Day-10

Python DSA

https://github.com/im-amit-kumar/Python-DSA/tree/main/Day-10

Pascal Triangle's

https://leetcode.com/problems/pascals-triangle/

```
class Solution:
    def generate(self, numRows: int) -> List[List[int]]:
        pascal_triangle= [[1]]

        for i in range(1, numRows):
            new_row=[1]
            for j in range(1,i):
                prev_left = pascal_triangle[i-1][j-1]
                prev_right= pascal_triangle[i-1][j]
                sum_of_elements = prev_left + prev_right
                new_row.append(sum_of_elements)

            new_row.append(1)

            pascal_triangle.append(new_row)
        return pascal_triangle
```

Dry run Code for Jupyter notebook/Visual Studio

from typing import List

class Solution: def generate(self, numRows: int) -> List[List[int]]: # Initialize the first row of Pascal's Triangle pascal_triangle = [[1]] print(f"Row 0: {pascal_triangle[0]}")

```
   # Generate each row of Pascal's Triangle
    for i in range(1, numRows):
        print(f"\nGenerating row {i}...")
        # The first element of each row is always 1
        new_row = [1]
        print(f"  Start new_row with [1]")
        # Calculate the intermediate elements of the row
        for j in range(1, i):
            print("Current Pascal Triangle",pascal_triangle)
            prev_left = pascal_triangle[i - 1][j - 1]
            prev_right = pascal_triangle[i - 1][j]
            sum_of_elements = prev_left + prev_right
            new_row.append(sum_of_elements)
            print(f"  new_row[{j}] = {prev_left} + {prev_right} = 
{sum_of_elements}")

        # The last element of each row is also always 1
        new_row.append(1)
        print(f"  End new_row with 1: {new_row}")

        # Append the newly generated row to Pascal's Triangle
        pascal_triangle.append(new_row)
        print(f"Row {i}: {new_row}")

    # Return the completed Pascal's Triangle
    print(f"\nFinal Pascal's Triangle with {numRows} rows:")
    for row in pascal_triangle:
        print(row)
    return pascal_triangle
```

sol = Solution() sol.generate(5)

```
Row 0: [1]

Generating row 1...

 Start new_row with [1]

 End new_row with 1: [1, 1]

Row 1: [1, 1]
```

```
Generating row 2...

  Start new_row with [1]
  new_row[1] = 1 + 1 = 2
  End new_row with 1: [1, 2, 1]
Row 2: [1, 2, 1]

Generating row 3...

  Start new_row with [1]
  new_row[1] = 1 + 2 = 3
  new_row[2] = 2 + 1 = 3
  End new_row with 1: [1, 3, 3, 1]
Row 3: [1, 3, 3, 1]

Generating row 4...
  Start new_row with [1]
  new_row[1] = 1 + 3 = 4
  new_row[2] = 3 + 3 = 6
  new_row[3] = 3 + 1 = 4
  End new_row with 1: [1, 4, 6, 4, 1]
```

Row 4: [1, 4, 6, 4, 1]

Final Pascal's Triangle with 5 rows:

```
[1]

[1, 1]

[1, 2, 1]

[1, 3, 3, 1]

[1, 4, 6, 4, 1]
```

Time Complexity- O(N^2)

Space Complexity – O(N^2)