Pascal's Triangle Documentation

Problem Statement:

Given an integer numRows, the task is to return the first numRows of Pascal's triangle. In Pascal's triangle, each number is the sum of the two numbers directly above it.

Example:

For numRows = 5, the output should be:

[

[1],

[1, 1],

[1, 2, 1],

[1, 3, 3, 1],

[1, 4, 6, 4, 1]

]

For numRows = 1, the output should be:

[

[1]

]

Constraints:

- 1 <= numRows <= 30

Solution Explanation

Class and Method:

- The solution is implemented in a class Solution with a method generate(numRows: int) -> List[List[int]].

Algorithm:

1. **Initialization:**

- Start by initializing the result list triangle with the first row of Pascal's triangle, which is [1].

2. **Generating Rows:**

- *For each subsequent row from 1 to numRows 1:*
  - Initialize the current row with the first element as 1.
  - Retrieve the previous row from the triangle.
  - Compute the middle elements of the current row by summing adjacent elements from the previous row.
  - Append 1 to the end of the current row.
  - Add the current row to the triangle.

3. **Return the Result:**

- After constructing all the required rows, return the triangle.

Example Usage:

- **To generate the first 5 rows of Pascal's triangle:**

  ```
  solution = Solution()
  print(solution.generate(5))
  ```

- **To generate the first row of Pascal's triangle:**

  ```
  solution = Solution()
  print(solution.generate(1))
  ```

By following this approach, you can efficiently generate Pascal's triangle for any given number of rows within the specified constraint.