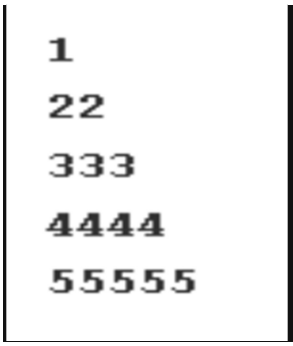
Pattern - 4: Right-Angled Number Pyramid - II

110

Problem Statement: Given an integer **N,** print the following pattern :



Here, N = 5.

Examples:

```
Input Format: N = 3
Result:
1
2 2
3 3 3

Input Format: N = 6
Result:
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
6 6 6 6 6 6
```

Solution

Disclaimer: *Don't jump directly to the solution, try it out yourself first.*

Problem Link

Approach:

There are 4 general rules for solving a pattern-based question :

- We always use nested loops for printing the patterns. For the outer loop, we count the number of lines/rows and loop for them.
- Next, for the inner loop, we focus on the number of columns and somehow connect them to the rows by forming a logic such that for each row we get the required number of columns to be printed.
- We print the '*' inside the inner loop.
- Observe symmetry in the pattern or check if a pattern is a combination of two or more similar patterns or not.

In this pattern, we run the outer loop for N times as we have to print N rows, and since we have to print a right-angled triangle/pyramid which must be upright, so the inner loop will run for the row number in each iteration. For eg: 1's for row 1, 5's 5 times for row 5, and so on. The only difference between this pattern and pattern 2 is that here we print numbers in each row instead of printing stars.

Code:

C++Java

```
#include <bits/stdc++.h>
using namespace std;

void pattern4(int N)
{
    // This is the outer loop which will loop for the rows.
    for (int i = 1; i <= N; i++)
    {
        // This is the inner loop which loops for the columns
       // no. of columns = row number for each line here.
       // Here, we print numbers equal to the row number
       // instead of stars in each row.
        for (int j = 1; j <=i; j++)
        {
            cout <<i<<" ";
        }
```

```
        // As soon as numbers for each iteration are printed, we move to the
        // next row and give a line break otherwise all numbers
        // would get printed in 1 line.
        cout << endl;
    }
}

int main()
{
    // Here, we have taken the value of N as 5.
    // We can also take input from the user.
    int N = 5;

    pattern4(N);

    return 0;
}
```

Output

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```