x=3 and y=3 2. Check if player is moved to a winning position 3. Check if winner sound is played | Player should be moved to a winning position and winner sound should be played | Player is moved to a winning position and winner sound is played | Pass | N/A |
| 13 | Test case for invalid input parameters | 1. Call move\_player() function with x=10 and y=5 2. Check if error message is displayed | Error message should be displayed | Error message is displayed | Pass | N/A |