Problem solving & Pattern

Context

1. Problem solving
    - Definition
    - Importance
2. Patterns
    - Examples
    - Steps

PROBLEM SOLVING

Definition

Problem solving is the process of finding solutions to obstacles or challenges encountered in life or work. It is a crucial skill that allows individuals to tackle complex situations, adapt to changes, and overcome difficulties with ease

Steps:

1. Understanding the problem
2. Designing an algorithm
3. Implementing the solution

Importance

1. Enables software developers to identify, analyze, and solve complex problems.
2. Improving the efficiency and quality of the software.

Problem Solving Examples

1. Problem: Create a Function to Add Two Numbers:

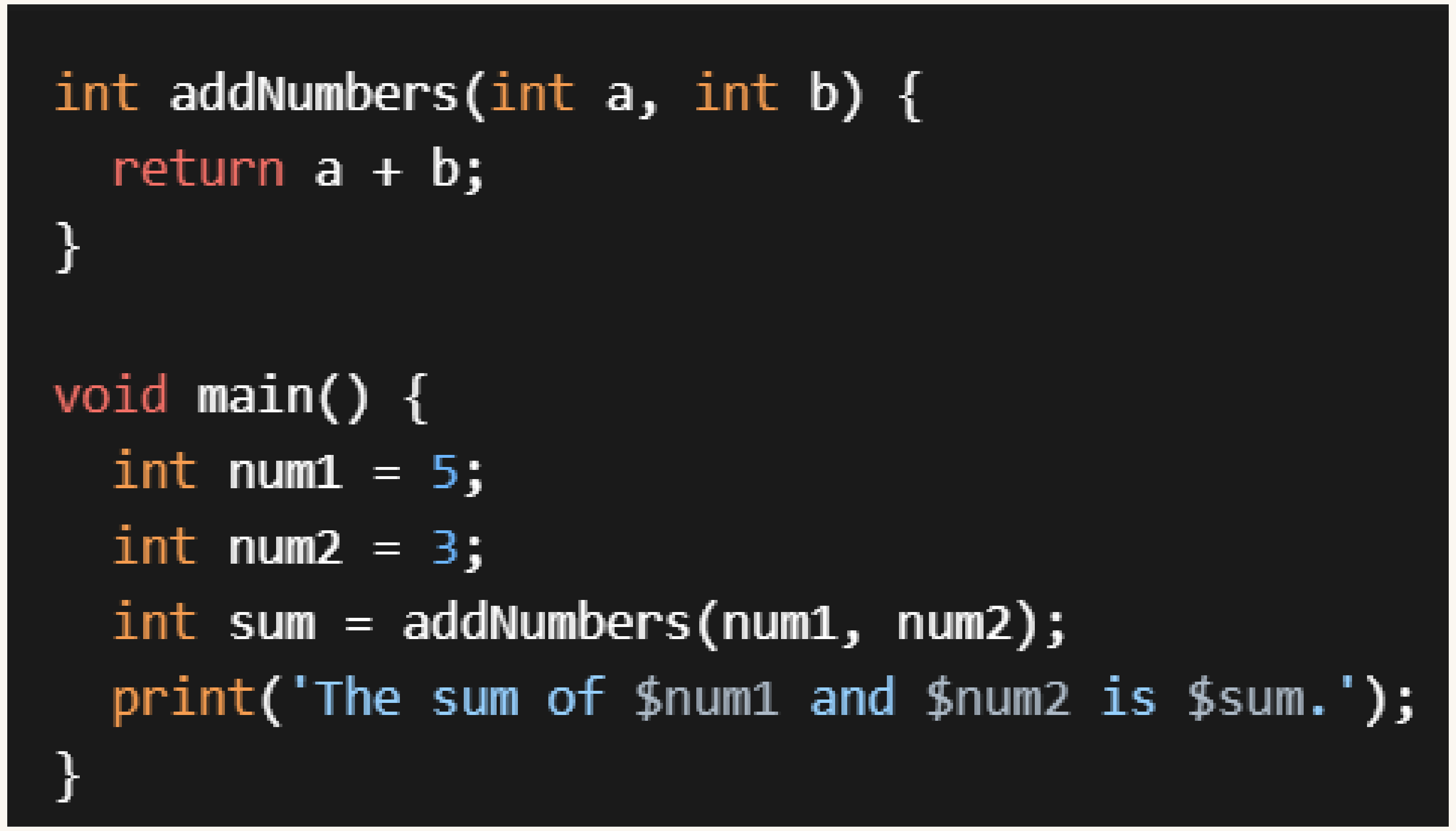
- Problem Statement: Write a function that takes two numbers as input and returns their sum.

- Step 1: Understand the Problem

  Before diving into code, let's ensure we fully understand the problem:
  - What are the inputs? We need two numbers.
  - What's the desired output? The sum of those two numbers.
- Step 2: Devise a Plan

  Now that we understand the problem, let's devise a plan:
  - Define a function that takes two parameters.
  - Inside the function, add the two numbers.
  - Return the result.

- Step 3: Implement the Solution



```
int addNumbers(int a, int b) {
  return a + b;
}

void main() {
  int num1 = 5;
  int num2 = 3;
  int sum = addNumbers(num1, num2);
  print('The sum of $num1 and $num2 is $sum.');
}
```

Output:- 8

- Step 4: Look Back

  After implementing the solution, we should test it with different inputs to ensure it works correctly. In this case, we've tested it with num1 = 5 and num2 = 3.

Problem Solving Examples

2. Problem: Sort User Input in Ascending or Descending Order

- Step 1: Understand the Problem

  Before diving into code, let's ensure we fully understand the problem:

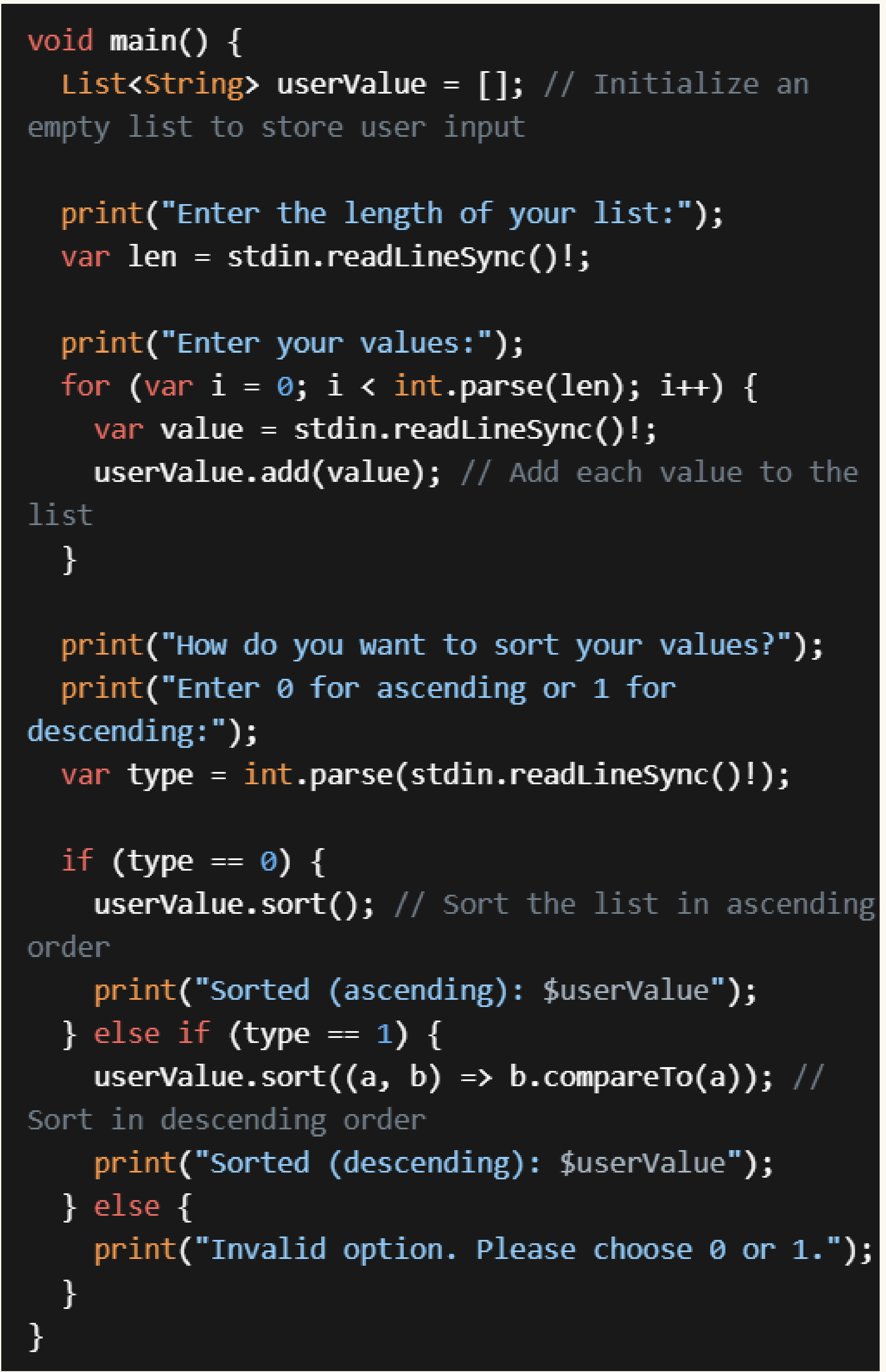
  - What are the inputs? We need Input from user.
  - What's the desired output? Maybe user value in ascending order or descending order.

- Step 2: Devise a Plan

  Now that we understand the problem, let's devise a plan:

  - You prompt the user to enter the length of the list.
  - Then, you collect the user's values one by one and store them in the list.
  - You ask the user whether they want the values sorted in ascending or descending order.
  - If the user chooses ascending, you sort the user Value list in ascending order.
  - If the user chooses descending, you sort the list descending order.
  - Then print the sorted list.

- ## Step 3: Implement the Solution



```
void main() {
  List<String> userValue = []; // Initialize an empty list to store user input

  print("Enter the length of your list:");
  var len = stdin.readLineSync()!;

  print("Enter your values:");
  for (var i = 0; i < int.parse(len); i++) {
    var value = stdin.readLineSync()!;
    userValue.add(value); // Add each value to the list
  }

  print("How do you want to sort your values?");
  print("Enter 0 for ascending or 1 for descending:");
  var type = int.parse(stdin.readLineSync()!);

  if (type == 0) {
    userValue.sort(); // Sort the list in ascending order
    print("Sorted (ascending): $userValue");
  } else if (type == 1) {
    userValue.sort((a, b) => b.compareTo(a)); // Sort in descending order
    print("Sorted (descending): $userValue");
  } else {
    print("Invalid option. Please choose 0 or 1.");
  }
}
```

Output:-

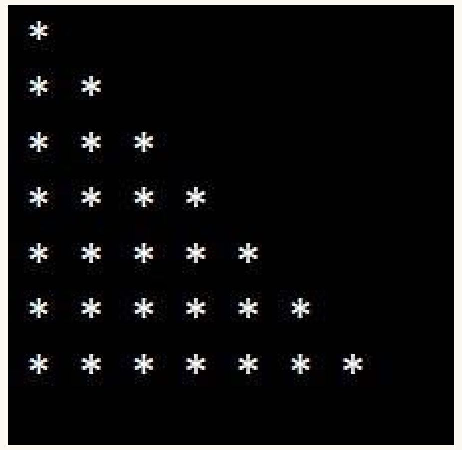
```
first enter your values length
3
Enter Your values
10
3
50
How you want your value in
ascending(option:0)
or
descending(option:1),
enter your option:
0
[3, 10, 50]
```

PATTERNS

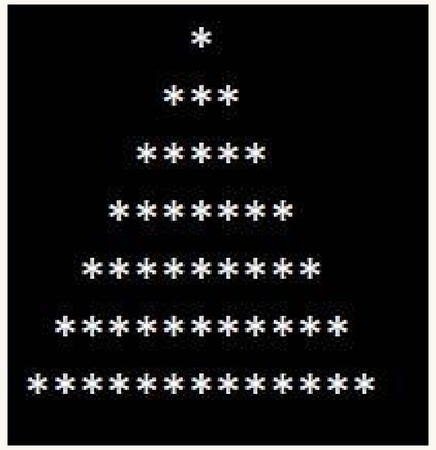
Patterns

Problem solving is the process of finding solutions to obstacles or challenges encountered in life or work. It is a crucial skill that allows individuals to tackle complex situations, adapt to changes, and overcome difficulties with ease

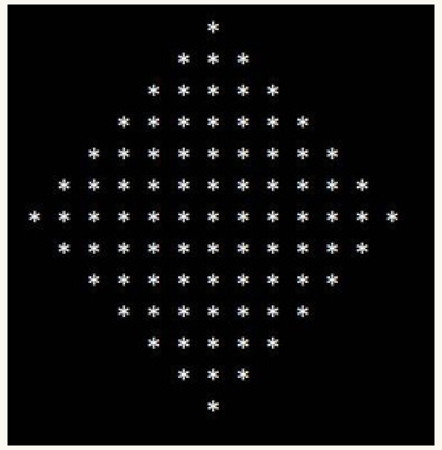
Examples



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



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



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

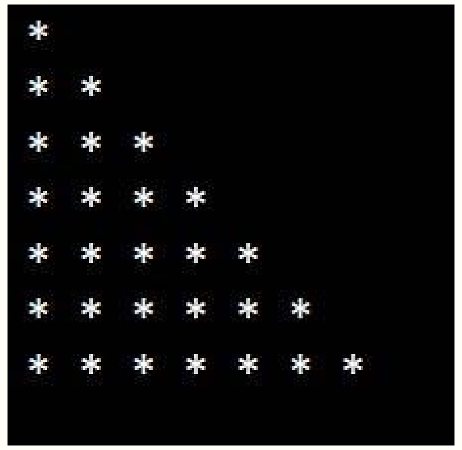
Steps:

1 Before starting the program we have to find the condition for printing star. so first we have to check there is space in prefix side. first pattern we are doing without prefix space.

2 Make all stars into row and column like table, then declare 3 variables

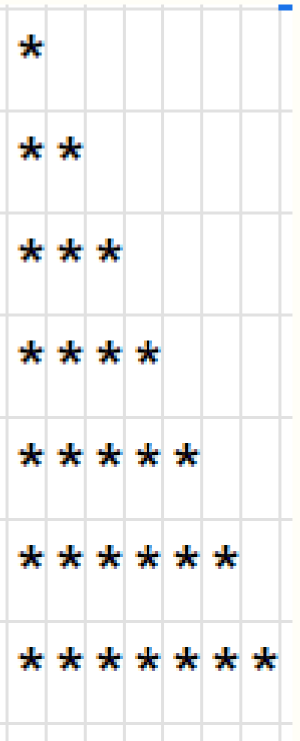
Now check how to easily solve pattern

1



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

2



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

3 variables

n,i,j

n: number of rows

i: rows index(starting with 0)

j; count of stars

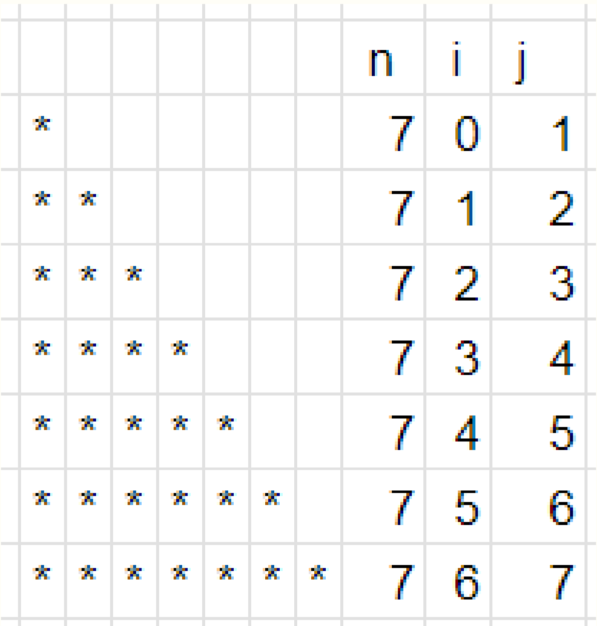
Steps:

3 Now assign all value to the table, n= number of rows, i= index of rows starting with 0, j = star count.

4 Now compare j variable with other and find the relation, it is the condition.

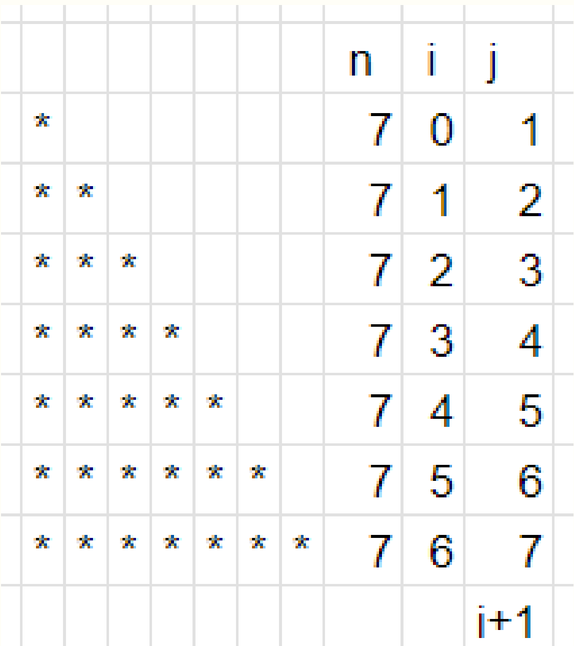
Now check how to easily solve pattern

3



							n	i	j
*							7	0	1
*	*						7	1	2
*	*	*					7	2	3
*	*	*	*				7	3	4
*	*	*	*	*			7	4	5
*	*	*	*	*	*		7	5	6
*	*	*	*	*	*	*	7	6	7

4



							n	i	j
*							7	0	1
*	*						7	1	2
*	*	*					7	2	3
*	*	*	*				7	3	4
*	*	*	*	*			7	4	5
*	*	*	*	*	*		7	5	6
*	*	*	*	*	*	*	7	6	7
									i+1

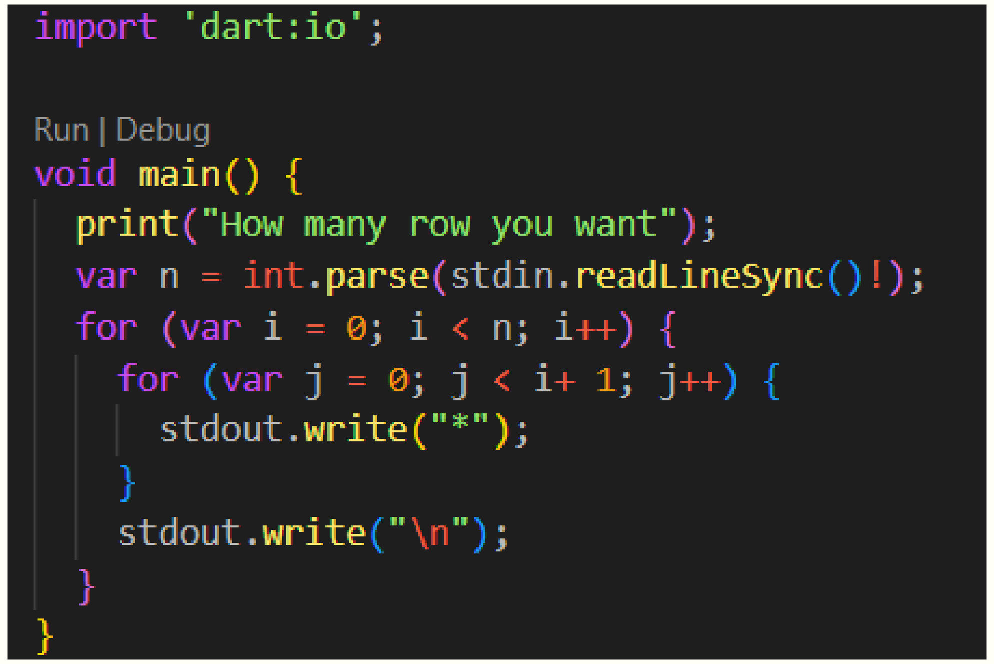
$i + 1 = j$

Steps:

5 Now apply the condition on code

Now check how to easily solve pattern

5

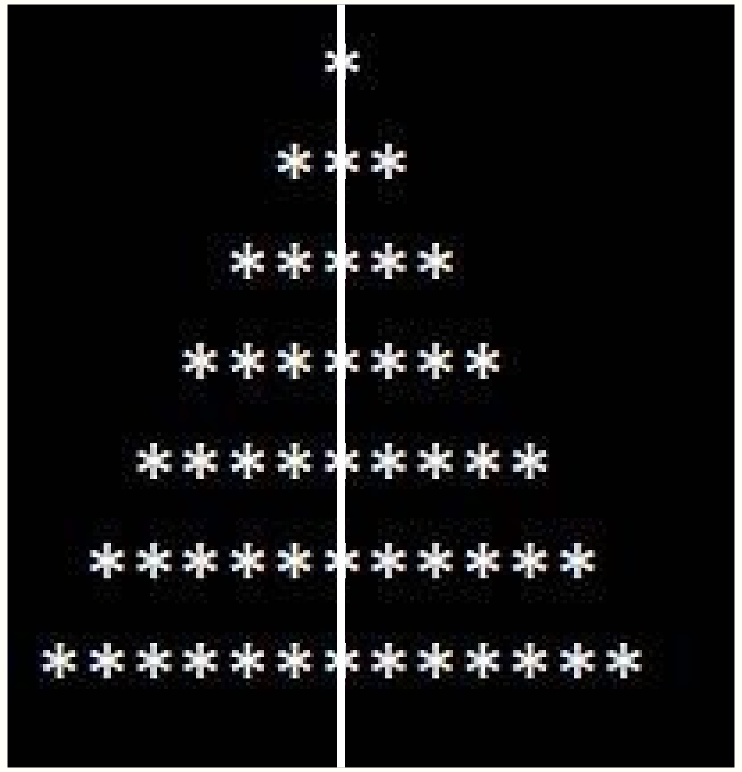


```
import 'dart:io';

Run | Debug
void main() {
  print("How many row you want");
  var n = int.parse(stdin.readLineSync()!);
  for (var i = 0; i < n; i++) {
    for (var j = 0; j < i+ 1; j++) {
      stdout.write("*");
    }
    stdout.write("\n");
  }
}
```

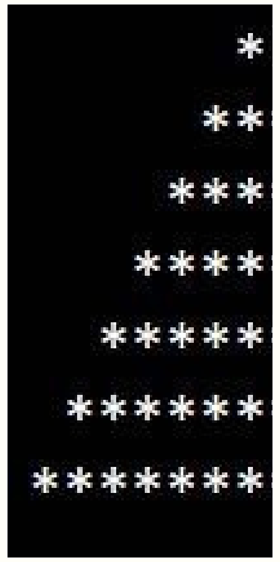
NOW WE CHECK WITH SPACE PATTERN

1

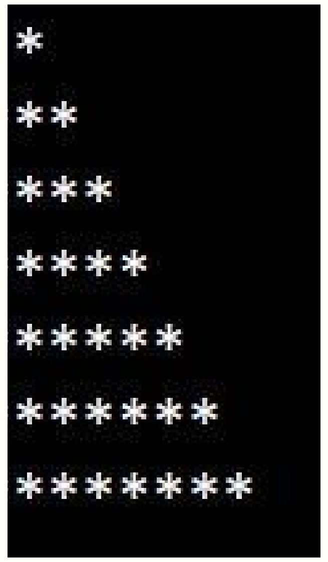


```
      *
     ***
    *****
   *******
  *********
 ***********
*************
```

2



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



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

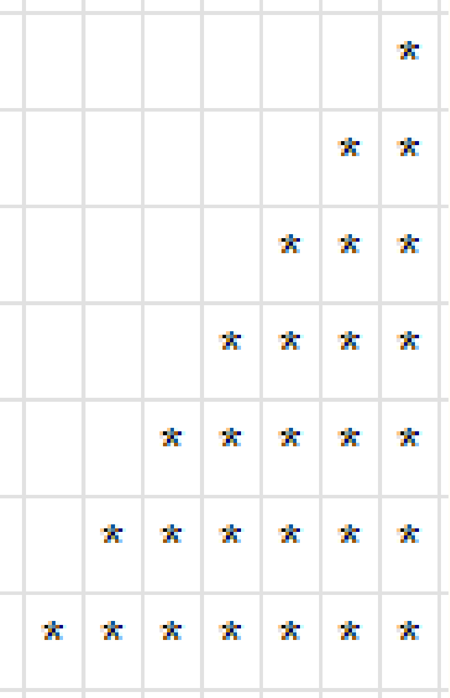
STEPS

1 In this time it is an full pyramid, it have an prefix space, when we get an image we have to check the pattern carefully, we can see a pattern inside a pattern, for this we have to cut the pattern to symmetrically.

2 Now we got two patterns first print the first pattern and then print second pattern

NOW WE CHECK WITH SPACE PATTERN

3



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

4 variables

n,i,s,j

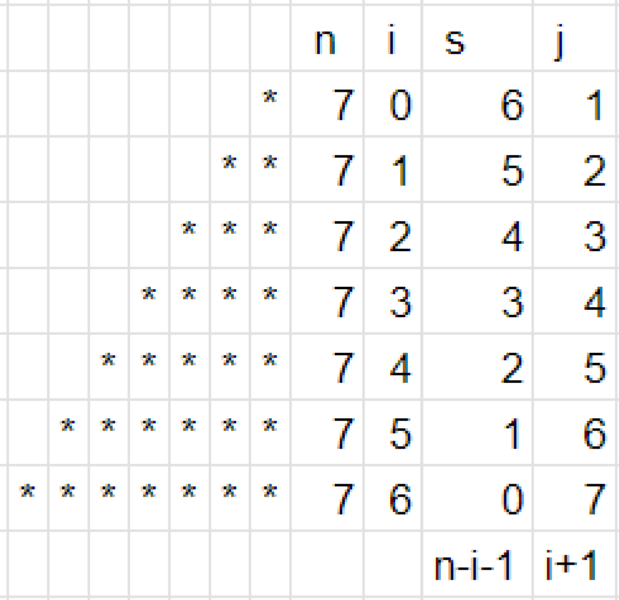
n: number of rows

i: rows index(starting with 0)

s: count of space

j; count of stars

4



							n	i	s	j
						*	7	0	6	1
					*	*	7	1	5	2
				*	*	*	7	2	4	3
			*	*	*	*	7	3	3	4
		*	*	*	*	*	7	4	2	5
	*	*	*	*	*	*	7	5	1	6
*	*	*	*	*	*	*	7	6	0	7
									n-i-1	i+1

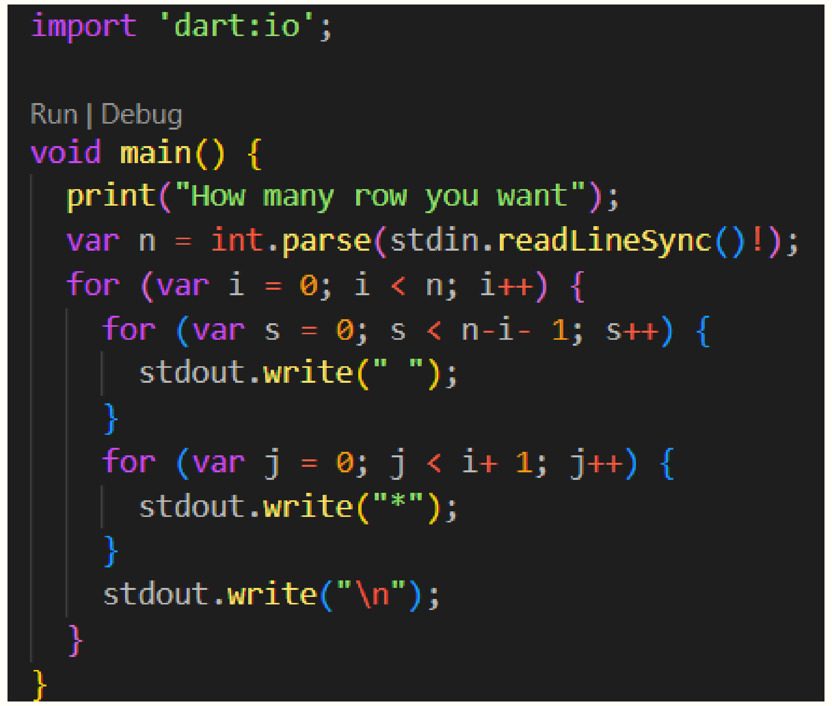
STEPS

3 Here in this pattern we have prefix space, so we have to take 4 variables.

4 Here we assign the value to each variables, and find the relation, here condition for s is n - i -1 =s ,so the condition is n-i-1 and i+1 =j so condition of j is i+1.

NOW WE CHECK WITH SPACE PATTERN

5



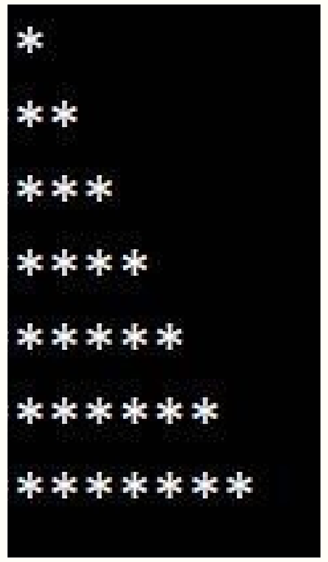
```
import 'dart:io';

Run | Debug
void main() {
  print("How many row you want");
  var n = int.parse(stdin.readLineSync()!);
  for (var i = 0; i < n; i++) {
    for (var s = 0; s < n-i- 1; s++) {
      stdout.write(" ");
    }
    for (var j = 0; j < i+ 1; j++) {
      stdout.write("*");
    }
    stdout.write("\n");
  }
}
```

Output:-

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

6



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

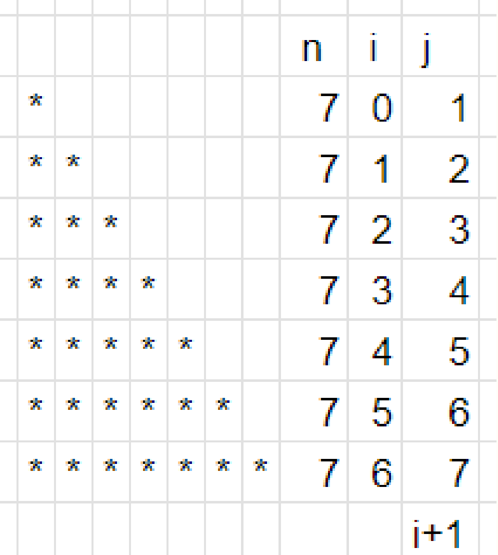
STEPS

5 Now apply the condition on code

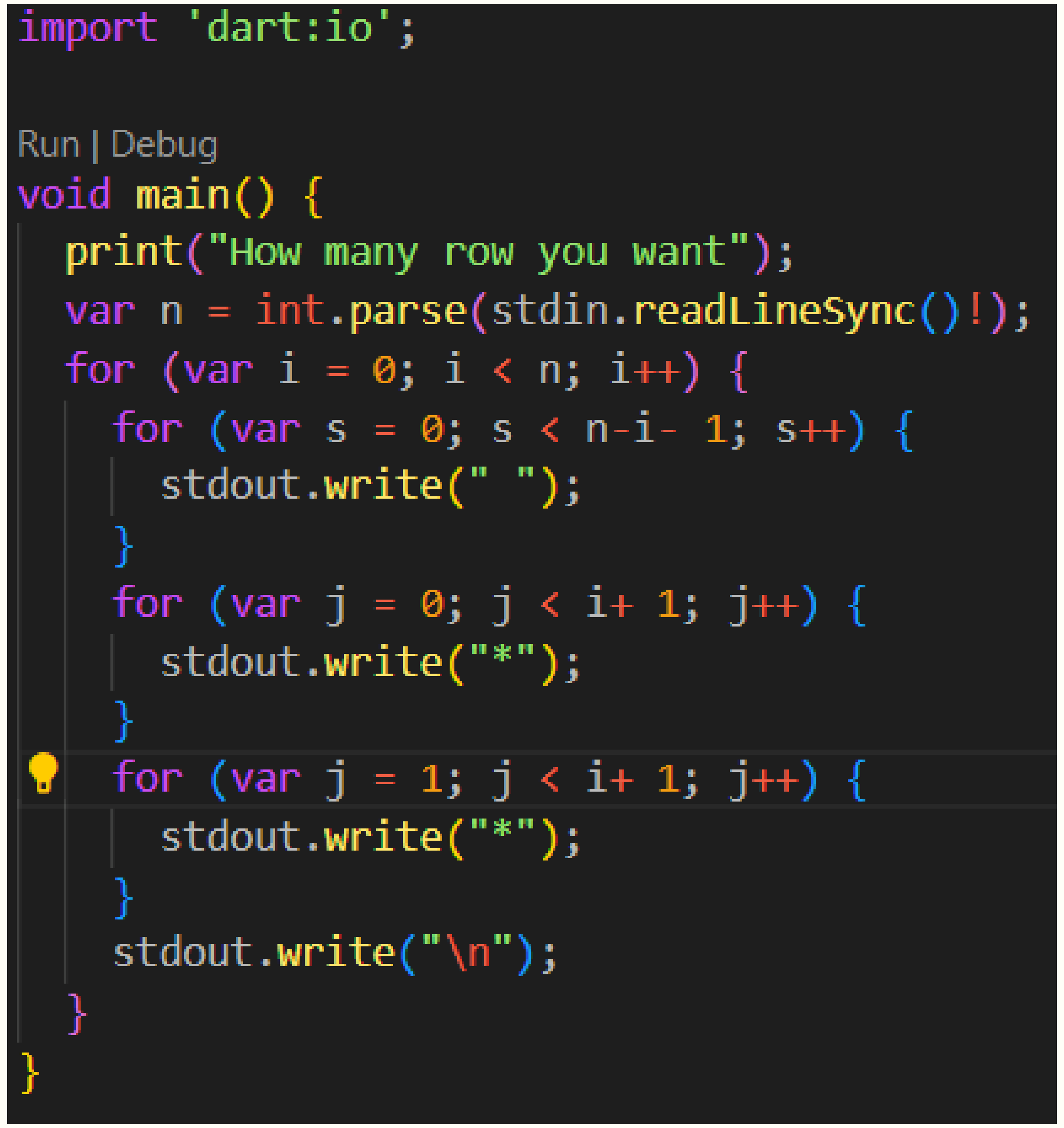
6 Next is to print another side of pyramid

NOW WE CHECK WITH SPACE PATTERN

7



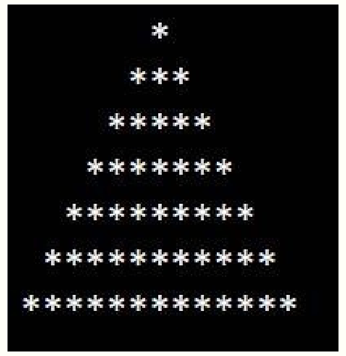
							n	i	j
*							7	0	1
*	*						7	1	2
*	*	*					7	2	3
*	*	*	*				7	3	4
*	*	*	*	*			7	4	5
*	*	*	*	*	*		7	5	6
*	*	*	*	*	*	*	7	6	7
									i+1



```
import 'dart:io';

Run | Debug
void main() {
  print("How many row you want");
  var n = int.parse(stdin.readLineSync()!);
  for (var i = 0; i < n; i++) {
    for (var s = 0; s < n-i- 1; s++) {
      stdout.write(" ");
    }
    for (var j = 0; j < i+ 1; j++) {
      stdout.write("*");
    }
    for (var j = 1; j < i+ 1; j++) {
      stdout.write("*");
    }
    stdout.write("\n");
  }
}
```

Output



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

STEPS

7

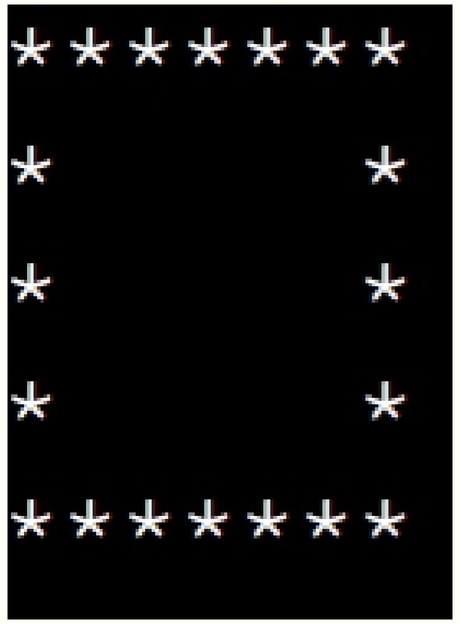
This is the condition and apply it to code, we want this pyramid also in same row, so we have to write the code before the new line, also the initial value is 1 because we don't need first line.

STEPS:

1. In this time it is an Hollow square, in here we have to print star some of place

2. check the pattern and check where we want to print the star, i=0,i=n-1,j=0,j=n-1,

3. Here 0th row ,n-1th row, 0th column, n-1th column have stars, others have space, so apply this condition when printing the star.



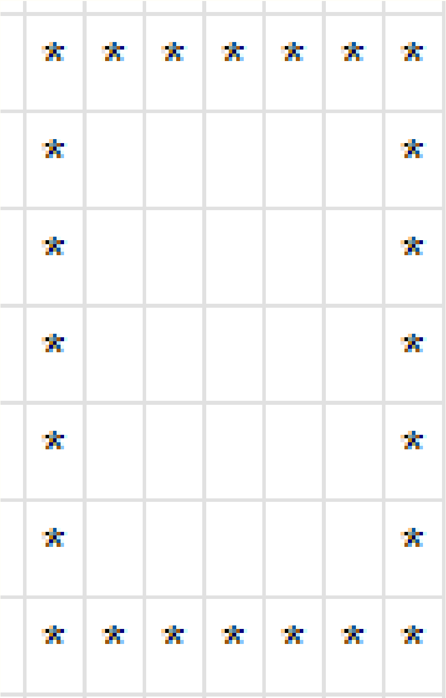
```
* * * * * * *

*           *

*           *

*           *

* * * * * * *
```

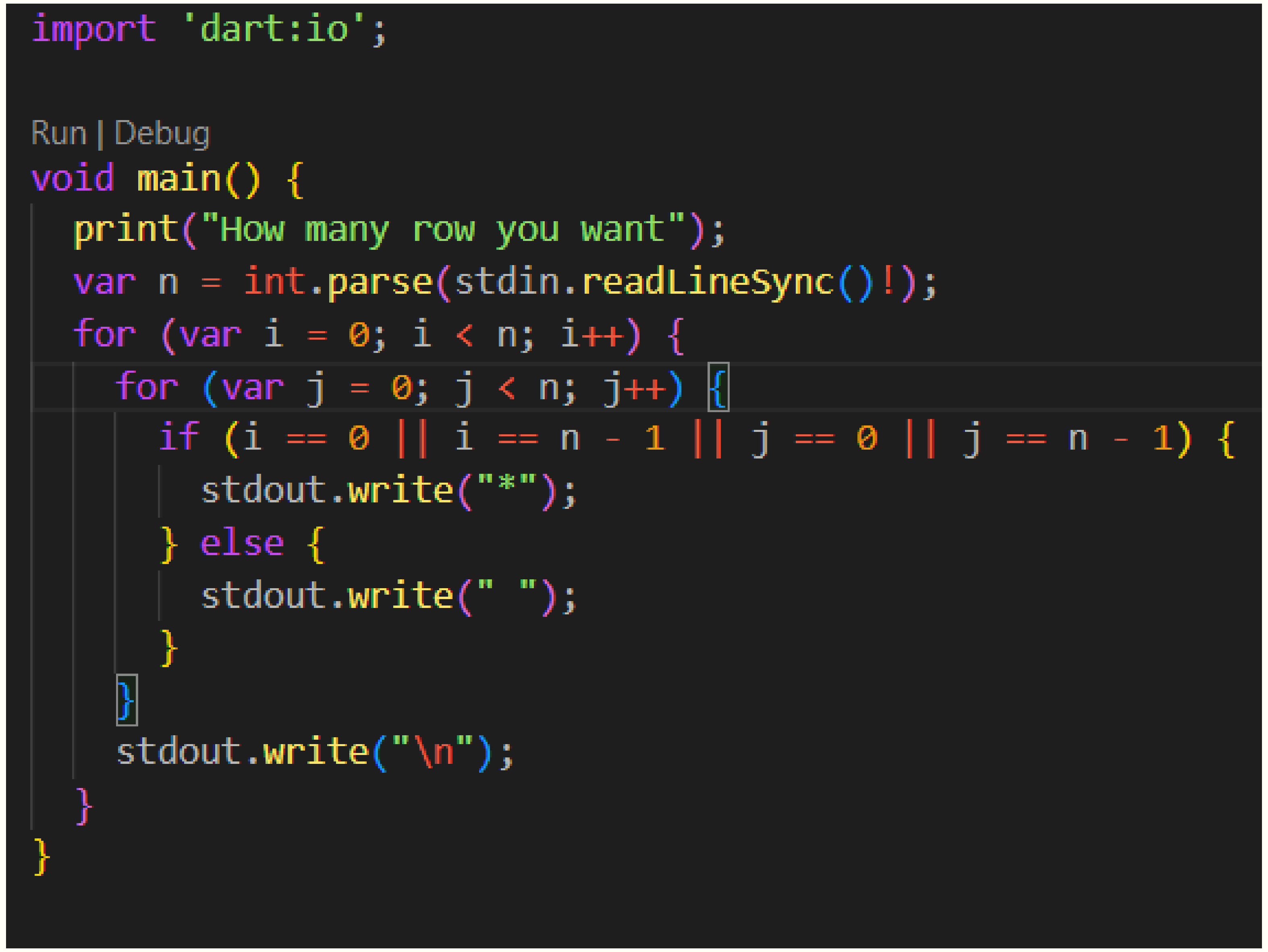


*	*	*	*	*	*	*
*						*
*						*
*						*
*						*
*						*
*	*	*	*	*	*	*

STEPS:

4 Apply the condition on code

5 After the code run we will get the output hollow square.



```
import 'dart:io';

void main() {
  print("How many row you want");
  var n = int.parse(stdin.readLineSync()!);
  for (var i = 0; i < n; i++) {
    for (var j = 0; j < n; j++) {
      if (i == 0 || i == n - 1 || j == 0 || j == n - 1) {
        stdout.write("*");
      } else {
        stdout.write(" ");
      }
    }
    stdout.write("\n");
  }
}
```

THANK YOU!