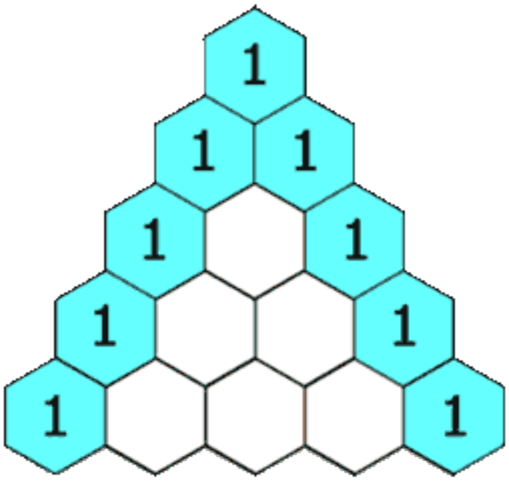
Leetcode Problem 1. (Easy)

Pascal's Triangle II

Given an integer rowIndex, return the rowIndexth (**0-indexed**) row of the **Pascal's triangle**.

In **Pascal's triangle**, each number is the sum of the two numbers directly above it as shown:



Example 1:

Input: rowIndex = 3
Output: [1,3,3,1]

Example 2:

Input: rowIndex = 0
Output: [1]

Example 3:

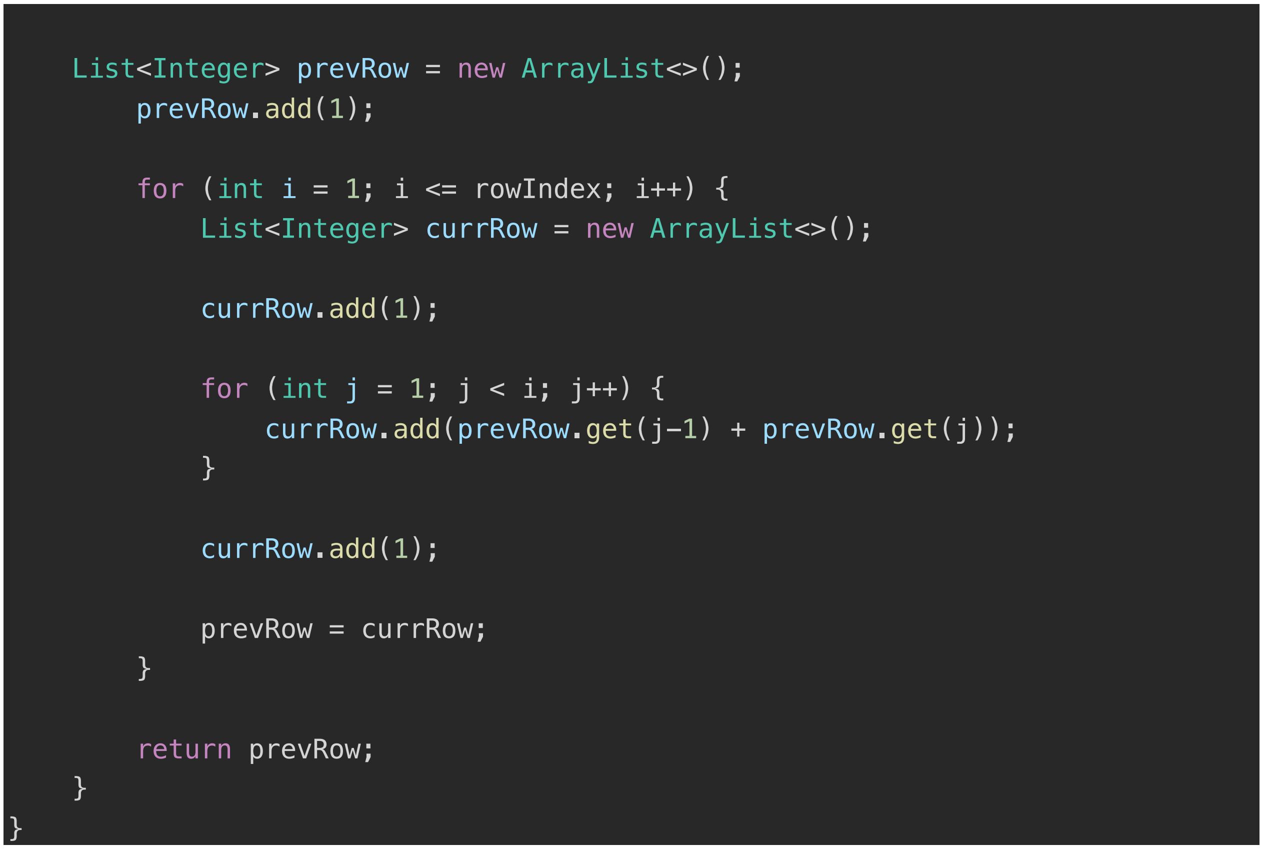
Input: rowIndex = 1
Output: [1,1]

Constraints:

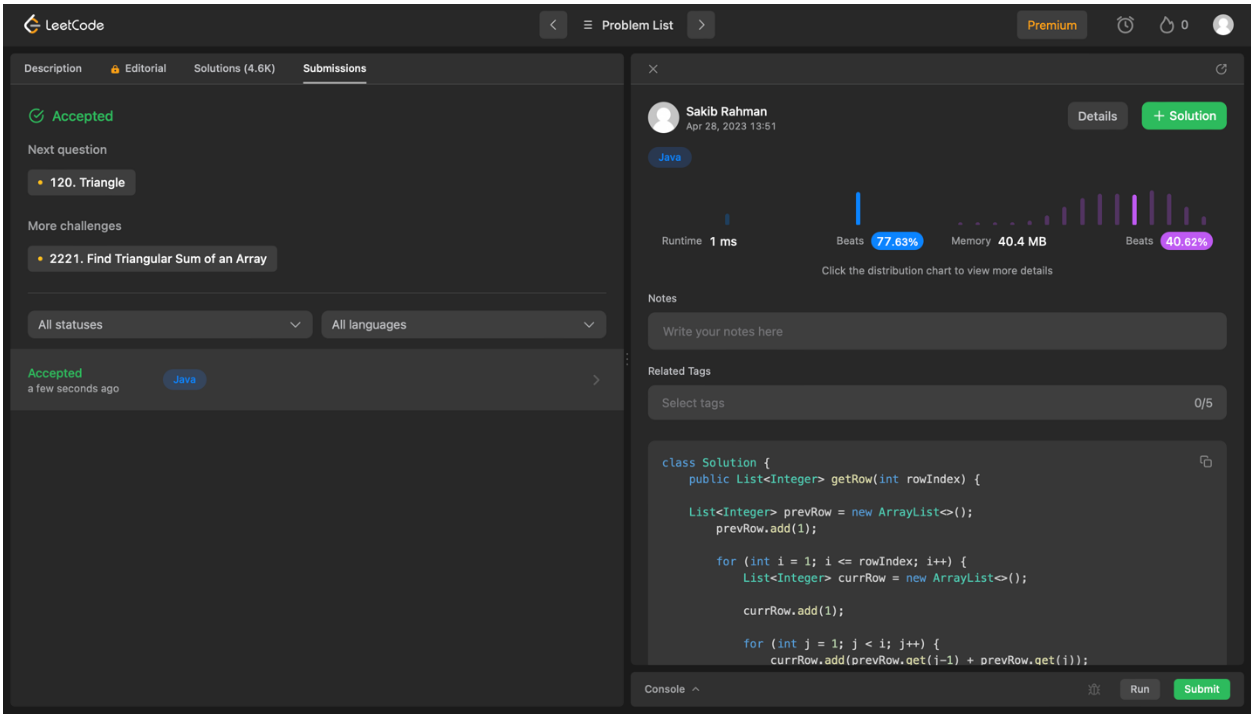
- 0 <= rowIndex <= 33

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

```
class Solution {
    public List<Integer> getRow(int rowIndex) {
```



```
    List<Integer> prevRow = new ArrayList<>();
        prevRow.add(1);

        for (int i = 1; i <= rowIndex; i++) {
            List<Integer> currRow = new ArrayList<>();

            currRow.add(1);

            for (int j = 1; j < i; j++) {
                currRow.add(prevRow.get(j-1) + prevRow.get(j));
            }

            currRow.add(1);

            prevRow = currRow;
        }

        return prevRow;
    }
}
```



Accepted

Next question

- 120. Triangle

More challenges

- 2221. Find Triangular Sum of an Array

All statuses | All languages

Accepted
a few seconds ago — Java

Sakib Rahman
Apr 28, 2023 13:51

Runtime 1 ms Beats 77.63% Memory 40.4 MB Beats 40.62%

Click the distribution chart to view more details

Notes: Write your notes here

Related Tags: Select tags 0/5

```
class Solution {
    public List<Integer> getRow(int rowIndex) {

    List<Integer> prevRow = new ArrayList<>();
        prevRow.add(1);

        for (int i = 1; i <= rowIndex; i++) {
            List<Integer> currRow = new ArrayList<>();

            currRow.add(1);

            for (int j = 1; j < i; j++) {
                currRow.add(prevRow.get(j-1) + prevRow.get(j));
```