# Basic approach towards solving a pattern question

Solving pattern-related questions using loops in C++ usually involves using nested loops and manipulating the loop variables in a specific way to create the desired pattern. The basic approach for solving these types of problems is as follows:

- **Understand the problem:** Read the question carefully and try to understand what the desired pattern is. Make sure you understand the pattern before attempting to code it.
- **Analyze the pattern:** Break down the pattern into smaller parts and try to identify any common characteristics or relationships between the parts.
- **Design the solution:** Once you have a good understanding of the pattern, you can start designing the solution. Think about the different loops you will need and the values you will need to use for the loop variables.
- **Write the code:** Write the code that implements your solution, making sure to use proper indentation and spacing to make the code readable.
- **Test the code:** Test the code with different inputs and make sure it produces the desired output.
- **Optimize the code:** Once you have a working solution, see if there's any way to make it more efficient, for example, if you are using multiple loops see if you can optimize the innermost one.

Here is an example of solving a pattern-related question using loops in C++:

```
#include <iostream>

int main() {
    int rows = 5;
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j <= i; j++) {
            std::cout << "*";
        }
        std::cout << std::endl;
    }
```

```
    return 0;
}
```

**OUTPUT:**

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

Here is how it works:

- The outer loop, represented by the variable **i**, controls the number of rows in the pattern. It starts at 0 and goes up to **rows-1**.
- The inner loop, represented by the variable **j**, controls the number of asterisks in each row. It starts at 0 and goes up to **i**.
- The **cout** statement inside the inner loop is executed once for each value of **j**, printing an asterisk each time.
- The **endl** is used to move to the next line after the inner loop completes to print the next row of asterisks.

Note that this is just one example, the approach may vary depending on the pattern you want to solve. The key point is to analyze the pattern and design the solution accordingly, and then implement it efficiently.