

Sure, here are some Java programs to print different star patterns:

Print a right triangle pattern of stars:

csharp

Copy code

```
public class RightTrianglePattern {  
    public static void main(String[] args) {  
        int n = 5;  
        for (int i = 1; i <= n; i++) {  
            for (int j = 1; j <= i; j++) {  
                System.out.print("*");  
            }  
            System.out.println();  
        }  
    }  
}
```

Output:

markdown

Copy code

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

Print an inverted right triangle pattern of stars:

csharp

Copy code

```
public class InvertedRightTrianglePattern {  
    public static void main(String[] args) {  
        int n = 5;  
        for (int i = n; i >= 1; i--) {  
            for (int j = 1; j <= i; j++) {
```

```

        System.out.print("*");
    }
    System.out.println();
}
}
}

```

Output:

markdown

Copy code

```

*****
****
***
**
*

```

Print a pyramid pattern of stars:

csharp

Copy code

```

public class PyramidPattern {
    public static void main(String[] args) {
        int n = 5;
        for (int i = 1; i <= n; i++) {
            for (int j = n - i; j >= 1; j--) {
                System.out.print(" ");
            }
            for (int k = 1; k <= 2 * i - 1; k++) {
                System.out.print("*");
            }
            System.out.println();
        }
    }
}

```

Output:

markdown

Copy code

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

Print a diamond pattern of stars:

csharp

Copy code

```
public class DiamondPattern {  
    public static void main(String[] args) {  
        int n = 5;  
        for (int i = 1; i <= n; i++) {  
            for (int j = n - i; j >= 1; j--) {  
                System.out.print(" ");  
            }  
            for (int k = 1; k <= 2 * i - 1; k++) {  
                System.out.print("*");  
            }  
            System.out.println();  
        }  
        for (int i = n - 1; i >= 1; i--) {  
            for (int j = n - i; j >= 1; j--) {  
                System.out.print(" ");  
            }  
            for (int k = 1; k <= 2 * i - 1; k++) {  
                System.out.print("*");  
            }  
            System.out.println();  
        }  
    }  
}
```

```
}
```

Output:

markdown

Copy code

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

These are just a few examples of the many possible star patterns that can be printed using Java. The key to creating different patterns is to use nested loops and to adjust the number of stars and spaces printed on each line based on the desired pattern.

junayedrahaman500@gmail.com

explain the third one

Sure, here's an explanation of the Java program to print a pyramid pattern of stars:

csharp

Copy code

```
public class PyramidPattern {
    public static void main(String[] args) {
        int n = 5;
        for (int i = 1; i <= n; i++) {
            for (int j = n - i; j >= 1; j--) {
                System.out.print(" ");
            }
        }
    }
}
```

```

    }
    for (int k = 1; k <= 2 * i - 1; k++) {
        System.out.print("*");
    }
    System.out.println();
}
}
}

```

In this program, we first declare an integer variable `n` and assign it a value of 5. This variable represents the number of rows in the pyramid pattern.

Next, we use a for loop to iterate over each row of the pyramid. The outer loop uses the variable `i` to keep track of the current row number, starting from 1 and going up to `n`.

Inside the outer loop, we have two nested loops. The first nested loop uses the variable `j` to print spaces before the stars in each row. We start by printing `n - i` spaces, where `n` is the total number of rows and `i` is the current row number. This means that for the first row, we print 4 spaces, for the second row we print 3 spaces, and so on, until we print no spaces for the last row.

The second nested loop uses the variable `k` to print the stars in each row. We start by printing 1 star for the first row, 3 stars for the second row, 5 stars for the third row, and so on, until we print `2 * i - 1` stars for the last row. The formula `2 * i - 1` gives us the number of stars to print in each row based on the current row number.

After printing the spaces and stars for each row, we use `System.out.println()` to move to the next line and start the next row of the pyramid.

When we run this program, we get the following output:

markdown

Copy code

```

*
**
***
****

```

As you can see, the program has successfully printed a pyramid pattern of stars with 5 rows, where each row has an increasing number of stars and the stars are aligned to form a pyramid shape.

junayedrahaman500@gmail.com

explain the first and second one also

Sure, here are explanations for the first and second Java programs to print star patterns:

Print a right triangle pattern of stars:

csharp

Copy code

```
public class RightTrianglePattern {  
    public static void main(String[] args) {  
        int n = 5;  
        for (int i = 1; i <= n; i++) {  
            for (int j = 1; j <= i; j++) {  
                System.out.print("*");  
            }  
            System.out.println();  
        }  
    }  
}
```

In this program, we first declare an integer variable `n` and assign it a value of 5. This variable represents the number of rows in the triangle pattern.

Next, we use a for loop to iterate over each row of the triangle. The outer loop uses the variable `i` to keep track of the current row number, starting from 1 and going up to `n`.

Inside the outer loop, we have a nested loop that uses the variable `j` to print the stars in each row. We start by printing 1 star for the first row, 2 stars for the second row, 3 stars for the third row, and so on, until we print `i` stars for the last row. The loop condition is `j <= i`, which means that we print the same number of stars as the current row number.

After printing the stars for each row, we use `System.out.println()` to move to the next line and start the next row of the triangle.

When we run this program, we get the following output:

markdown

Copy code

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

As you can see, the program has successfully printed a right triangle pattern of stars with 5 rows, where each row has an increasing number of stars.

Print an inverted right triangle pattern of stars:

csharp

Copy code

```
public class InvertedRightTrianglePattern {  
    public static void main(String[] args) {  
        int n = 5;  
        for (int i = n; i >= 1; i--) {  
            for (int j = 1; j <= i; j++) {  
                System.out.print("*");  
            }  
            System.out.println();  
        }  
    }  
}
```

In this program, we use the same approach as in the previous program to print stars in each row. However, we start with the maximum number of stars in the first row (i.e., n stars) and then decrease the number of stars in each subsequent row until we print 1 star in the last row.

To achieve this, we use a for loop to iterate over each row of the inverted triangle. The outer loop uses the variable i to keep track of the current row number, starting from n and going down to 1.

Inside the outer loop, we have a nested loop that uses the variable j to print the stars in each row. We start by printing i stars for the first row, i-1 stars for the second row, i-2 stars for the third row, and so on, until we print 1 star for the last row. The loop condition is $j \leq i$, which means that we print the same number of stars as the current row number.

After printing the stars for each row, we use `System.out.println()` to move to the next line and start the next row of the inverted triangle.

When we run this program, we get the following output:

markdown

Copy code

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

As you can see, the program has successfully printed an inverted right triangle pattern of stars with 5 rows, where each row has a decreasing number of stars.