

Loops



Loops

- In general, statements are executed sequentially: The first statement in a function is executed first, followed by the second, and so on.
- There may be a situation when you need to execute a block of code several number of times.
- Programming languages provide various control structures that allow for more complicated execution paths.
- A loop statement allows us to execute a statement or group of statements multiple times

Loops

```
graph TD; A[Loops] --> B[For Loop]; A --> C[While Loop]; A --> D[Do-While Loop];
```

The diagram is a hierarchical tree structure. At the top is a rounded rectangular box labeled 'Loops'. Three arrows point downwards from this box to three separate oval boxes below it. The left oval is labeled 'For Loop', the middle oval is labeled 'While Loop', and the right oval is labeled 'Do-While Loop'. The background features stylized blue and white waves at the top and green hills at the bottom.

For Loop

While Loop

Do-While
Loop

A stylized landscape illustration. In the foreground, there are rolling green hills. On the left, a tree with a brown trunk and a large, rounded canopy of purple and pink leaves stands on a small patch of orange ground. The background features more rolling hills in shades of blue and white, suggesting a distant or misty horizon. The text 'For Loop' is written in a brown, cursive font in the center-right of the image.

For Loop

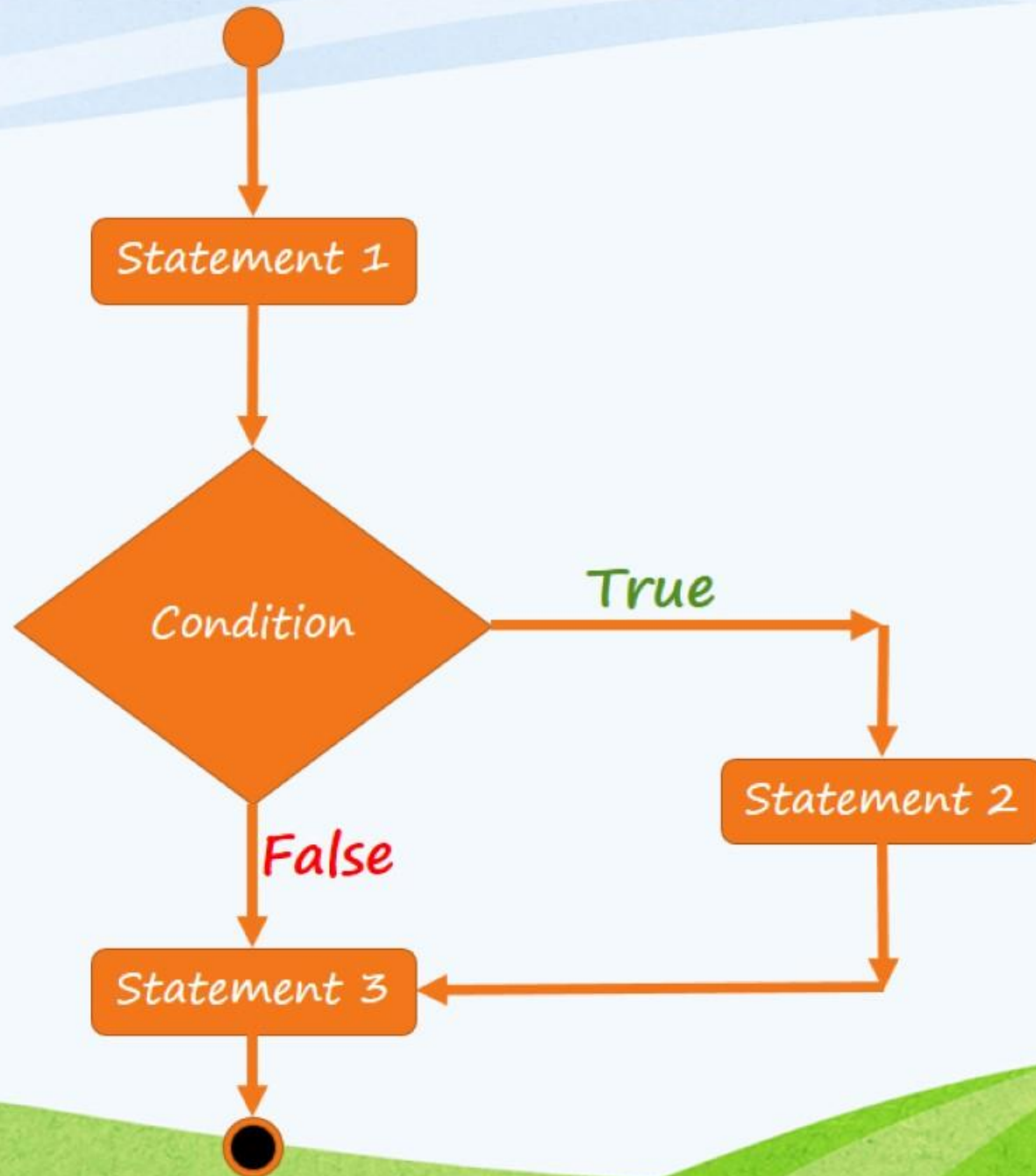
For Loop Syntax

```
for ( initialization; condition; increment/decrement )  
{  
    //statement  
}
```


How For Loop Works

- The initialization step is executed first, and only once. This step allows you to declare and initialize any loop control variables and this step ends with a semi colon (;).
- Next, the condition is executed. If it is true, the body of the loop is executed. If it is false, the body of the loop will not be executed and control jumps to the next statement.
- After the body of the for loop gets executed, the control jumps back up to the increment/decrement statement. This statement allows you to update any loop control variables.

Flow Chart



```
for(int i=0; i<5; i++)  
    {  
        System.out.println(i);  
    }
```

Output: 0

1

2

3

4


```
for(int i=0; i<5;)
{
    System.out.println(i);
    i++;
}
```

Output: 0
1
2
3
4

A stylized landscape illustration featuring rolling green hills in the foreground, a small tree with a brown trunk and purple and pink foliage on the left, and a background of light blue and white wavy bands representing a sky or distant hills. The text 'While Loop' is centered in the white area.

While Loop

While Loop Syntax

//initialization

while (condition)

{

//statement

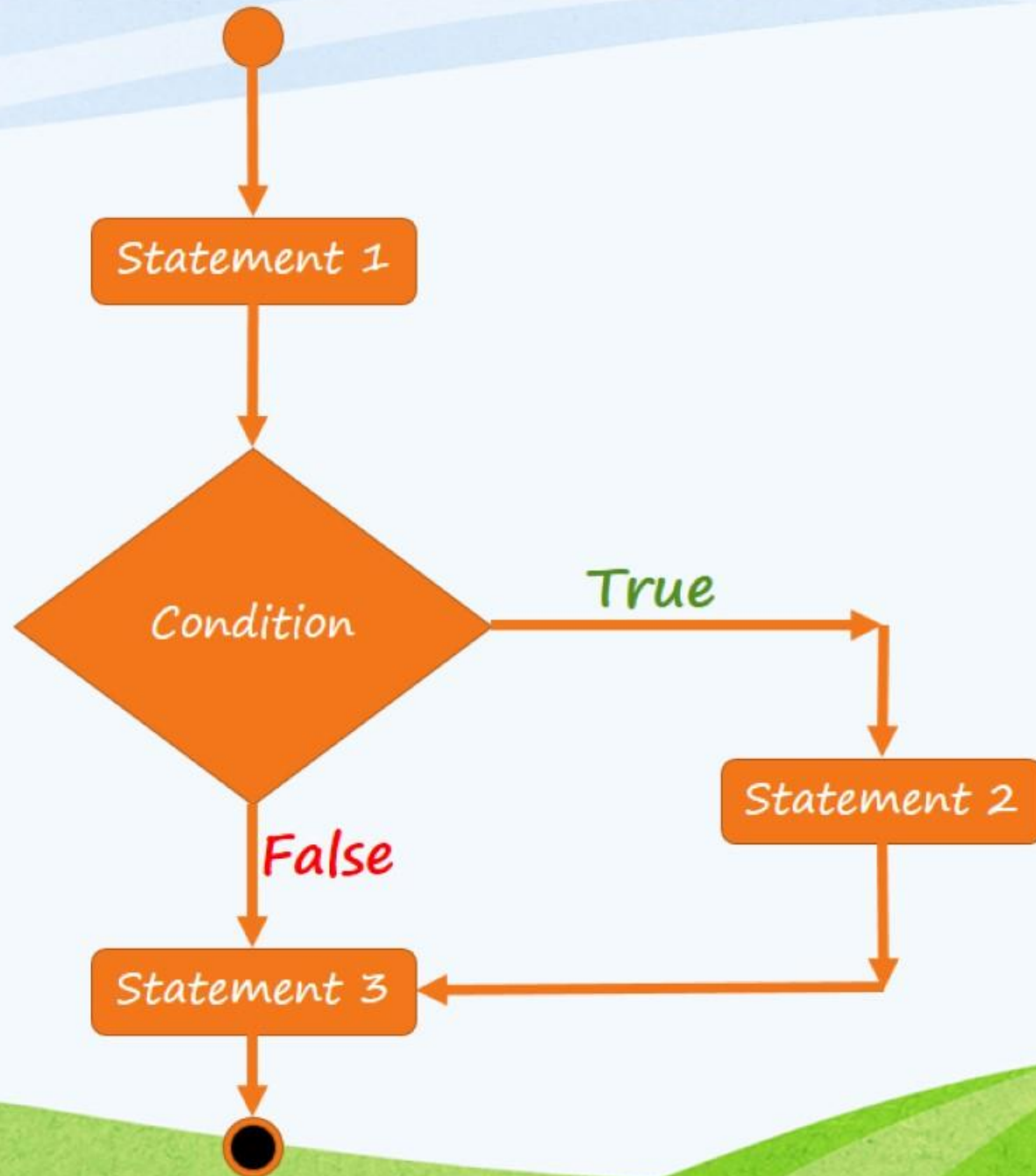
//increment/decrement

}

How While Loop Works

- When executing, if the condition result is true, then the actions inside the loop will be executed. This will continue as long as the expression result is true.
- When the condition becomes false, program control passes to the line immediately following the loop.

Flow Chart



```
int i = 0;
while(i < 5)
{
    System.out.println(i);
    i++;
}
```

Output:

0

1

2

3

4



Do While Loop

Do While Syntax

//initialization

do{

 //statement

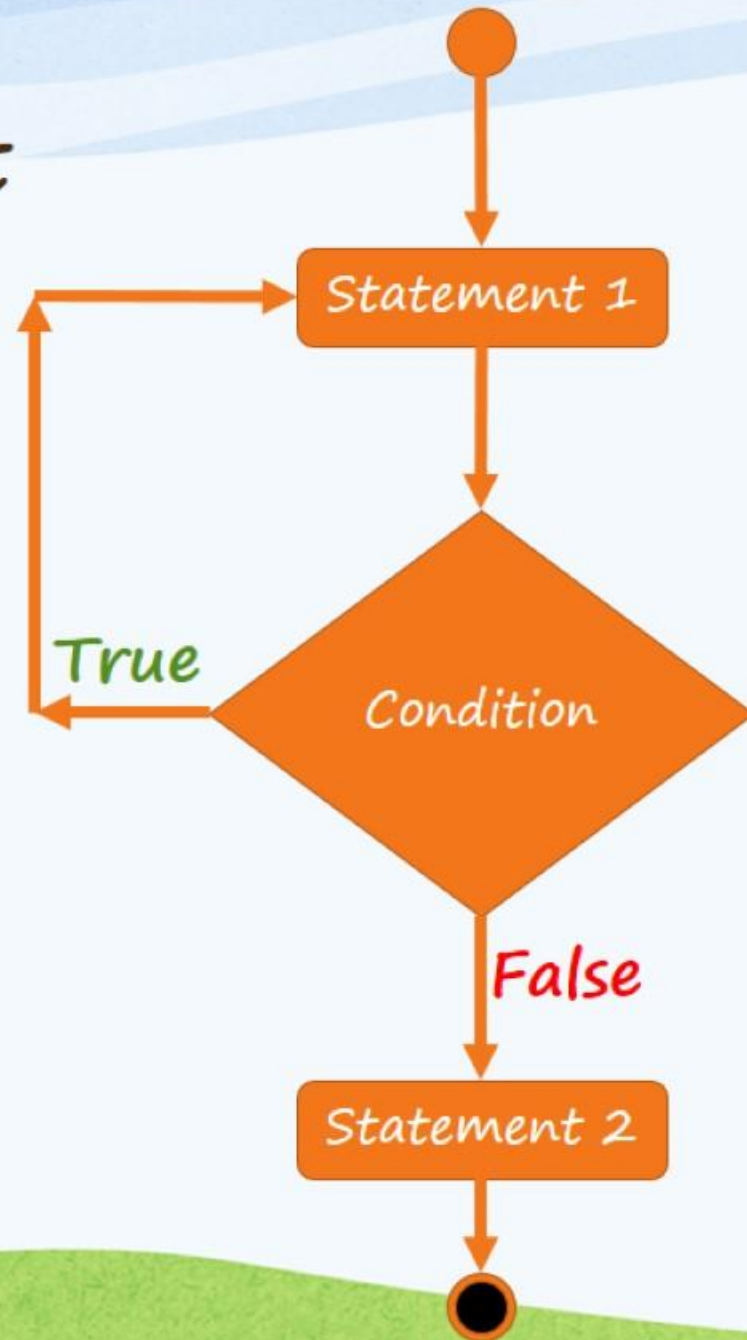
 //increment/decrement

} while (condition);

How Do While Loop Works

- The statements in the loop at least execute once before the Condition is tested.
- If the Condition is true, the control jumps back up to do statement, and the statements in the loop execute again. This process repeats until the Condition is false.

Flow Chart



```
int i = 0;  
do{  
    System.out.println(i);  
    i++;  
}while(i<5);
```

Output:

0

1

2

3

4

```
int i = 0;
```

```
do{
```

```
    System.out.println(i);
```

```
    i++;
```

```
}while(i>5);
```

Output: 0