Looping 2

Programming Fundamentals Teaching Team 2023

Objectives

After studying this material, students should be able to:

- Understand the concept of nested loops in programming
- Get to know nested loop syntax
- Implement nested loops to solve the problem

Outlines

- Nested loop
- Case study

Preface

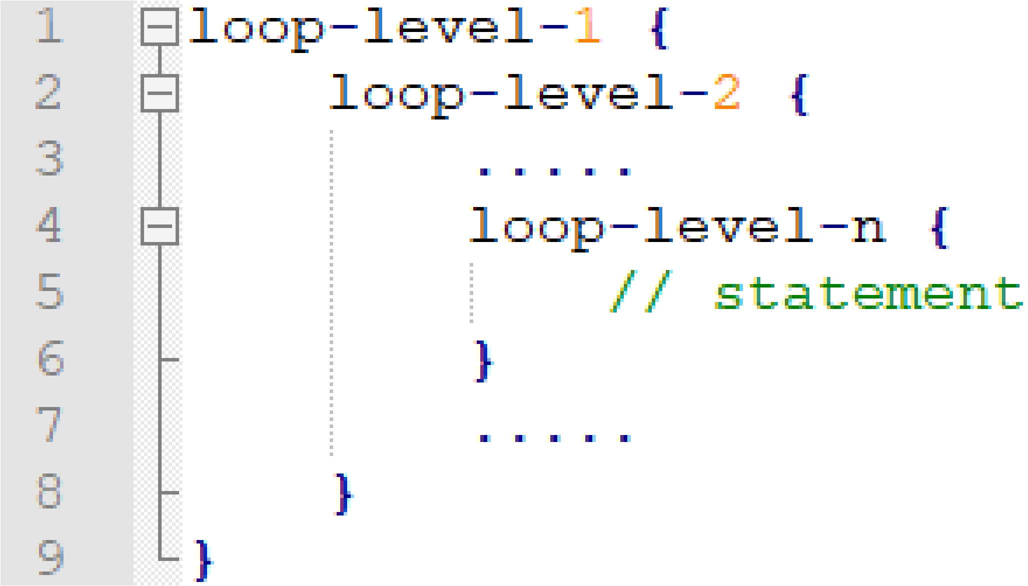
- In the basic concept of looping, looping logic is used to perform several **process / program** statements **repeatedly**, with a **certain pattern**.
- Process / statement will continue to be **executed** repeatedly, as long as **the loop condition is true**. Otherwise, the loop will **stop** and the process / statement will not be executed again when **the loop condition is false**.
- **A looping condition** is needed to determine whether a loop will continue or should stop.

Definition

- A nested loop is a loop structure that lies inside another loop.
- In a nested loop, there is an **outer loop** that is in the outermost position and an **inner loop** that is in its inner position.
- A nested loop can consist of more than 2 levels (minimum 2 levels).

Pseudocode

- In general, a nested loop should look like this:

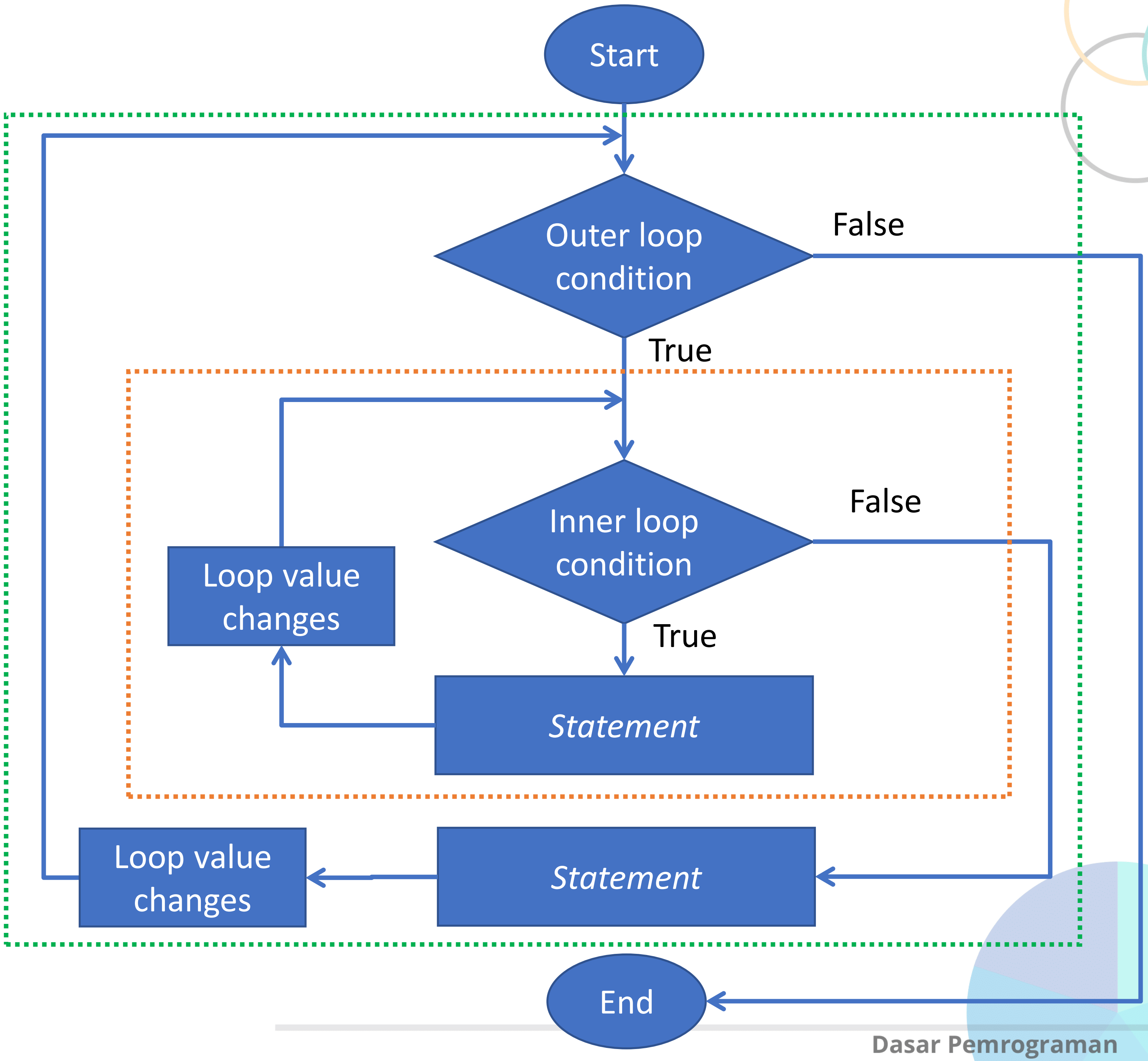


```
loop-level-1 {
    loop-level-2 {
        .....
        loop-level-n {
            // statement
        }
        .....
    }
}
```

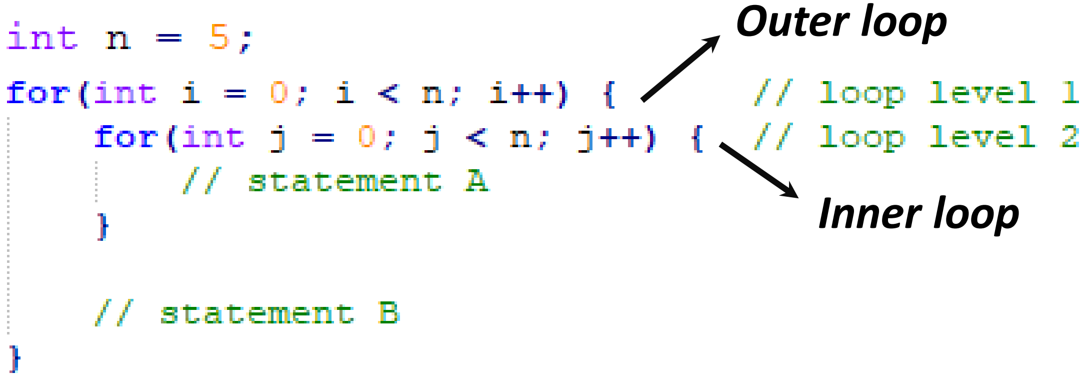
Flowchart

⬚ *Outer loop*

⬚ *Inner loop*



Nested Loop: FOR



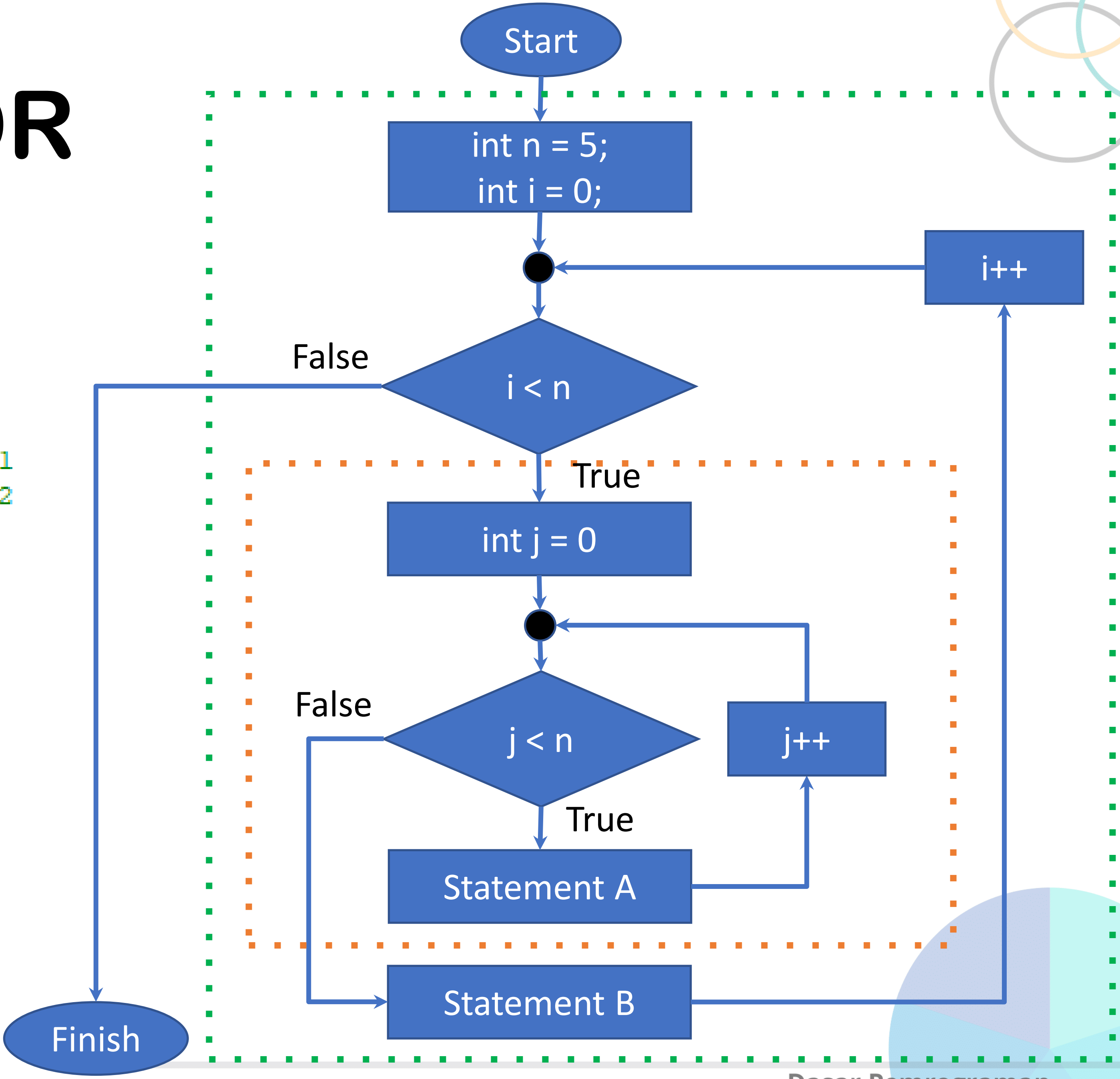
```
int n = 5;
for(int i = 0; i < n; i++) {     // loop level 1
    for(int j = 0; j < n; j++) { // loop level 2
        // statement A
    }

    // statement B
}
```

Outer loop

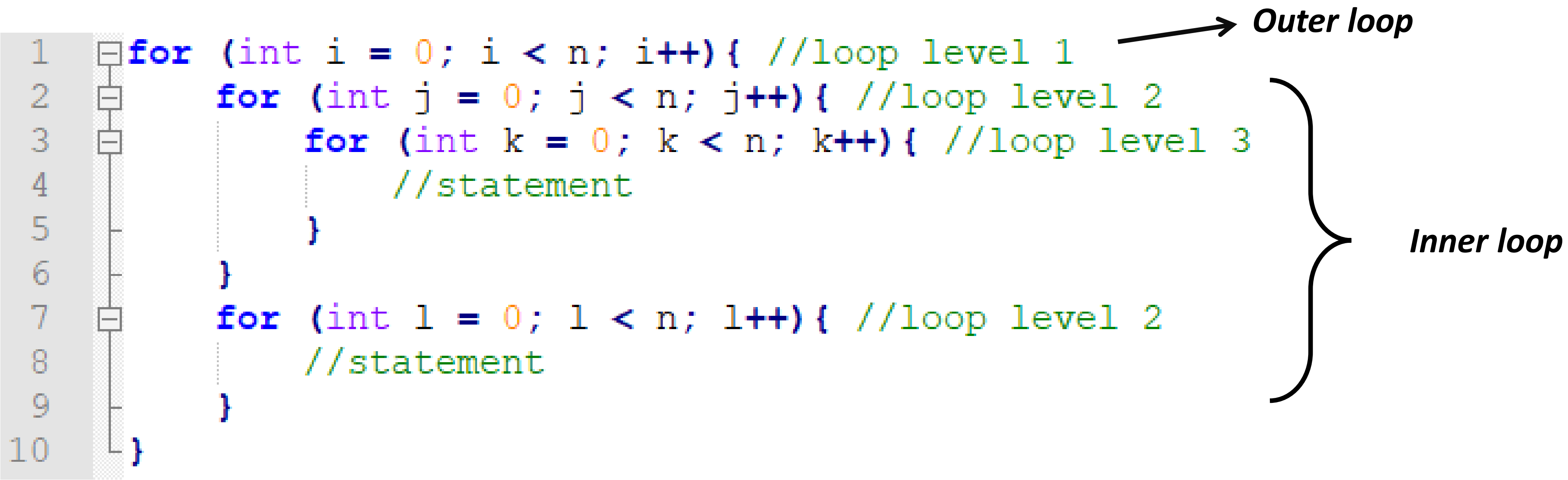
Inner loop

Outer loop

Inner loop



Nested Loop: FOR (more than 2 levels)



```
for (int i = 0; i < n; i++){ //loop level 1
    for (int j = 0; j < n; j++){ //loop level 2
        for (int k = 0; k < n; k++){ //loop level 3
            //statement
        }
    }
    for (int l = 0; l < n; l++){ //loop level 2
        //statement
    }
}
```

Outer loop

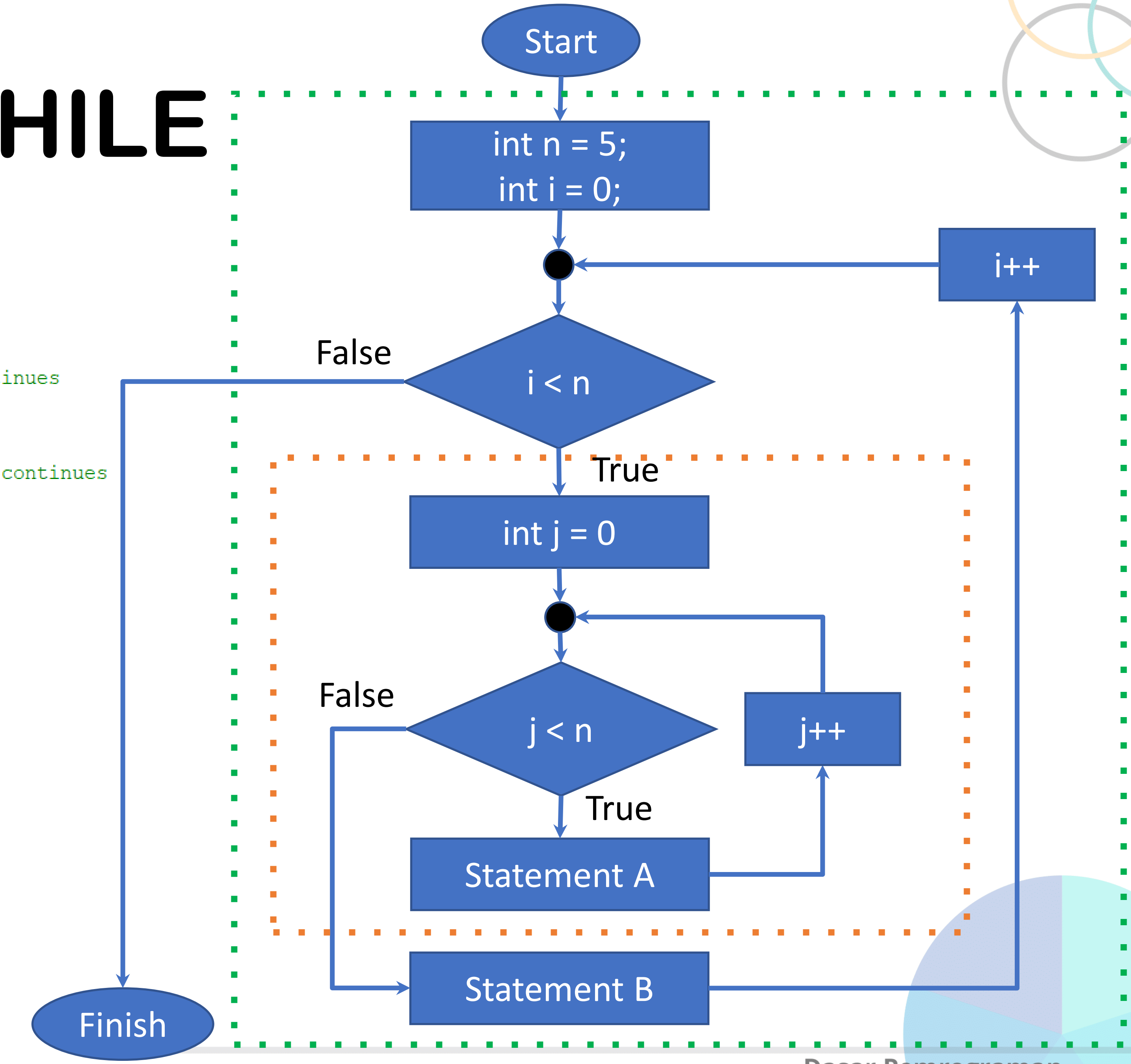
Inner loop

Nested Loop: WHILE

Outer loop

```
int n = 5;
int i = 0;
// Loop checking. As long as condition (i < n) is true, the loop continues
while (i < n) { //loop level 1
    int j = 0;

    // Loop checking. As long as condition (j < n) is true, the loop continues
    while (j < n) { //loop level 2
        // Statement A
        j++;
    }

    // Statement B
    i++;
}
```

Inner loop



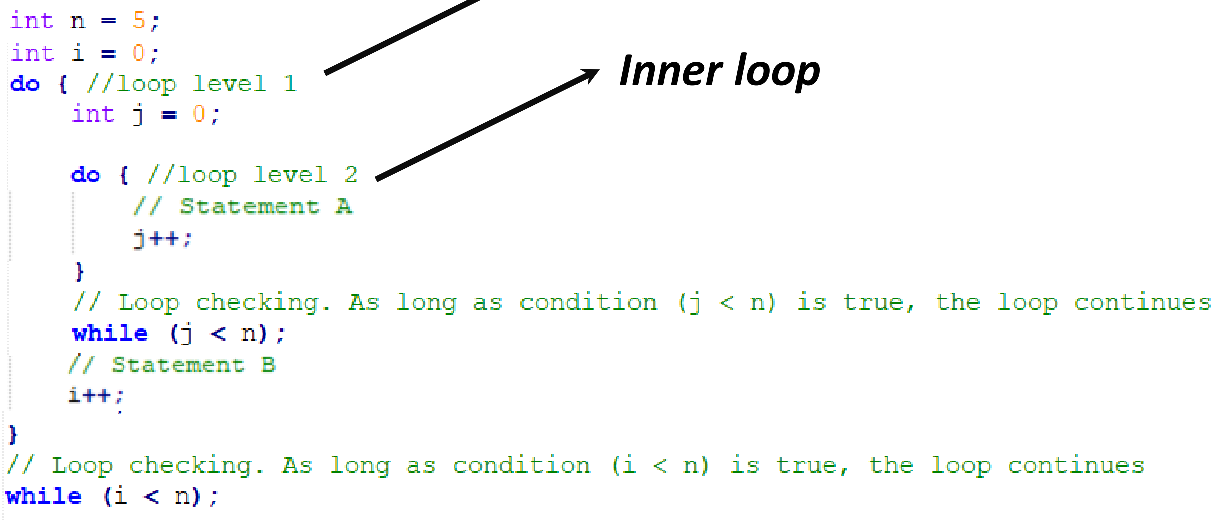
Outer loop

Inner loop

Nested Loop: DO-WHILE

Outer loop

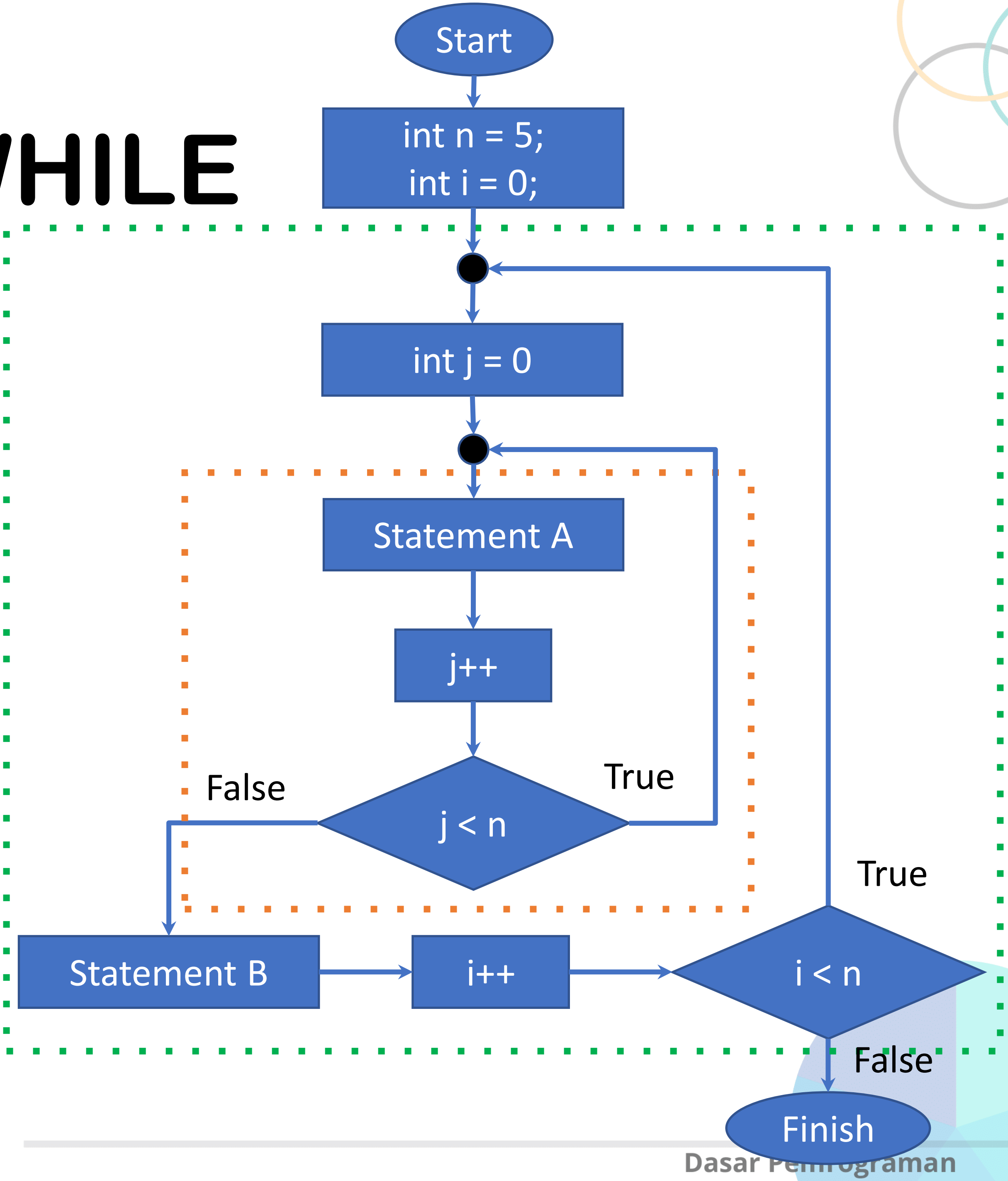
Inner loop



```
int n = 5;
int i = 0;
do { //loop level 1
    int j = 0;

    do { //loop level 2
        // Statement A
        j++;
    }
    // Loop checking. As long as condition (j < n) is true, the loop continues
    while (j < n);
    // Statement B
    i++;
}
// Loop checking. As long as condition (i < n) is true, the loop continues
while (i < n);
```

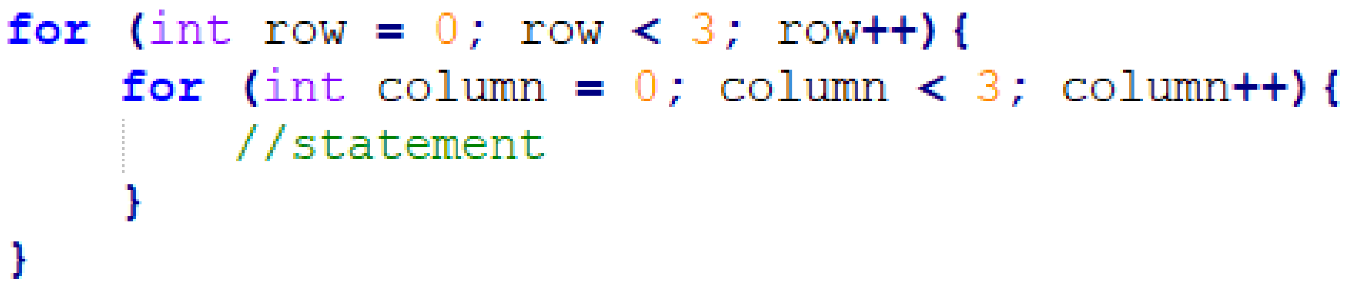
Outer loop

Inner loop

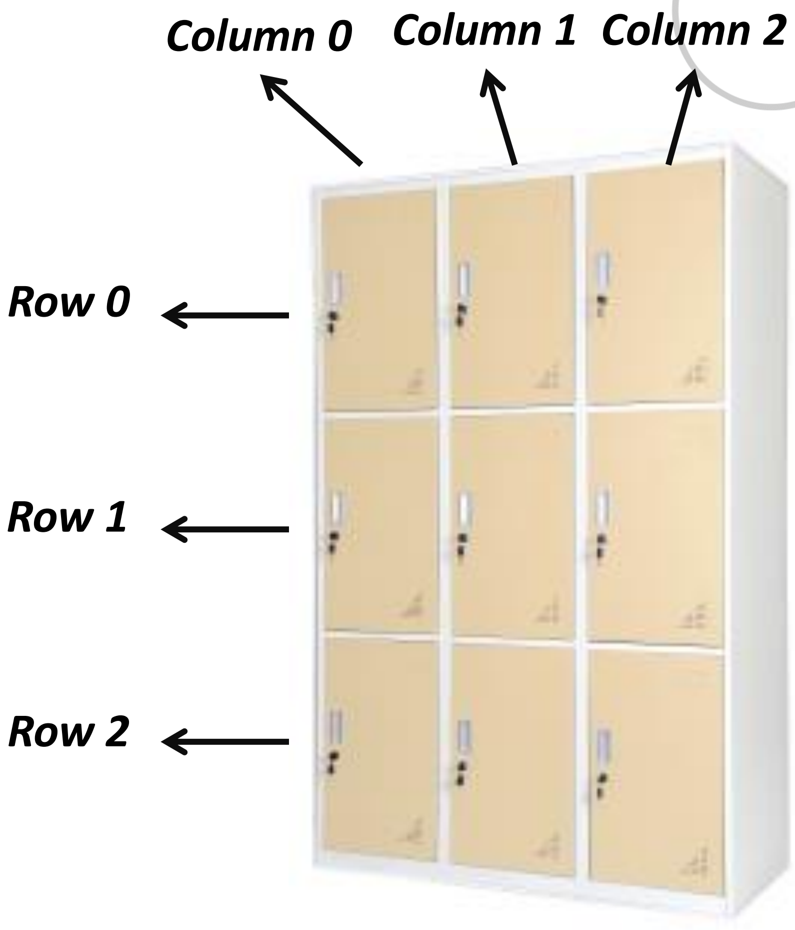


Locker Logic

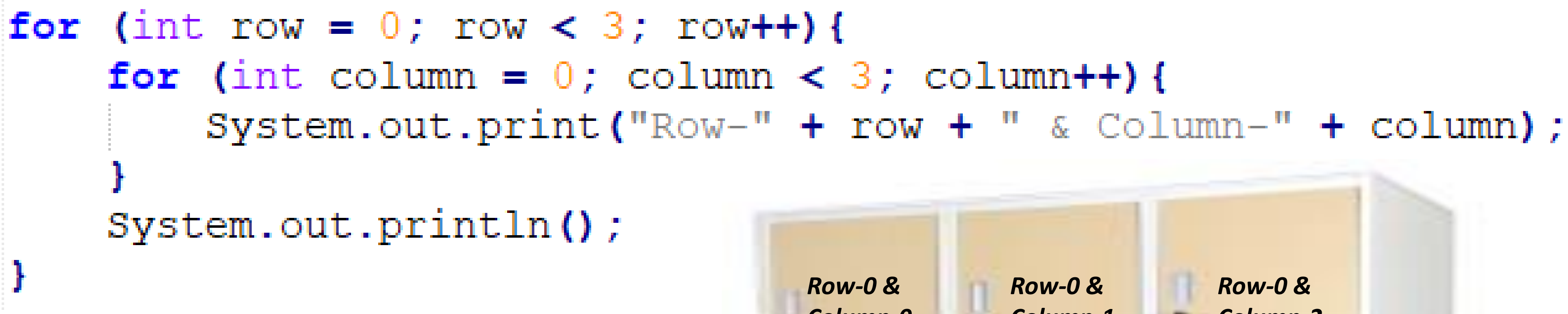
- A nested loop with 2 levels is like a locker.
- Where the **outer loop** is identified as a **row** pointer, and the **inner loop** is identified as a **column** pointer.



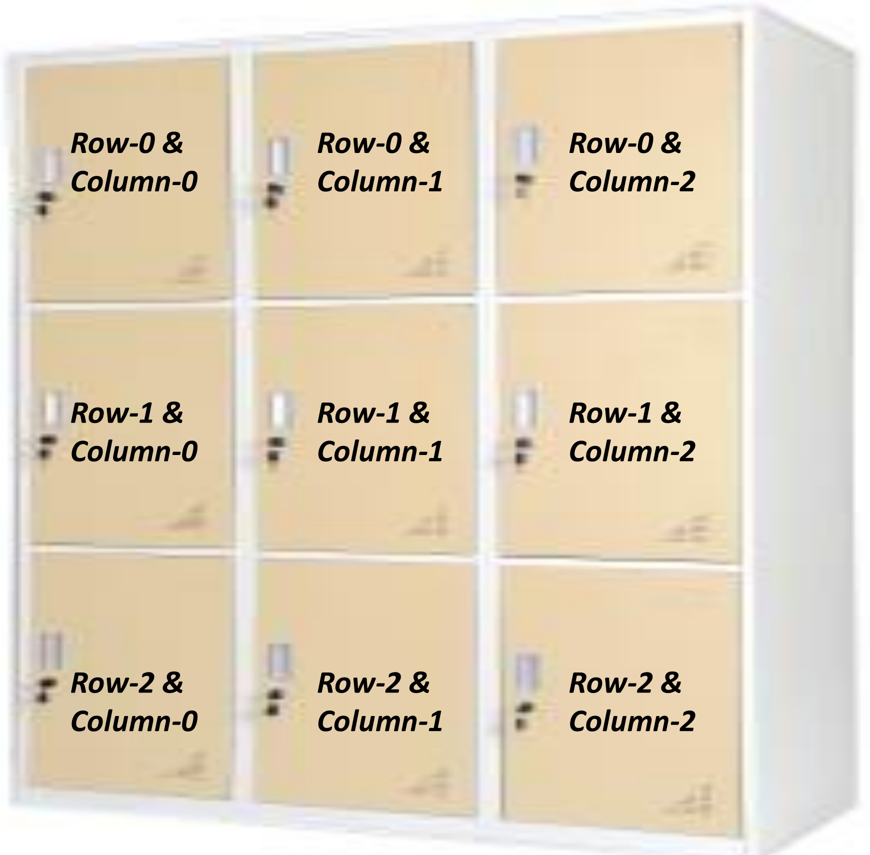
```
for (int row = 0; row < 3; row++){
    for (int column = 0; column < 3; column++){
        //statement
    }
}
```



Locker Logic

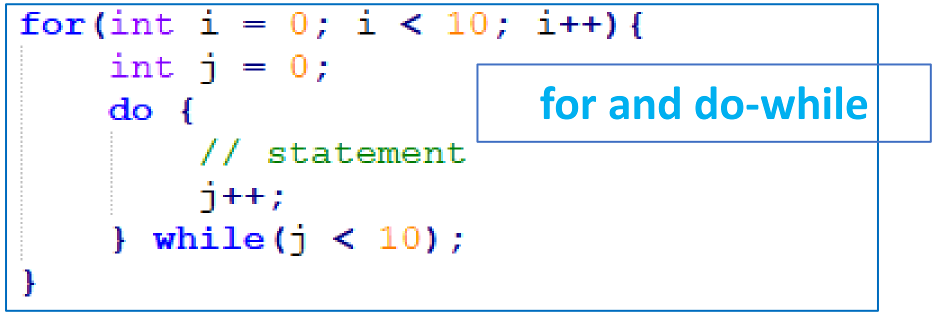


```java
for (int row = 0; row < 3; row++){
    for (int column = 0; column < 3; column++){
        System.out.print("Row-" + row + " & Column-" + column);
    }
    System.out.println();
}
```



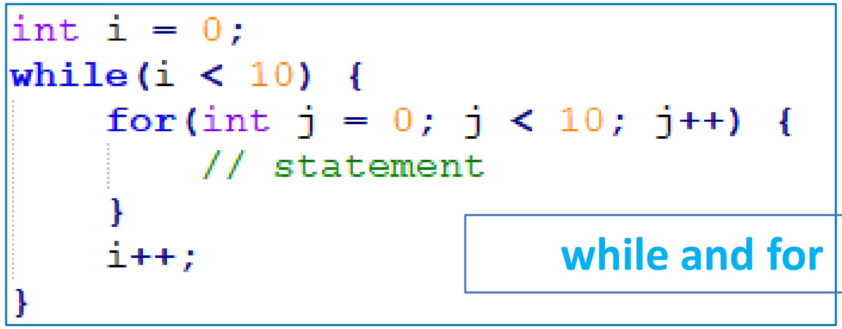
Nested Loop Combinations

- Nested loops not only consist of one type of nested loop, but it can be a combination of nested loops.



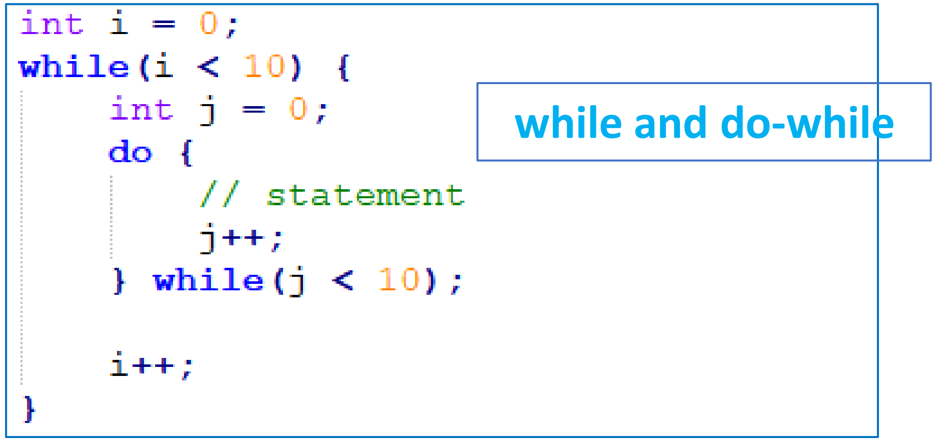
```
for(int i = 0; i < 10; i++){
    int j = 0;
    do {
        // statement
        j++;
    } while(j < 10);
}
```

for and do-while



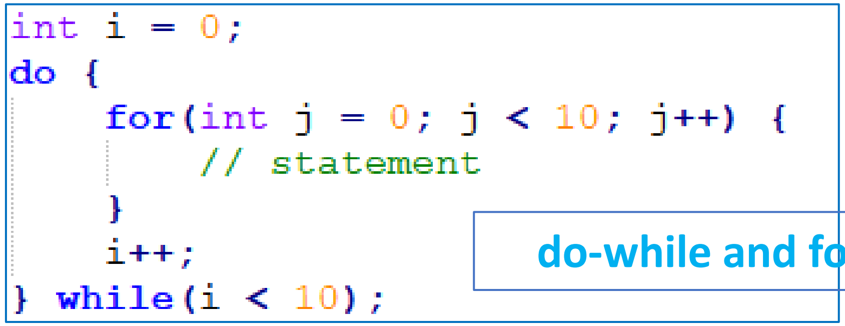
```
int i = 0;
while(i < 10) {
    for(int j = 0; j < 10; j++) {
        // statement
    }
    i++;
}
```

while and for



```
int i = 0;
while(i < 10) {
    int j = 0;
    do {
        // statement
        j++;
    } while(j < 10);

    i++;
}
```

while and do-while

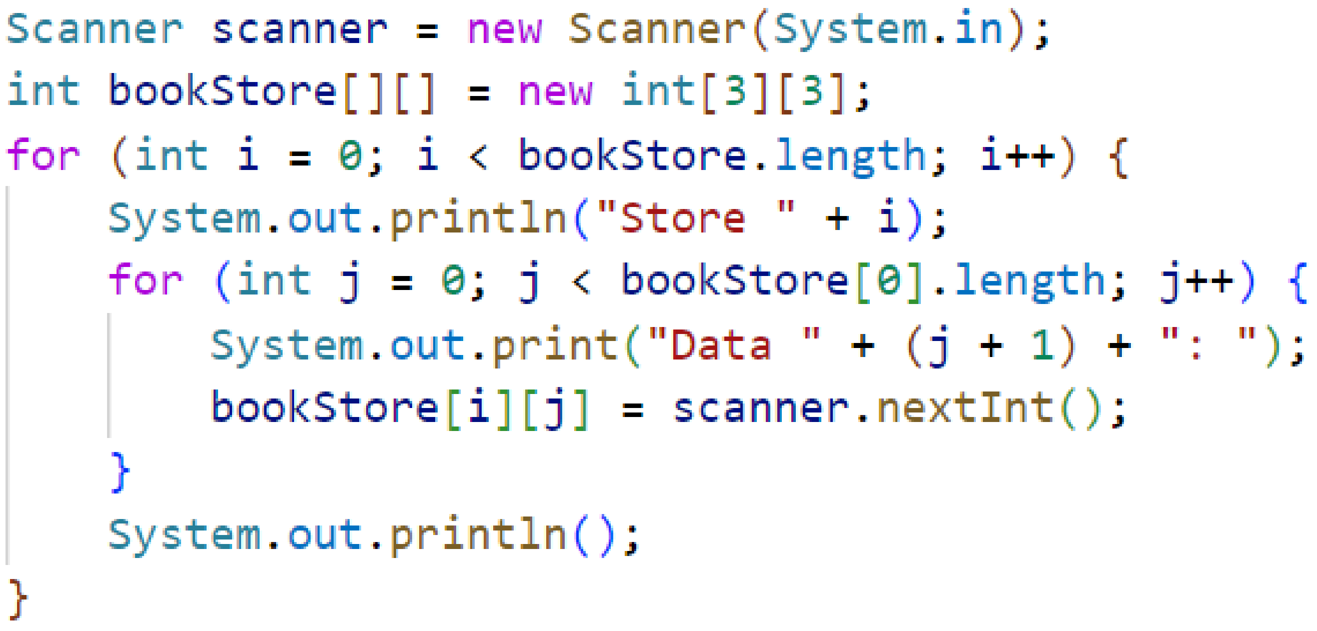


```
int i = 0;
do {
    for(int j = 0; j < 10; j++) {
        // statement
    }
    i++;
} while(i < 10);
```

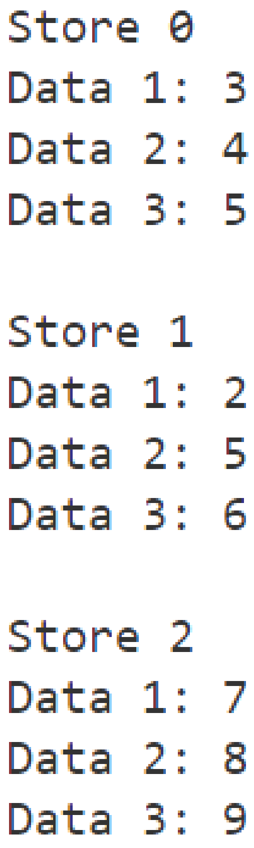
do-while and for

Nested Loops in Arrays

Nested loop to fill array values



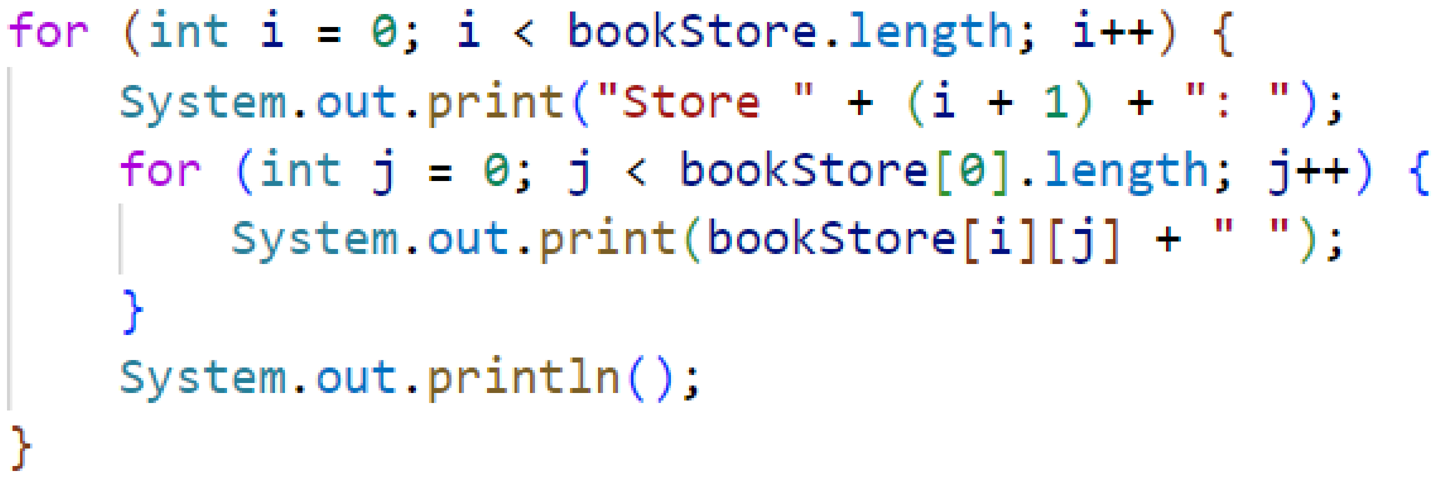
```
Scanner scanner = new Scanner(System.in);
int bookStore[][] = new int[3][3];
for (int i = 0; i < bookStore.length; i++) {
    System.out.println("Store " + i);
    for (int j = 0; j < bookStore[0].length; j++) {
        System.out.print("Data " + (j + 1) + ": ");
        bookStore[i][j] = scanner.nextInt();
    }
    System.out.println();
}
```



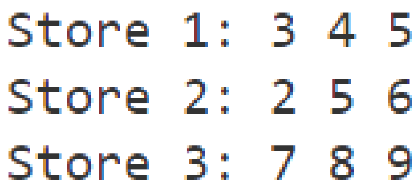
```
Store 0
Data 1: 3
Data 2: 4
Data 3: 5

Store 1
Data 1: 2
Data 2: 5
Data 3: 6

Store 2
Data 1: 7
Data 2: 8
Data 3: 9
```

Nested Loops in Arrays

Nested loop to display array values



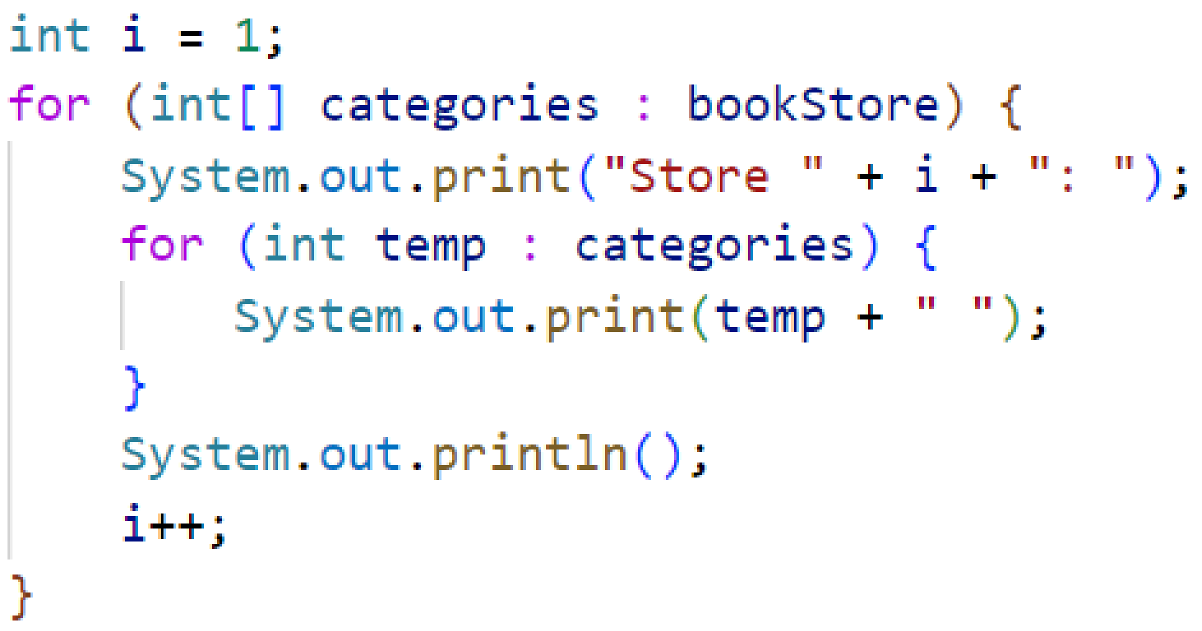
```
for (int i = 0; i < bookStore.length; i++) {
    System.out.print("Store " + (i + 1) + ": ");
    for (int j = 0; j < bookStore[0].length; j++) {
        System.out.print(bookStore[i][j] + " ");
    }
    System.out.println();
}
```



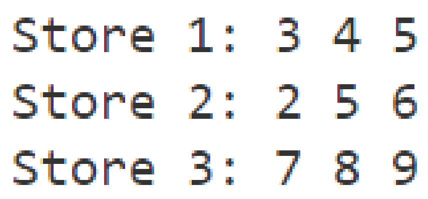
```
Store 1: 3 4 5
Store 2: 2 5 6
Store 3: 7 8 9
```

Nested Loops in Arrays

Nested foreach loop to display array values



```java
int i = 1;
for (int[] categories : bookStore) {
    System.out.print("Store " + i + ": ");
    for (int temp : categories) {
        System.out.print(temp + " ");
    }
    System.out.println();
    i++;
}
```

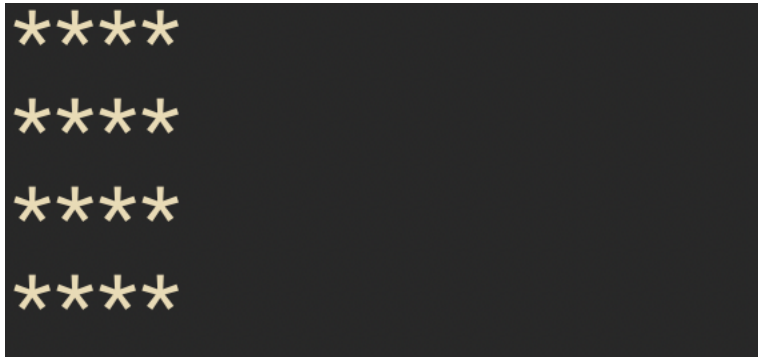


```
Store 1: 3 4 5
Store 2: 2 5 6
Store 3: 7 8 9
```

Case Study

Case Study 1 - Square Star

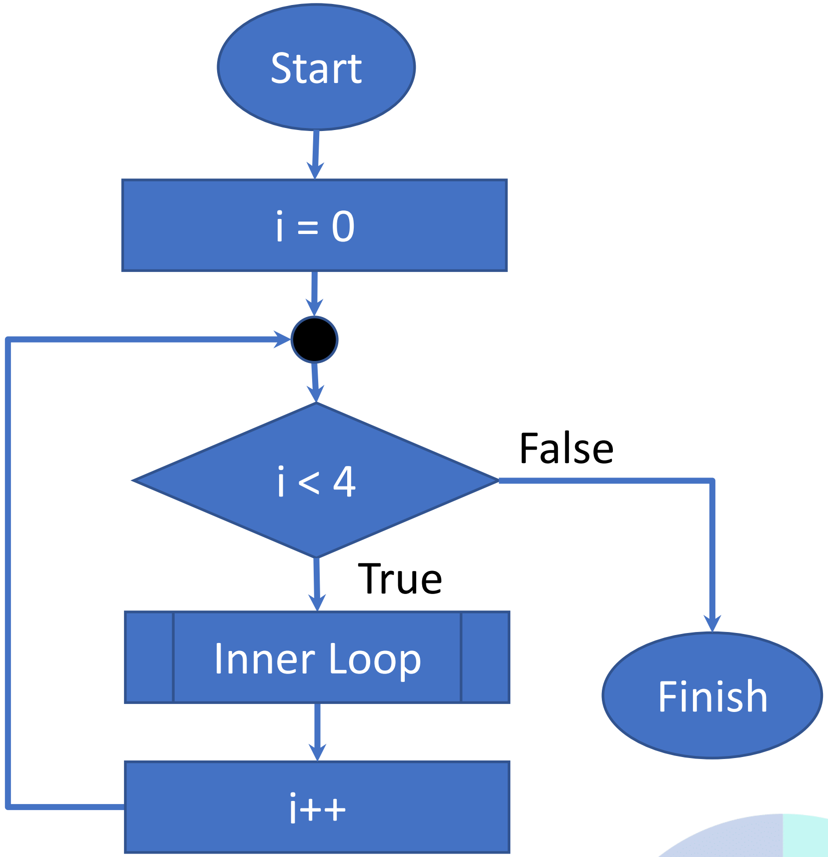
- How to display * as shown in the image using a nested loop?
- By using nested **for**, **while**, and **do-while** loops



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

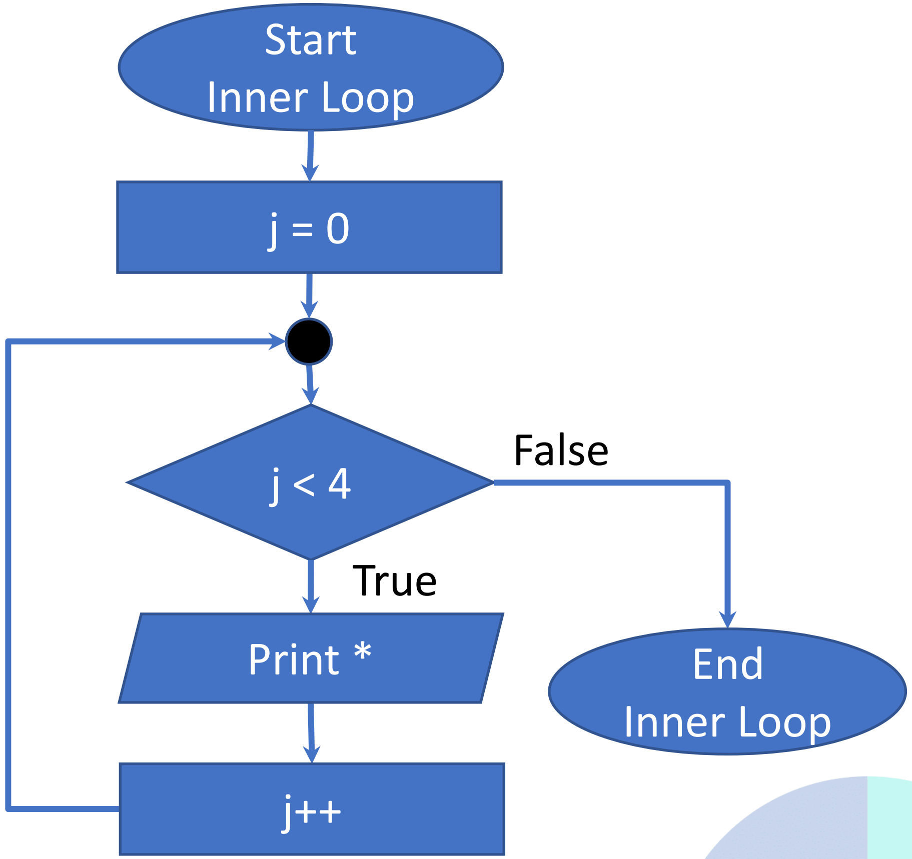
Case Study 1: Answer Logic

- The program to be created must have an **outer loop** and an **inner loop**.
- The **outer loop** is used to count or repeat the number of **rows**, which consists of 4 lines (i = 0; i < 4; i++). Every time the inner loop has finished executing, **a new line will be created**.

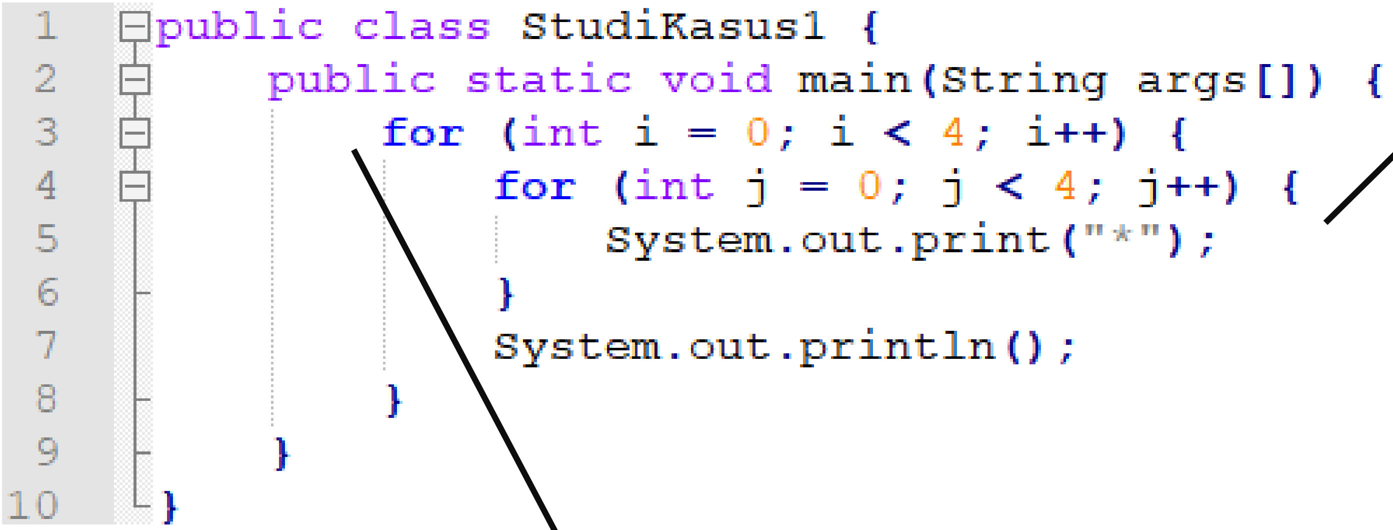


Case Study 1: Answer Logic

- The **inner loop** is used to print the * symbol on the screen, the number of symbols displayed per line will **correspond to the value in variable j**, which consists of 4 symbols * (j = 0; j < 4; j++).



Case Study 1: FOR

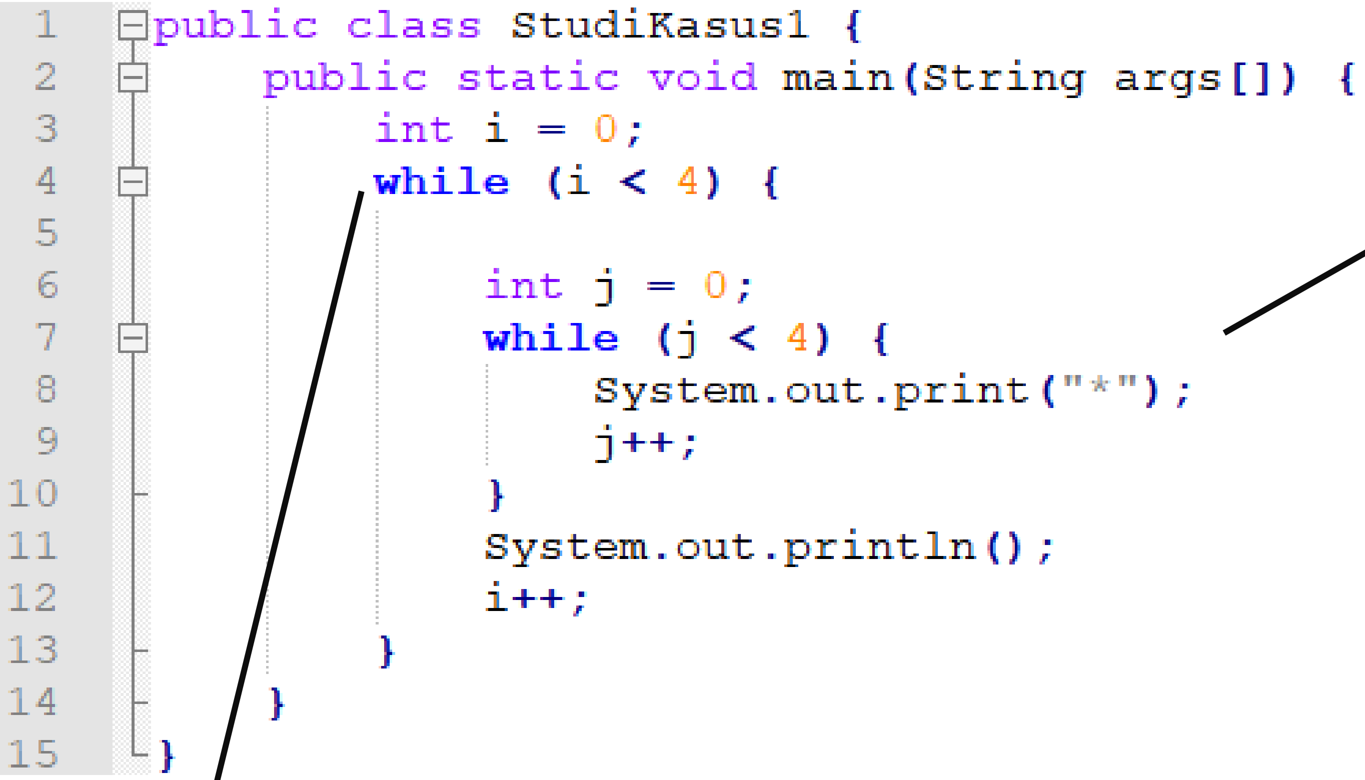


```
public class StudiKasus1 {
    public static void main(String args[]) {
        for (int i = 0; i < 4; i++) {
            for (int j = 0; j < 4; j++) {
                System.out.print("*");
            }
            System.out.println();
        }
    }
}
```

Inner loop produces a row of stars (with 4 stars in each row)

Outer loop repeats the output of the inner loop

Case Study 1: WHILE

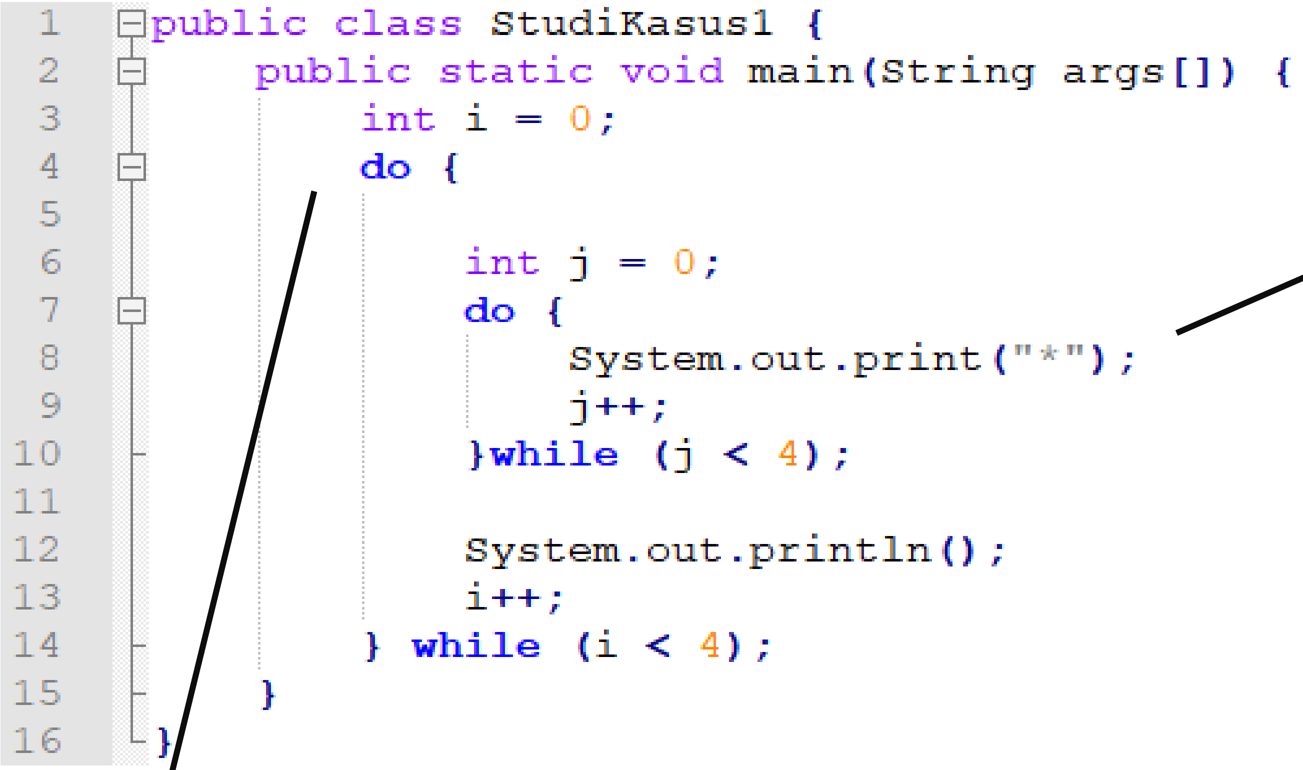


```
public class StudiKasus1 {
    public static void main(String args[]) {
        int i = 0;
        while (i < 4) {

            int j = 0;
            while (j < 4) {
                System.out.print("*");
                j++;
            }
            System.out.println();
            i++;
        }
    }
}
```

Inner loop produces a row of stars (with 4 stars in each row)

Outer loop repeats the output of the inner loop

Case Study 1: DO-WHILE



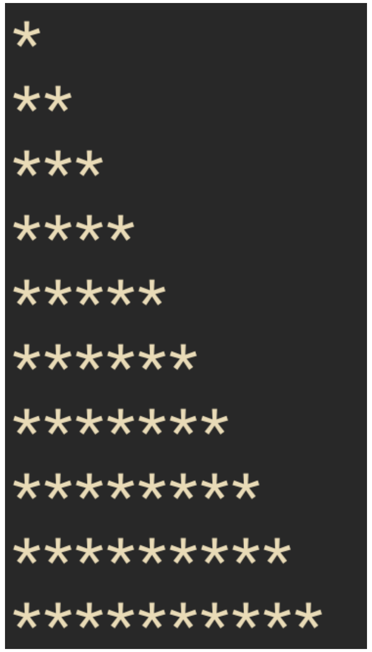
```java
public class StudiKasus1 {
    public static void main(String args[]) {
        int i = 0;
        do {

            int j = 0;
            do {
                System.out.print("*");
                j++;
            }while (j < 4);

            System.out.println();
            i++;
        } while (i < 4);
    }
}
```

Inner loop produces a row of stars (with 4 stars in each row)

Outer loop repeats the output of the inner loop

Case Study 2 - Triangle Star

- How to display * as shown in the image using a nested loop?
- The number of stars in a row depends on the row number (10 lines total)

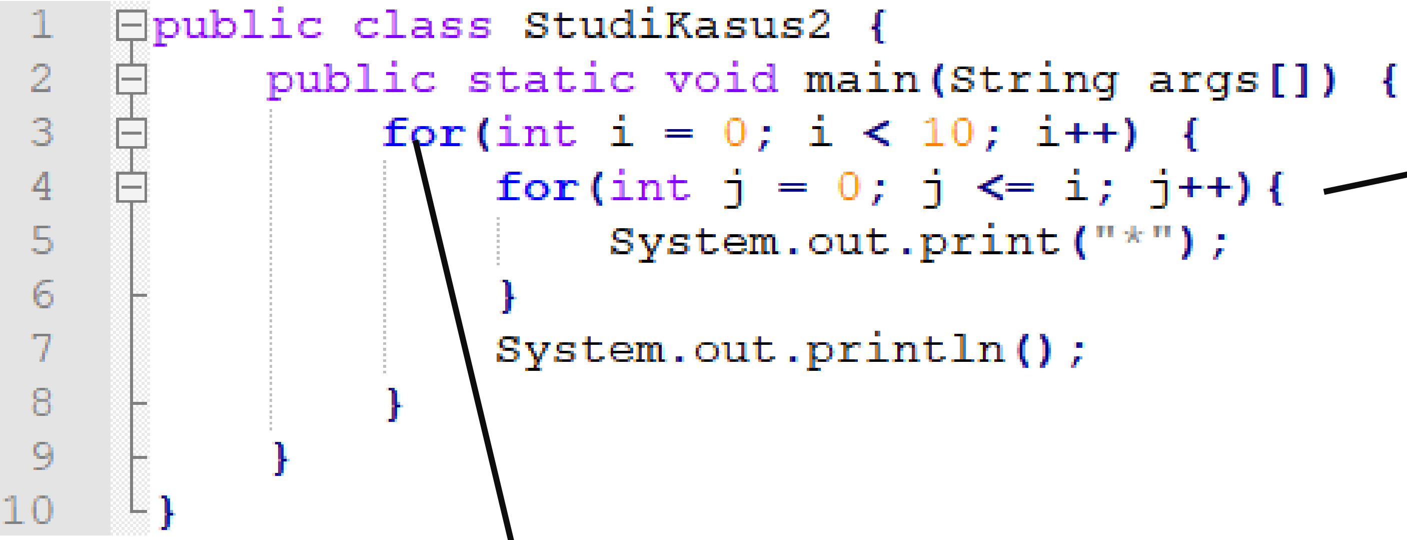


```
*
**
***
****
*****
******
*******
********
*********
**********
```

Case Study 2: Answer Logic

- The program to be created must have an **outer loop** and an **inner loop**.
- The **outer loop** is used to count or repeat the number of **rows**, which consists of 10 lines (i = 0; i < 10; i++). Every time the inner loop has finished executing, **a new line will be created**.
- The **inner loop** is used to print the * symbol on the screen, the number of symbols displayed per line will **correspond (equal to) to the row number in i** (j = 0; j <= i; j++).

Case Study 2: FOR

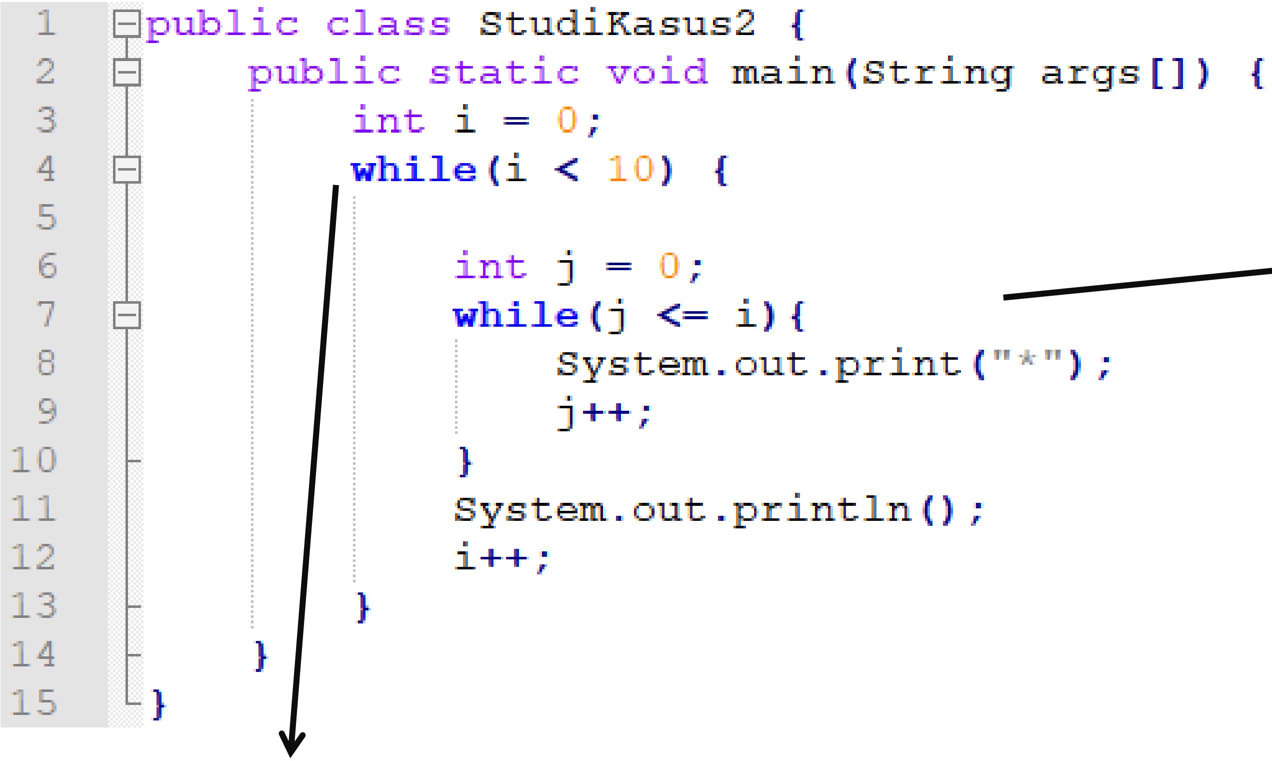


```
public class StudiKasus2 {
    public static void main(String args[]) {
        for(int i = 0; i < 10; i++) {
            for(int j = 0; j <= i; j++){
                System.out.print("*");
            }
            System.out.println();
        }
    }
}
```

The inner loop generates a row of stars with the number corresponding to the row number in i

Outer loop repeats the output of the inner loop

Case Study 2: WHILE

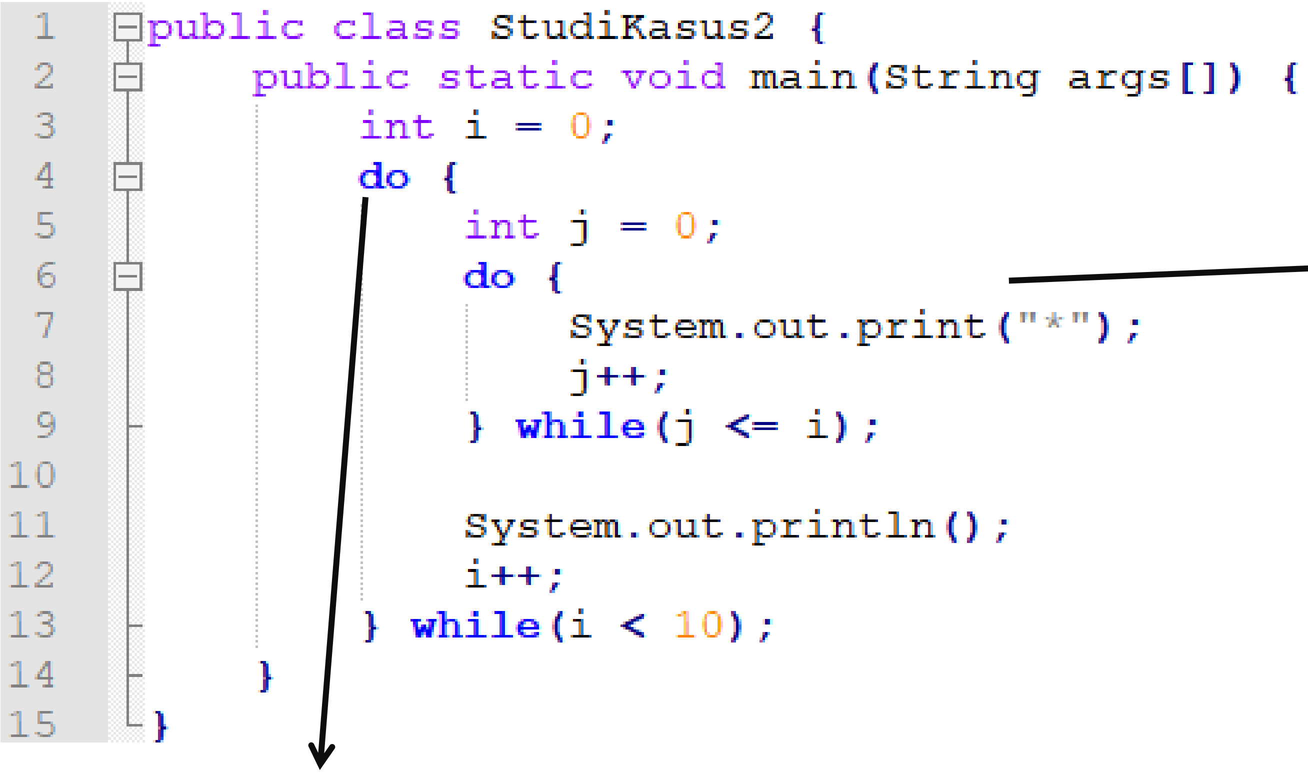


```java
public class StudiKasus2 {
    public static void main(String args[]) {
        int i = 0;
        while(i < 10) {

            int j = 0;
            while(j <= i){
                System.out.print("*");
                j++;
            }
            System.out.println();
            i++;
        }
    }
}
```

The inner loop generates a row of stars with the number corresponding to the row number in i

Outer loop repeats the output of the inner loop

Case Study 2: DO-WHILE



```java
public class StudiKasus2 {
    public static void main(String args[]) {
        int i = 0;
        do {
            int j = 0;
            do {
                System.out.print("*");
                j++;
            } while(j <= i);

            System.out.println();
            i++;
        } while(i < 10);
    }
}
```

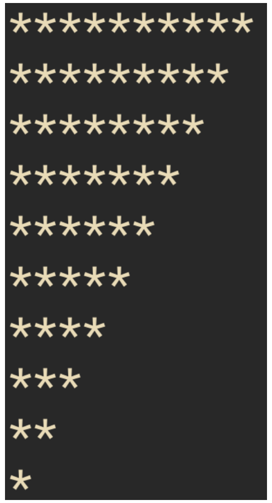
The inner loop generates a row of stars with the number corresponding to the row number in i

Outer loop repeats the output of the inner loop

Group Assignments

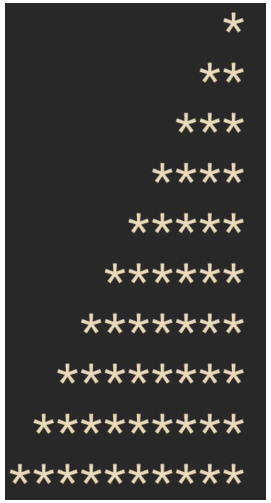
1. Identify in the form of a flowchart according to each group's project, features that require the use nested loops.
2. Create a flowchart to produce a solution like the following image (you can choose one of the nested loop types: for/while/do-while)

a



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

b



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