118. Pascal's Triangle

Easy

Given a non-negative integer *numRows*, generate the first *numRows* of Pascal's triangle.



In Pascal's triangle, each number is the sum of the two numbers directly above it.

**Example:**

**Input:** 5  
**Output:**  
[  
 [1],  
 [1,1],  
 [1,2,1],  
 [1,3,3,1],  
 [1,4,6,4,1]  
]

class Solution {

public:

vector<vector<int>> generate(int numRows) {

vector<vector<int>> ret;

if(numRows==0) return ret;

vector<int> prev\_vec;

prev\_vec.push\_back(1);

ret.push\_back(prev\_vec);

for(int i=1;i<numRows;i++){

prev\_vec=ret[i-1];

vector<int> curr\_vec;

curr\_vec.push\_back(1);

for(int j=0;j<=i-2;j++){

int a=prev\_vec[j];

int b=(prev\_vec.size()-1>=j)?prev\_vec[j+1]:prev\_vec[j];

curr\_vec.push\_back(a+b);

}

curr\_vec.push\_back(1);

ret.push\_back(curr\_vec);

//prev\_vec=curr\_vec;

}

return ret;

}

};

Success

[Details](https://leetcode.com/submissions/detail/211115019/)

Runtime: 4 ms, faster than 100.00% of C++ online submissions for Pascal's Triangle.

Memory Usage: 8.9 MB, less than 21.45% of C++ online submissions for Pascal's Triangle.