SYNTAX TECHNOLOGIES

JAVA

Class 7

Agenda

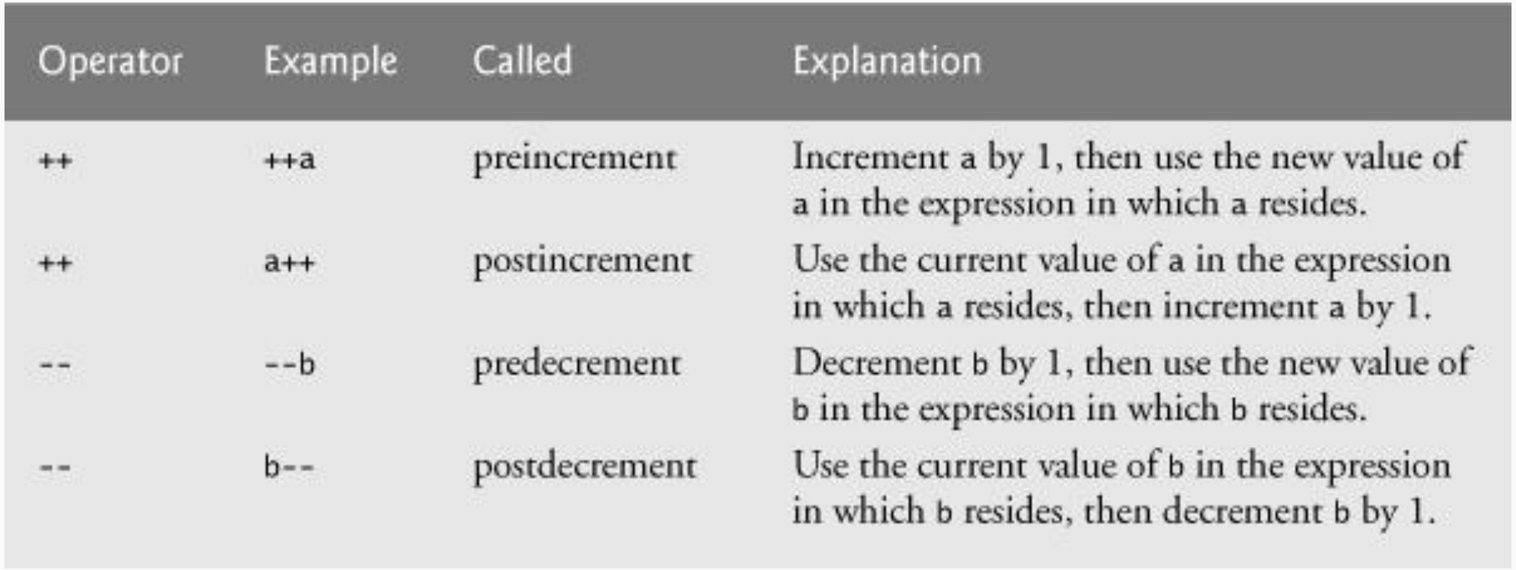
Increment & Decrement Operators

Loops in JAVA:

- while loop
- do while loop
- for loop

Increment and Decrement Shorthand Operators

- Increment (++) and decrement (--) operators in Java programming let you easily add 1 to, or subtract 1 from a variable.
- Increment and decrement operators are widely used in controlling loops.
- Increment and Decrement Operators can not be applied to boolean : We can apply ++ and -- operators for all primitive data types except Boolean.



Operator	Example	Called	Explanation
++	++a	preincrement	Increment a by 1, then use the new value of a in the expression in which a resides.
++	a++	postincrement	Use the current value of a in the expression in which a resides, then increment a by 1.
--	--b	predecrement	Decrement b by 1, then use the new value of b in the expression in which b resides.
--	b--	postdecrement	Use the current value of b in the expression in which b resides, then decrement b by 1.

Why use loop ?

Scenario

you have a script written , if you run you program without the loop it will only run once. What if you want to run that script 20 times? Would you like to wait for the program to finish so you can click the run button again **IMAGINE** you have to repeat this step for 20 times. **IT WILL BE A WASTE OF TIME**

What is loop? Importance of loop

- Loop is a general concept in any programming language which allow to run certain piece/block of code multiple times
- We use loops for repetitive execution
- We use loops to avoid code redundancy
- Time consuming process to execute the program is reduced.

Type of loops

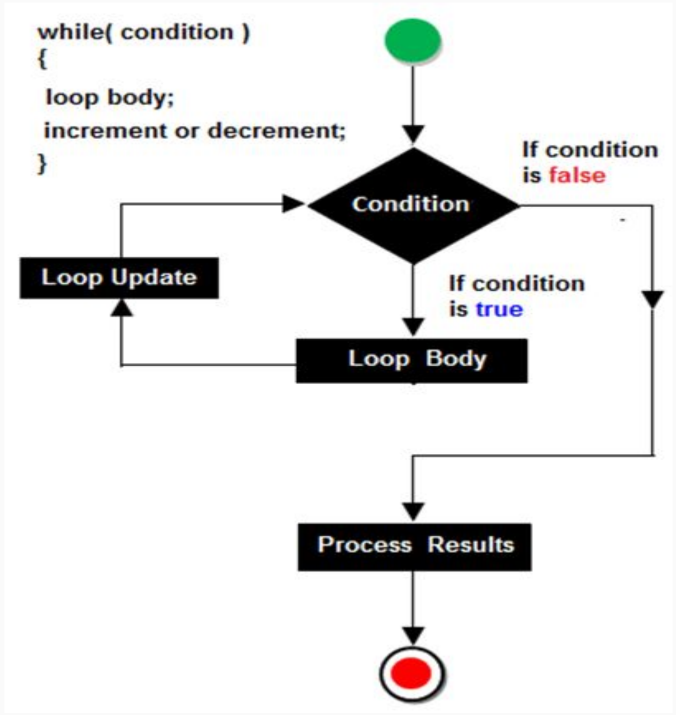
A loop statement allows us to execute a statement or group of statements multiple times

Types of loops:

1. **While loop**
2. **Do while loop**
3. **For loop**
4. **Enhanced loop**

While loop

while loop first checks the condition if condition is true then control goes inside the loop body otherwise goes outside of the body.


```
while( condition )
{
 loop body;
 increment or decrement;
}
```

While loop

```
while (condition) {
     statement(s);
 increment/decrement
}
```

The while statement evaluates expression, which must return a boolean value. If the expression evaluates to true, the while statement executes the statement(s) in the while block. The while statement continues testing the expression and executing its block until the expression evaluates to false.

Example:

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

While loop

Note: The important point to note when using while loop is that we need to use increment or decrement statement inside while loop so that the loop variable gets changed on each iteration, and at some point condition returns false.

This way we can end the execution of while loop otherwise the loop would execute indefinitely.

do while loop

```
do {
    statement(s);
}while (condition);
```

do while loop is similar to while loop with only difference that it checks for condition after executing the statements

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

do while loop

A do while loop is a control flow statement that executes a block of **code at least once,** and then repeatedly executes the block, or not, depending on a given condition at the end of the block (in while)

When use do.. while loop

- When we need to repeat the statement block **at least one time** then use do-while loop.
- In do-while loop post-checking process will occur, that is after execution of the statement block, the condition part will be executed.

for loop

The Java for loop is used to iterate a part of the program several times. If the number of iteration is **fixed**, it is recommended to use for loop.

Syntax:

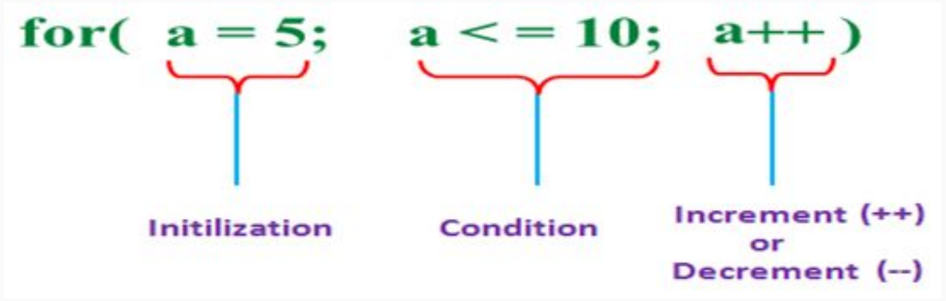
```
for (initialization;condition;increment/decrement) {
      block of code
  }
                        (or)

for (start Value;endValue;increment/decrement) {
      block of code
  }
```

for(initialization;condition;increment/decrement{ block of code}

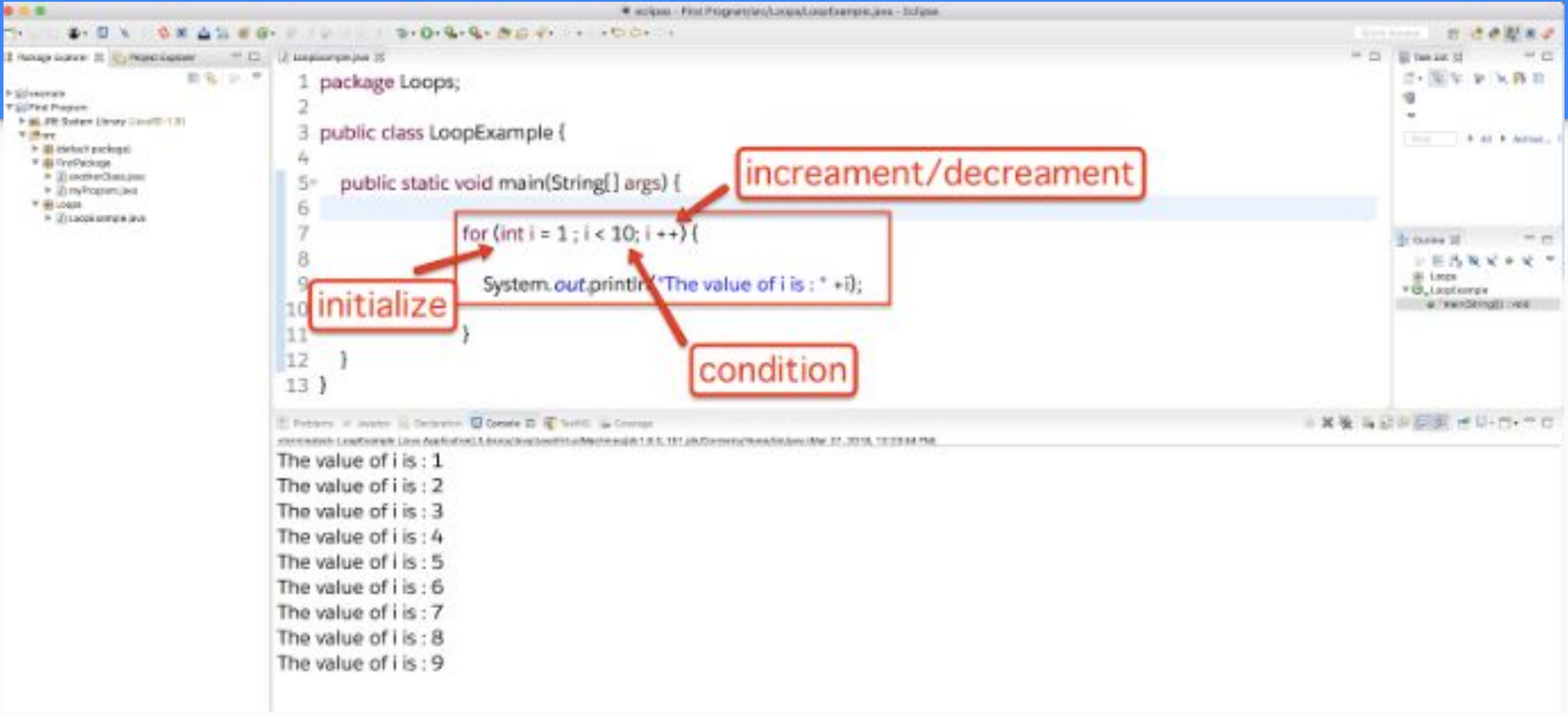
Initialization: This step executes first and is executed only once when we are entering into the loop for the first time. This step is used to declare and initialize any loop control variables.

Condition: This is the next step after initialization, if it is true, the body of the loop is executed, if it is false the body of the loop does not execute and flow of control goes outside of the for loop.

Increment or Decrements: After completion of Initialization and Condition steps loop body code is executed and then Increment or Decrements steps is executed. This statement allows to update any loop control variables.



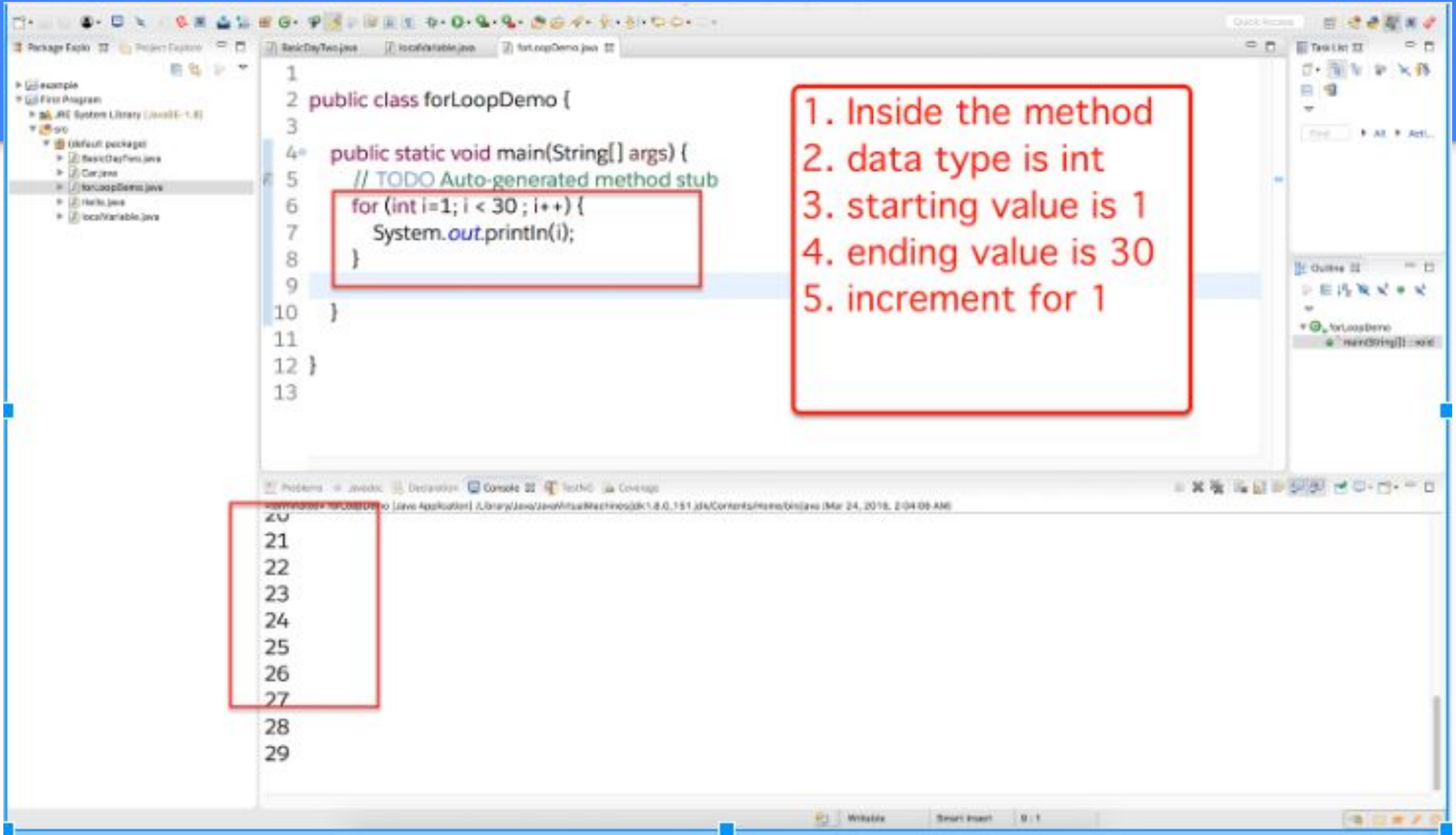
for(initialization;condition;increment/decrement
{ block of code }



```java
package Loops;

public class LoopExample {

    public static void main(String[] args) {

            for (int i = 1 ; i < 10; i ++) {

                System.out.println("The value of i is : " +i);

            }
    }
}
```

```
The value of i is : 1
The value of i is : 2
The value of i is : 3
The value of i is : 4
The value of i is : 5
The value of i is : 6
The value of i is : 7
The value of i is : 8
The value of i is : 9
```

for(start value;end value;increment/decrement)
{block of code }



```java
public class forLoopDemo {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        for (int i=1; i < 30 ; i++) {
            System.out.println(i);
        }

    }

}
```

1. Inside the method
2. data type is int
3. starting value is 1
4. ending value is 30
5. increment for 1

```
20
21
22
23
24
25
26
27
28
29
```