Pascal's-Triangle

May 14, 2020

0.0.1 Problem Statement

Find and return the `nth` row of Pascal's triangle in the form a list. `n` is 0-based.

For exmaple, if `n = 4`, then `output = [1, 4, 6, 4, 1]`.

To know more about Pascal's triangle: https://www.mathsisfun.com/pascals-triangle.html

```
In [4]: def nth_row_pascal(n):
            """
            :param: - n - index (0 based)
            return - list() representing nth row of Pascal's triangle
            """
```

Hide Solution

```
In [ ]: # Solution

        '''
        Points to note:
        1. We have to return a list.
        2. The elements of n^th row are made up of elements of (n-1)^th row. This comes up till
        3. Except for the first and last element, any other element at position `j` in the curre
        4. Be careful about the edge cases, example, an index should never be a NEGATIVE at any
        '''


        def nth_row_pascal(n):

            if n == 0:
                return [1]

            current_row = [1] # First row

            ''' Loop from 1 to n; `i` denotes the row number'''
            for i in range(1, n + 1):
                # Set the `current_row` from previous iteration as the `previous_row`
                previous_row = current_row

                # Let's build the fresh current_row gradually
                current_row = [1] # add the default first element at the 0^th index of the `i^th

                '''Loop from 1 to (i-1); `j` denotes the index of an element with in the `i^th`
                # Example, for 5th row we have considered n=4,
                # we will iterate index from 1 to 3, because
                # the default element at the 0^th index has already been added
                for j in range(1, i):

                    # An element at position `j` in the current row is the
                    # sum of elements at position `j` and `j-1` in the previous row.
                    next_number = previous_row[j] + previous_row[j - 1]

                    # Append the new element to the current_row
                    current_row.append(next_number)

                current_row.append(1) # append the default last element
            return current_row
```

```
In [18]: def test_function(test_case):
             n = test_case[0]
             solution = test_case[1]
             output = nth_row_pascal(n)
             if solution == output:
                 print("Pass")
             else:
                 print("Fail")
```

```
In [19]: n = 0
         solution = [1]

         test_case = [n, solution]
         test_function(test_case)
```

```
Pass
```

```
In [20]: n = 1
         solution = [1, 1]

         test_case = [n, solution]
         test_function(test_case)
```

```
Pass
```

```
In [21]: n = 2
         solution = [1, 2, 1]

         test_case = [n, solution]
         test_function(test_case)
```

```
Pass
```

```
In [22]: n = 3
         solution = [1, 3, 3, 1]

         test_case = [n, solution]
         test_function(test_case)
```

```
Pass
```

```
In [23]: n = 4
         solution = [1, 4, 6, 4, 1]

         test_case = [n, solution]
         test_function(test_case)
```

```
Pass
```