Baseball World Series Simulation

On average, how many games (out of 7) does it take for a team to win the World Series?

- An American League (AL) baseball team is considered to have a 60% chance of beating the National League (NL) team in any given World Series game. A team wins the World Series by being the first to win four individual games.
- Model: How we are encoding our question as numbers
- Trial: A single World Series result
- Simulation: A collection of many trials

Simulating a single game ¶

- Model: Generate a random integer between 1 and 100 to simulate an individual game
  - #'s 1-60 will represent an American League (AL) team win
    - 60% chance to win
  - #'s 61-100 will represent a National League (NL) team win
    - 40% chance to win

In [1]:
```
# Import the numpy package
import numpy as np

# Generate random number between 1-100
single_game = np.random.randint(1,101) # THIS IS NOT A TRIAL YET

# Test the percentage ranges
if (single_game <=60):
    print("AL Wins")
else:
    print("NL Wins")
```

```
NL Wins
```

Copy and paste code into a function

- Return a String of the winning division
- Abstraction!

In [2]:
```
# Create function to simulate a single game
def play_single_game():
    single_game = np.random.randint(1,101)
    if (single_game <= 60):
        return "AL"
    else:
        return "NL"
print(play_single_game())
```

```
AL
```

Simulating a single World Series

- Create two variables for AL and NL win counts
- While both counts are below 4 wins, play a single game
    - Update variables appropriately
- Return a tuple in the form (total games played, winning division)

In [3]:
```
# Import the numpy package
import numpy as np

# Create variables
nl_wins = 0
al_wins = 0

# While loop
while nl_wins < 4 and al_wins < 4:
    wsg = play_single_game()
    if wsg == "AL":
        al_wins = al_wins + 1
    else:
        nl_wins = nl_wins + 1

# Total games played before a division reached four wins
total_num_games = al_wins + nl_wins

# Print winner and games
if al_wins == 4:
    print("AL", total_num_games)
else:
    print("NL", total_num_games)
```

```
NL 7
```

The above block of code is a single "trial" of our simulation

Copy and paste code into a function

- Return a tuple in the format (winning division, total number of games played)
- Abstraction!

In [4]:
```
# Create function to simulate a World Series
def play_world_series():
    # Import the numpy package
    import numpy as np

    # Create variables
    nl_wins = 0
    al_wins = 0

    # While loop
    while nl_wins < 4 and al_wins < 4:
        wsg = play_single_game()
        if wsg == "AL":
            al_wins = al_wins + 1
        else:
            nl_wins = nl_wins + 1

    # Total games played before a division reached four wins
    total_num_games = al_wins + nl_wins

    # Print winner and games
    if al_wins == 4:
        return("AL", total_num_games)
    else:
        return("NL", total_num_games)

print(play_world_series())
```

```
('NL', 5)
```

Finding the average number of total games played to win a World Series

Simulate a large number of World Series, analyze the results

- Create an array to hold your results
- Create a loop to run 1000 times
  - Add results of a single World Series to the array

In [5]:
```
# Create variables
ws_results = []

# Create a loop that repeats a large number of trials (World Series), in this
 case 10,000
for i in range(1000):
    single_ws = play_world_series()
    ws_results.append(single_ws)

# Print the results
print(ws_results)
```

```
[('AL', 6), ('AL', 6), ('AL', 7), ('AL', 5), ('NL', 7), ('NL', 5), ('AL', 6),
('AL', 7), ('NL', 7), ('AL', 7), ('AL', 5), ('NL', 6), ('NL', 6), ('AL', 4),
('AL', 5), ('NL', 7), ('AL', 7), ('AL', 4), ('NL', 7), ('AL', 4), ('AL', 4),
('NL', 7), ('AL', 6), ('AL', 4), ('AL', 5), ('NL', 6), ('AL', 6), ('NL', 7),
('AL', 4), ('AL', 7), ('AL', 7), ('NL', 7), ('AL', 5), ('AL', 7), ('NL', 5),
('NL', 6), ('AL', 5), ('AL', 7), ('AL', 7), ('NL', 7), ('NL', 6), ('AL', 6),
('AL', 4), ('AL', 5), ('AL', 4), ('AL', 7), ('AL', 7), ('NL', 7), ('AL', 5),
('AL', 6), ('NL', 7), ('AL', 5), ('AL', 5), ('NL', 7), ('NL', 5), ('AL', 7),
('AL', 7), ('NL', 7), ('AL', 6), ('AL', 5), ('NL', 6), ('AL', 5), ('AL', 7),
('AL', 4), ('NL', 4), ('NL', 6), ('AL', 5), ('AL', 6), ('AL', 6), ('AL', 6),
('AL', 5), ('AL', 5), ('AL', 6), ('AL', 4), ('AL', 6), ('NL', 5), ('AL', 6),
('AL', 4), ('AL', 5), ('AL', 7), ('NL', 7), ('NL', 5), ('AL', 6), ('AL', 5),
('NL', 4), ('AL', 4), ('AL', 5), ('AL', 7), ('AL', 7), ('AL', 4), ('AL', 5),
('AL', 4), ('NL', 5), ('NL', 7), ('AL', 6), ('AL', 4), ('AL', 6), ('AL', 7),
('AL', 6), ('AL', 6), ('AL', 4), ('NL', 7), ('AL', 5), ('NL', 5), ('AL', 7),
('NL', 7), ('AL', 6), ('AL', 7), ('AL', 7), ('NL', 7), ('NL', 7), ('AL', 4),
('AL', 4), ('NL', 5), ('AL', 7), ('AL', 6), ('AL', 5), ('NL', 7), ('AL', 7),
('AL', 5), ('AL', 7), ('NL', 7), ('AL', 4), ('AL', 5), ('AL', 7), ('AL', 4),
('AL', 6), ('NL', 7), ('AL', 7), ('AL', 6), ('AL', 5), ('NL', 5), ('NL', 5),
('AL', 6), ('NL', 7), ('NL', 6), ('NL', 7), ('AL', 6), ('NL', 4), ('NL', 7),
('AL', 6), ('NL', 6), ('AL', 7), ('NL', 5), ('AL', 5), ('NL', 5), ('AL', 5),
('AL', 7), ('AL', 5), ('NL', 6), ('AL', 5), ('NL', 7), ('AL', 5), ('AL', 7),
('NL', 6), ('AL', 6), ('AL', 7), ('NL', 7), ('AL', 6), ('AL', 5), ('AL', 7),
('AL', 6), ('NL', 5), ('AL', 6), ('AL', 7), ('AL', 6), ('AL', 6), ('AL', 7),
('AL', 5), ('AL', 7), ('AL', 7), ('AL', 5), ('NL', 7), ('AL', 5), ('AL', 5),
('NL', 6), ('AL', 5), ('AL', 6), ('AL', 5), ('AL', 6), ('AL', 7), ('NL', 5),
('NL', 6), ('AL', 7), ('NL', 5), ('AL', 6), ('AL', 4), ('AL', 7), ('AL', 5),
('NL', 7), ('AL', 7), ('NL', 7), ('AL', 7), ('NL', 6), ('AL', 6), ('AL', 7),
('AL', 6), ('AL', 5), ('NL', 7), ('NL', 7), ('AL', 5), ('NL', 7), ('AL', 6),
('AL', 5), ('AL', 5), ('AL', 6), ('AL', 4), ('AL', 5), ('AL', 4), ('NL', 5),
('NL', 6), ('AL', 7), ('AL', 6), ('AL', 6), ('AL', 4), ('AL', 7), ('AL', 7),
('AL', 4), ('AL', 7), ('NL', 5), ('AL', 5), ('AL', 6), ('NL', 5), ('NL', 7),
('AL', 6), ('AL', 4), ('AL', 6), ('AL', 5), ('AL', 7), ('AL', 6), ('AL', 6),
('AL', 6), ('AL', 5), ('AL', 4), ('NL', 4), ('AL', 5), ('NL', 7), ('AL', 6),
('NL', 7), ('NL', 7), ('NL', 5), ('AL', 6), ('NL', 7), ('AL', 4), ('AL', 6),
('AL', 5), ('AL', 5), ('AL', 7), ('AL', 5), ('AL', 5), ('AL', 4), ('AL', 6),
('AL', 7), ('AL', 7), ('NL', 4), ('AL', 5), ('AL', 7), ('AL', 7), ('AL', 6),
('AL', 6), ('AL', 6), ('NL', 5), ('AL', 4), ('AL', 6), ('AL', 6), ('NL', 6),
('NL', 6), ('NL', 6), ('NL', 5), ('AL', 6), ('NL', 7), ('AL', 6), ('NL', 7),
('NL', 6), ('AL', 6), ('NL', 6), ('AL', 5), ('AL', 6), ('AL', 6), ('AL', 7),
('AL', 6), ('NL', 7), ('AL', 4), ('AL', 5), ('AL', 7), ('AL', 4), ('AL', 6),
('AL', 7), ('NL', 7), ('AL', 7), ('NL', 7), ('NL', 7), ('NL', 4), ('NL', 5),
('AL', 6), ('NL', 6), ('AL', 5), ('AL', 4), ('AL', 5), ('NL', 7), ('AL', 4),
('AL', 5), ('NL', 7), ('AL', 7), ('AL', 6), ('NL', 6), ('AL', 5), ('AL', 7),
('AL', 6), ('NL', 7), ('AL', 6), ('AL', 4), ('NL', 5), ('AL', 5), ('AL', 4),
('AL', 5), ('AL', 4), ('NL', 7), ('AL', 6), ('NL', 5), ('AL', 4), ('NL', 7),
('AL', 6), ('NL', 5), ('AL', 5), ('AL', 5), ('AL', 4), ('AL', 6), ('NL', 6),
('NL', 6), ('NL', 7), ('AL', 6), ('AL', 7), ('AL', 5), ('AL', 6), ('NL', 7),
('AL', 7), ('NL', 5), ('AL', 5), ('AL', 5), ('AL', 6), ('AL', 5), ('NL', 7),
('AL', 5), ('AL', 7), ('AL', 7), ('AL', 6), ('AL', 5), ('NL', 4), ('AL', 7),
('AL', 6), ('AL', 4), ('AL', 6), ('AL', 5), ('AL', 5), ('NL', 5), ('AL', 7),
('AL', 7), ('AL', 7), ('AL', 6), ('AL', 5), ('AL', 4), ('NL', 5), ('AL', 5),
('AL', 6), ('AL', 5), ('NL', 6), ('AL', 4), ('AL', 7), ('AL', 7), ('NL', 7),
('NL', 7), ('NL', 7), ('AL', 5), ('AL', 6), ('AL', 7), ('NL', 6), ('AL', 5),
('AL', 7), ('AL', 5), ('AL', 7), ('NL', 6), ('AL', 7), ('AL', 4), ('NL', 6),
('AL', 5), ('AL', 4), ('AL', 5), ('NL', 7), ('AL', 5), ('AL', 7), ('AL', 4),
('AL', 4), ('NL', 7), ('NL', 6), ('AL', 5), ('AL', 5), ('NL', 7), ('AL', 6),
('AL', 5), ('NL', 5), ('NL', 7), ('NL', 5), ('NL', 4), ('NL', 7), ('NL', 7),
('AL', 6), ('NL', 7), ('NL', 7), ('NL', 7), ('NL', 5), ('NL', 7), ('AL', 5),
('NL', 7), ('NL', 6), ('AL', 5), ('NL', 7), ('NL', 5), ('AL', 6), ('AL', 4),
('NL', 5), ('NL', 4), ('NL', 7), ('NL', 4), ('AL', 4), ('AL', 6), ('NL', 6),
('NL', 7), ('AL', 6), ('NL', 6), ('AL', 5), ('NL', 6), ('AL', 4), ('NL', 5),
('AL', 4), ('AL', 5), ('AL', 6), ('NL', 6), ('AL', 6), ('NL', 5), ('AL', 6),
('AL', 7), ('AL', 7), ('AL', 6), ('NL', 5), ('NL', 6), ('AL', 4), ('NL', 6),
('AL', 4), ('AL', 5), ('AL', 7), ('AL', 6), ('AL', 6), ('AL', 7), ('AL', 5),
('AL', 5), ('AL', 6), ('NL', 6), ('AL', 5), ('NL', 7), ('AL', 5), ('NL', 5),
('AL', 5), ('AL', 7), ('AL', 7), ('AL', 6), ('NL', 6), ('AL', 5), ('AL', 6),
('NL', 7), ('AL', 7), ('AL', 6), ('NL', 7), ('AL', 5), ('AL', 4), ('AL', 7),
('AL', 5), ('AL', 5), ('AL', 5), ('NL', 5), ('AL', 6), ('NL', 7), ('NL', 7),
('AL', 5), ('NL', 4), ('AL', 5), ('AL', 6), ('AL', 7), ('AL', 5), ('AL', 6),
('NL', 6), ('NL', 7), ('AL', 4), ('AL', 5), ('AL', 7), ('AL', 6), ('AL', 7),
('AL', 7), ('AL', 5), ('AL', 6), ('AL', 5), ('AL', 5), ('AL', 6), ('NL', 4),
('AL', 5), ('AL', 4), ('AL', 5), ('AL', 5), ('AL', 6), ('AL', 5), ('AL', 6),
('NL', 6), ('AL', 5), ('NL', 7), ('AL', 7), ('AL', 5), ('AL', 6), ('AL', 4),
('AL', 6), ('AL', 4), ('AL', 6), ('NL', 6), ('AL', 5), ('NL', 7), ('AL', 6),
('AL', 4), ('AL', 7), ('AL', 7), ('AL', 6), ('NL', 4), ('AL', 6), ('NL', 6),
('AL', 6), ('NL', 4), ('AL', 6), ('NL', 7), ('AL', 6), ('NL', 7), ('AL', 5),
('AL', 4), ('AL', 6), ('AL', 4), ('AL', 4), ('AL', 7), ('NL', 5), ('AL', 7),
('AL', 4), ('NL', 6), ('AL', 5), ('AL', 5), ('NL', 5), ('AL', 6), ('AL', 5),
('AL', 6), ('AL', 6), ('NL', 5), ('AL', 5), ('NL', 7), ('NL', 6), ('AL', 6),
('AL', 4), ('AL', 4), ('NL', 6), ('NL', 5), ('NL', 7), ('AL', 4), ('NL', 7),
('AL', 7), ('AL', 6), ('AL', 7), ('AL', 6), ('AL', 6), ('AL', 5), ('NL', 6),
('AL', 6), ('AL', 6), ('AL', 5), ('AL', 5), ('NL', 4), ('AL', 6), ('NL', 6),
('AL', 6), ('NL', 4), ('AL', 5), ('NL', 5), ('AL', 6), ('AL', 7), ('NL', 5),
('AL', 7), ('AL', 7), ('AL', 7), ('AL', 5), ('NL', 6), ('AL', 5), ('AL', 5),
('AL', 6), ('NL', 7), ('AL', 7), ('AL', 5), ('AL', 5), ('AL', 4), ('AL', 7),
('NL', 7), ('AL', 4), ('AL', 5), ('NL', 6), ('NL', 7), ('AL', 4), ('AL', 6),
('NL', 7), ('AL', 6), ('NL', 6), ('AL', 7), ('NL', 5), ('AL', 7), ('NL', 7),
('NL', 7), ('NL', 5), ('AL', 7), ('AL', 7), ('AL', 7), ('AL', 6), ('AL', 7),
('AL', 7), ('AL', 4), ('AL', 5), ('AL', 7), ('NL', 6), ('AL', 6), ('NL', 7),
('AL', 6), ('AL', 7), ('AL', 7), ('AL', 6), ('AL', 4), ('AL', 6), ('AL', 7),
('NL', 5), ('AL', 4), ('NL', 5), ('AL', 6), ('NL', 7), ('AL', 5), ('NL', 5),
('AL', 7), ('NL', 4), ('AL', 7), ('AL', 7), ('AL', 6), ('NL', 7), ('NL', 6),
('AL', 5), ('AL', 5), ('AL', 5), ('AL', 6), ('AL', 4), ('NL', 6), ('NL', 4),
('AL', 6), ('AL', 7), ('NL', 7), ('NL', 6), ('AL', 7), ('AL', 5), ('AL', 5),
('AL', 4), ('NL', 5), ('AL', 5), ('AL', 6), ('AL', 6), ('AL', 4), ('AL', 4),
('AL', 4), ('AL', 7), ('AL', 5), ('AL', 5), ('NL', 6), ('AL', 6), ('AL', 4),
('AL', 4), ('NL', 5), ('NL', 7), ('NL', 7), ('AL', 5), ('AL', 5), ('AL', 6),
('AL', 5), ('AL', 7), ('NL', 7), ('AL', 4), ('AL', 5), ('AL', 7), ('AL', 6),
('AL', 6), ('AL', 5), ('NL', 6), ('AL', 5), ('NL', 7), ('NL', 5), ('AL', 6),
('AL', 5), ('AL', 5), ('AL', 5), ('AL', 6), ('AL', 5), ('AL', 6), ('NL', 4),
('NL', 7), ('AL', 6), ('NL', 6), ('AL', 7), ('AL', 5), ('NL', 5), ('AL', 5),
('AL', 7), ('AL', 5), ('AL', 4), ('AL', 6), ('AL', 5), ('AL', 4), ('AL', 5),
('AL', 5), ('AL', 5), ('AL', 5), ('NL', 7), ('AL', 6), ('AL', 6), ('AL', 7),
('AL', 5), ('AL', 4), ('AL', 5), ('NL', 5), ('AL', 6), ('NL', 6), ('AL', 4),
('AL', 6), ('AL', 5), ('NL', 6), ('AL', 4), ('AL', 6), ('AL', 5), ('NL', 7),
('AL', 4), ('AL', 5), ('AL', 7), ('AL', 7), ('AL', 7), ('AL', 7), ('NL', 6),
('NL', 4), ('AL', 6), ('NL', 4), ('NL', 7), ('AL', 5), ('NL', 7), ('AL', 6),
('AL', 6), ('NL', 6), ('NL', 6), ('AL', 7), ('NL', 7), ('AL', 5), ('AL', 6),
('AL', 5), ('NL', 7), ('AL', 5), ('AL', 4), ('AL', 6), ('NL', 5), ('AL', 4),
('AL', 4), ('NL', 7), ('AL', 6), ('AL', 7), ('AL', 7), ('NL', 5), ('AL', 6),
('NL', 6), ('AL', 6), ('NL', 7), ('AL', 6), ('NL', 6), ('AL', 5), ('AL', 5),
('AL', 6), ('AL', 7), ('NL', 6), ('NL', 6), ('AL', 7), ('AL', 6), ('AL', 5),
('NL', 7), ('AL', 5), ('AL', 5), ('AL', 6), ('AL', 6), ('NL', 4), ('AL', 6),
('AL', 6), ('AL', 5), ('AL', 7), ('NL', 7), ('AL', 5), ('NL', 5), ('AL', 6),
('AL', 5), ('NL', 7), ('AL', 6), ('AL', 4), ('NL', 6), ('AL', 6), ('AL', 7),
('AL', 4), ('AL', 5), ('NL', 6), ('AL', 6), ('AL', 6), ('NL', 6), ('AL', 5),
('AL', 5), ('AL', 6), ('AL', 5), ('AL', 6), ('AL', 7), ('AL', 6), ('AL', 6),
('AL', 5), ('AL', 7), ('NL', 7), ('AL', 7), ('NL', 7), ('NL', 7), ('NL', 6),
('AL', 6), ('NL', 5), ('NL', 7), ('AL', 7), ('AL', 7), ('AL', 6), ('AL', 7),
('AL', 7), ('NL', 5), ('AL', 4), ('NL', 7), ('AL', 5), ('AL', 7), ('NL', 5),
('AL', 6), ('AL', 7), ('AL', 6), ('AL', 7), ('NL', 7), ('AL', 6), ('AL', 6),
('AL', 6), ('AL', 7), ('NL', 7), ('AL', 4), ('AL', 7), ('AL', 5), ('AL', 6),
('NL', 6), ('AL', 5), ('AL', 7), ('NL', 7), ('NL', 6), ('AL', 5), ('AL', 7),
('AL', 7), ('AL', 6), ('NL', 6), ('NL', 7), ('AL', 6), ('AL', 5), ('AL', 7),
('AL', 6), ('AL', 5), ('AL', 4), ('AL', 5), ('NL', 6), ('AL', 5), ('AL', 6),
('AL', 6), ('AL', 5), ('AL', 4), ('AL', 6), ('NL', 5), ('NL', 5), ('AL', 4),
('AL', 6), ('NL', 7), ('AL', 5), ('AL', 7), ('AL', 6), ('AL', 4), ('AL', 7),
('AL', 6), ('AL', 6), ('AL', 5), ('AL', 5), ('AL', 4), ('AL', 5), ('NL', 6),
('AL', 7), ('AL', 7), ('NL', 7), ('AL', 6), ('NL', 6), ('NL', 7), ('AL', 4),
('AL', 7), ('AL', 7), ('AL', 6), ('AL', 6), ('NL', 5), ('AL', 5), ('AL', 4),
('AL', 4), ('AL', 7), ('NL', 6), ('AL', 6), ('AL', 6), ('AL', 4), ('AL', 5),
('NL', 7), ('AL', 7), ('AL', 7), ('AL', 5), ('AL', 5), ('NL', 7), ('AL', 7),
('NL', 4), ('NL', 6), ('AL', 6), ('AL', 5), ('AL', 7), ('AL', 7), ('AL', 5),
('AL', 6), ('AL', 7), ('AL', 7), ('AL', 7), ('AL', 5), ('NL', 6), ('AL', 7),
('AL', 7), ('AL', 5), ('AL', 4), ('NL', 5), ('NL', 7), ('AL', 4), ('AL', 6),
('AL', 7), ('AL', 6), ('NL', 7), ('AL', 6), ('AL', 6), ('AL', 7), ('AL', 7),
('AL', 5), ('AL', 5), ('AL', 5), ('AL', 5), ('AL', 5), ('NL', 5), ('AL', 6),
('NL', 7), ('NL', 7), ('AL', 5), ('AL', 5), ('NL', 7), ('AL', 6), ('NL', 7),
('AL', 7), ('AL', 5), ('NL', 5), ('AL', 7), ('NL', 4), ('NL', 7), ('AL', 4),
('AL', 5), ('AL', 7), ('NL', 6), ('AL', 5), ('AL', 6), ('AL', 6), ('NL', 7),
('AL', 6), ('AL', 7), ('NL', 7), ('AL', 7), ('NL', 4), ('AL', 5), ('AL', 5),
('AL', 4), ('NL', 7), ('AL', 5), ('AL', 7), ('NL', 6), ('AL', 4)]
```

Analyze the results

Find the counts of each number of total games (How many times did it take 4, 5, 6, 7 games to win?)

- Create an array to hold the results
- Loop through each tuple, read second value (total games played)
- Add the value to array

In [6]:
```
# Create variables
total_games_array = []

# Loop through each individual World Series within the ws_results array
    # Store second value of the tuple at index 1 (total number of games)
    # Add game total to the total_games_array
for world_series in ws_results:
    game_total = world_series[1]
    total_games_array.append(game_total)

# Print results
print(total_games_array)
```

```
[6, 6, 7, 5, 7, 5, 6, 7, 7, 7, 5, 6, 6, 4, 5, 7, 7, 4, 7, 4, 4, 7, 6, 4, 5,
6, 6, 7, 4, 7, 7, 7, 5, 7, 5, 6, 5, 7, 7, 7, 6, 6, 4, 5, 4, 7, 7, 7, 5, 6, 7,
5, 5, 7, 5, 7, 7, 7, 6, 5, 6, 5, 7, 4, 4, 6, 5, 6, 6, 6, 5, 5, 6, 4, 6, 5, 6,
4, 5, 7, 7, 5, 6, 5, 4, 4, 5, 7, 7, 4, 5, 4, 5, 7, 6, 4, 6, 7, 6, 6, 4, 7, 5,
5, 7, 7, 6, 7, 7, 7, 7, 4, 4, 5, 7, 6, 5, 7, 7, 5, 7, 7, 4, 5, 7, 4, 6, 7, 7,
6, 5, 5, 5, 6, 7, 6, 7, 6, 4, 7, 6, 6, 7, 5, 5, 5, 5, 7, 5, 6, 5, 7, 5, 7, 6,
6, 7, 7, 6, 5, 7, 6, 5, 6, 7, 6, 6, 7, 5, 7, 7, 5, 7, 5, 5, 6, 5, 6, 5, 6, 7,
5, 6, 7, 5, 6, 4, 7, 5, 7, 7, 7, 7, 6, 6, 7, 6, 5, 7, 7, 5, 7, 6, 5, 5, 6, 4,
5, 4, 5, 6, 7, 6, 6, 4, 7, 7, 4, 7, 5, 5, 6, 5, 7, 6, 4, 6, 5, 7, 6, 6, 6, 5,
4, 4, 5, 7, 6, 7, 7, 5, 6, 7, 4, 6, 5, 5, 7, 5, 5, 4, 6, 7, 7, 4, 5, 7, 7, 6,
6, 6, 5, 4, 6, 6, 6, 6, 6, 5, 6, 7, 6, 7, 6, 6, 6, 5, 6, 6, 7, 6, 7, 4, 5, 7,
4, 6, 7, 7, 7, 7, 7, 4, 5, 6, 6, 5, 4, 5, 7, 4, 5, 7, 7, 6, 6, 5, 7, 6, 7, 6,
4, 5, 5, 4, 5, 4, 7, 6, 5, 4, 7, 6, 5, 5, 5, 4, 6, 6, 6, 7, 6, 7, 5, 6, 7, 7,
5, 5, 5, 6, 5, 7, 5, 7, 7, 6, 5, 4, 7, 6, 4, 6, 5, 5, 5, 7, 7, 7, 6, 5, 4, 5,
5, 6, 5, 6, 4, 7, 7, 7, 7, 7, 5, 6, 7, 6, 5, 7, 5, 7, 6, 7, 4, 6, 5, 4, 5, 7,
5, 7, 4, 4, 7, 6, 5, 5, 7, 6, 5, 5, 7, 5, 4, 7, 7, 6, 7, 7, 7, 5, 7, 5, 7, 6,
5, 7, 5, 6, 4, 5, 4, 7, 4, 4, 6, 6, 7, 6, 6, 5, 6, 4, 5, 4, 5, 6, 6, 6, 5, 6,
7, 7, 6, 5, 6, 4, 6, 4, 5, 7, 6, 6, 7, 5, 5, 6, 6, 5, 7, 5, 5, 5, 7, 7, 6, 6,
5, 6, 7, 7, 6, 7, 5, 4, 7, 5, 5, 5, 5, 6, 7, 7, 5, 4, 5, 6, 7, 5, 6, 6, 7, 4,
5, 7, 6, 7, 7, 5, 6, 5, 5, 6, 4, 5, 4, 5, 5, 6, 5, 6, 6, 5, 7, 7, 5, 6, 4, 6,
4, 6, 6, 5, 7, 6, 4, 7, 7, 6, 4, 6, 6, 6, 4, 6, 7, 6, 7, 5, 4, 6, 4, 4, 7, 5,
7, 4, 6, 5, 5, 5, 6, 5, 6, 6, 5, 5, 7, 6, 6, 4, 4, 6, 5, 7, 4, 7, 7, 6, 7, 6,
6, 5, 6, 6, 6, 5, 5, 4, 6, 6, 6, 4, 5, 5, 6, 7, 5, 7, 7, 7, 5, 6, 5, 5, 6, 7,
7, 5, 5, 4, 7, 7, 4, 5, 6, 7, 4, 6, 7, 6, 6, 7, 5, 7, 7, 7, 5, 7, 7, 7, 6, 7,
7, 4, 5, 7, 6, 6, 7, 6, 7, 7, 6, 4, 6, 7, 5, 4, 5, 6, 7, 5, 5, 7, 4, 7, 7, 6,
7, 6, 5, 5, 5, 6, 4, 6, 4, 6, 7, 7, 6, 7, 5, 5, 4, 5, 5, 6, 6, 4, 4, 4, 7, 5,
5, 6, 6, 4, 4, 5, 7, 7, 5, 5, 6, 5, 7, 7, 4, 5, 7, 6, 6, 5, 6, 5, 7, 5, 6, 5,
5, 5, 6, 5, 6, 4, 7, 6, 6, 7, 5, 5, 5, 7, 5, 4, 6, 5, 4, 5, 5, 5, 5, 7, 6, 6,
7, 5, 4, 5, 5, 6, 6, 4, 6, 5, 6, 4, 6, 5, 7, 4, 5, 7, 7, 7, 7, 6, 4, 6, 4, 7,
5, 7, 6, 6, 6, 6, 7, 7, 5, 6, 5, 7, 5, 4, 6, 5, 4, 4, 7, 6, 7, 7, 5, 6, 6, 6,
7, 6, 6, 5, 5, 6, 7, 6, 6, 7, 6, 5, 7, 5, 5, 6, 6, 4, 6, 6, 5, 7, 7, 5, 5, 6,
5, 7, 6, 4, 6, 6, 7, 4, 5, 6, 6, 6, 6, 5, 5, 6, 5, 6, 7, 6, 6, 5, 7, 7, 7, 7,
7, 6, 6, 5, 7, 7, 7, 6, 7, 7, 5, 4, 7, 5, 7, 5, 6, 7, 6, 7, 7, 6, 6, 6, 7, 7,
4, 7, 5, 6, 6, 5, 7, 7, 6, 5, 7, 7, 6, 6, 7, 6, 5, 7, 6, 5, 4, 5, 6, 5, 6, 6,
5, 4, 6, 5, 5, 4, 6, 7, 5, 7, 6, 4, 7, 6, 6, 5, 5, 4, 5, 6, 7, 7, 7, 6, 6, 7,
4, 7, 7, 6, 6, 5, 5, 4, 4, 7, 6, 6, 6, 4, 5, 7, 7, 7, 5, 5, 7, 7, 4, 6, 6, 5,
7, 7, 5, 6, 7, 7, 7, 5, 6, 7, 7, 5, 4, 5, 7, 4, 6, 7, 6, 7, 6, 6, 7, 7, 5, 5,
5, 5, 5, 5, 6, 7, 7, 5, 5, 7, 6, 7, 7, 5, 5, 7, 4, 7, 4, 5, 7, 6, 5, 6, 6, 7,
6, 7, 7, 7, 4, 5, 5, 4, 7, 5, 7, 6, 4]
```

Create bins for each total

```
In [7]: # Create the bin
        game_bin = np.bincount(total_games_array)

        # Print the results
        print(game_bin)

        [  0   0   0   0 131 275 291 303]
```

```
In [8]: # Make sure this adds to 1,000!
        print(np.sum(game_bin))

        1000
```

Calculate the average of these numbers

- This will represent the average number of games played to win the World Series after 10,000 simulations

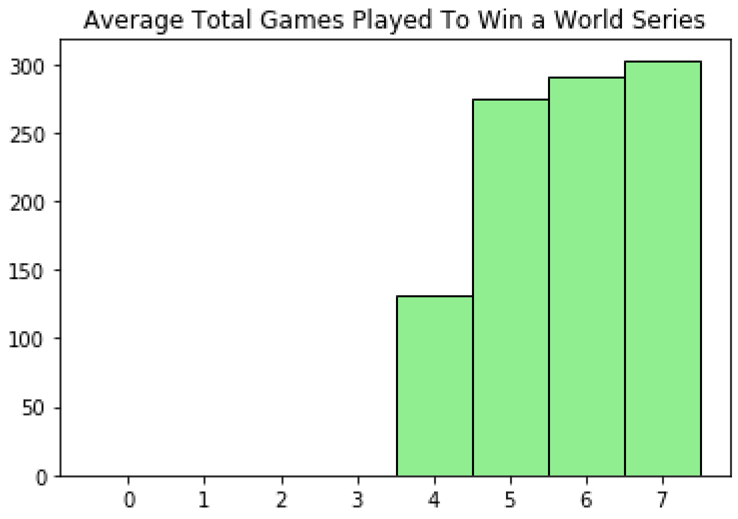
```
In [9]: # Use np.average(array)
        average = np.average(total_games_array)
        print(average)

        5.766
```

Graph the results

Histogram

- Frequency chart

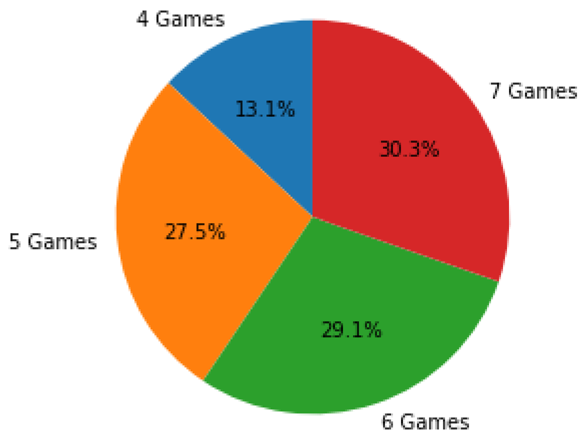
In [10]:
```
# Import the library
import matplotlib.pyplot as plot
%matplotlib inline
# Magic to allow the graph to display directly in this notebook

# Create bins/dividers for your data
bins = [0, 1, 2, 3, 4, 5, 6, 7, 8]

# plot.hist(array, bins, alignment, graph color, border color)
plot.hist(total_games_array, bins, align='left', color='lightgreen' ,edgecolor
='black')
plot.title("Average Total Games Played To Win a World Series")
plot.show()
```



Pie Chart

- Percentage chart

In [11]:
```
# source: http://matplotlib.org/examples/pie_and_polar_charts/pie_demo_feature
s.html

# Import the library
import matplotlib.pyplot as plot
%matplotlib inline
# Magic to allow the graph to display directly in this notebook

# Create an array of labels
labels = ["4 Games", "5 Games", "6 Games", "7 Games"]

# Crop the game_bin array to exclude the 0's
    # Only need indices 4 to the end
game_bin = game_bin[4:]

# Explode option
    # 'Slices' appear distanced from the center
        # Larger numbers = further explosion
    # Explode array should be same size as labels and
explode = (0, 0, 0, 0)

# Use matplotlib module subplots() to get data for various charts
    # Returns a tuple in the form (figure,axes)
fig1,ax1 = plot.subplots()

# Use axes to create a pie chart
    # ax1.pie(data array, explode array, labels array, starting angle)
ax1.pie(game_bin, explode, labels, autopct='%1.1f%%', startangle=90)
ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

plot.show()
```



Additional data points

- How many times did an American League team win the World Series compared to the National League?

In [12]:
```
# Create variables
winning_division_array = []

# Loop through each individual World Series within the ws_results array
    # Store first value of the tuple at index 0 (winning division)
    # Add winning division to the winning_division_array
for world_series in ws_results:
    division = world_series[0]
    winning_division_array.append(division)

# Print results
print(winning_division_array)
```

```
['AL', 'AL', 'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'N
L', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL',
'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'NL', 'AL', 'AL',
'AL', 'NL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL',
'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL',
'NL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL',
'AL', 'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL',
'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL',
'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL',
'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL',
'AL', 'AL', 'NL', 'NL', 'AL', 'NL', 'NL', 'NL', 'AL', 'NL', 'NL', 'AL', 'NL',
'AL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'AL', 'AL', 'NL',
'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL',
'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL',
'NL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'AL', 'NL',
'AL', 'AL', 'AL', 'AL', 'NL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL',
'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL',
'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL',
'AL', 'NL', 'AL', 'NL', 'AL', 'NL', 'NL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL',
'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL',
'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'NL', 'NL', 'NL', 'AL', 'NL', 'AL',
'NL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL',
'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'NL', 'NL', 'NL', 'AL', 'NL', 'AL', 'AL',
'AL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'AL',
'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'AL', 'NL', 'AL', 'NL',
'AL', 'AL', 'AL', 'AL', 'NL', 'NL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL',
'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL',
'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL',
'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'NL', 'NL', 'AL', 'AL', 'AL',
'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL',
'AL', 'AL', 'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'NL',
'NL', 'NL', 'NL', 'NL', 'AL', 'NL', 'NL', 'NL', 'NL', 'NL', 'AL', 'NL', 'NL',
'AL', 'NL', 'NL', 'AL', 'AL', 'NL', 'NL', 'NL', 'NL', 'AL', 'AL', 'NL', 'NL',
'AL', 'NL', 'AL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'AL',
'AL', 'AL', 'AL', 'NL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL',
'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL',
'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL',
'AL', 'NL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'NL', 'AL',
'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL',
'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL',
'AL', 'AL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL',
'AL', 'NL', 'AL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL',
'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL',
'NL', 'AL', 'AL', 'AL', 'NL', 'NL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL',
'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'AL', 'NL', 'AL',
'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL',
'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'NL',
'AL', 'NL', 'AL', 'NL', 'AL', 'NL', 'NL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL',
'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL',
'AL', 'NL', 'AL', 'NL', 'AL', 'NL', 'AL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL',
'NL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'NL', 'NL',
'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL',
'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'NL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL',
'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'NL', 'AL', 'AL',
'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'NL', 'AL', 'NL', 'AL', 'AL', 'NL', 'AL',
'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL',
'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL',
'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'NL', 'AL', 'NL', 'NL',
'AL', 'NL', 'AL', 'AL', 'NL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'AL',
'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'AL',
'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'AL', 'NL',
'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'AL',
'AL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL',
'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL',
'NL', 'NL', 'AL', 'NL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL',
'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL',
'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'AL', 'AL',
'NL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL',
'AL', 'AL', 'AL', 'NL', 'NL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL',
'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'AL', 'NL', 'NL',
'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL',
'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'NL', 'AL', 'AL',
'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL',
'NL', 'NL', 'AL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'AL', 'AL', 'AL',
'AL', 'AL', 'AL', 'NL', 'AL', 'NL', 'NL', 'AL', 'AL', 'NL', 'AL', 'NL', 'AL',
'AL', 'NL', 'AL', 'NL', 'NL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL',
'AL', 'AL', 'NL', 'AL', 'NL', 'AL', 'AL', 'AL', 'NL', 'AL', 'AL', 'NL', 'AL']
```

Count the totals of NL and AL

- Loop through each winning division in the array
- Update a count for each division accordingly

```
In [13]: # Create variables
         total_al = 0
         total_nl = 0

         # Loop through each division in the winning_division_array, update wins
         for division in winning_division_array:
             if division == "AL":
                 total_al = total_al + 1
             else:
                 total_nl = total_nl + 1

         # Display results
         print("Number NL wins: " + str(total_nl))
         print("Number AL wins: " + str(total_al))
```

```
Number NL wins: 303
Number AL wins: 697
```

What else can you learn from this data? In the markdown cell below, list at least 2 questions that you would be able to answer by further analyzing the data. (Note: you do not need to actually answer the questions).

1. How many games does it usually take to win the world series?
2. What is the ratio of the wins between Al and NL?