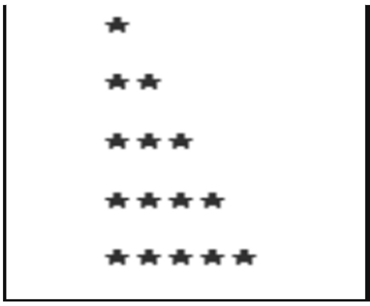
Pattern-2: Right-Angled Triangle Pattern

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



```
*
**
***
****
*****
```

Here, N = 5.

Examples:

```
Input Format: N = 3
Result:
*
* *
* * *

Input Format: N = 6
Result:
*
* *
* * *
* * * *
* * * * *
* * * * * *
```

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 problem, 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, the inner loop will run for the row number in each iteration. For eg: 1 star for row 1, 5 stars for row 5, and so on.

Code:

C++Java

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

void pattern2(int N)
{
    // This is the outer loop which will loop for the rows.
    for (int i = 0; i < N; i++)
    {
        // This is the inner loop which loops for the columns
       // no. of columns = row number for each line here.
        for (int j = 0; j <=i; j++)
        {
            cout << "* ";
        }

        // As soon as stars for each iteration are printed, we move
to the
        // next row and give a line break otherwise all stars
        // 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;

    pattern2(N);

    return 0;
}
```

Output

```
*
* *
* * *
* * * *
* * * * *
```