Module 1: Introduction to Java Fundamentals

Premanand S

Assistant Professor
School of Electronics Engineering
Vellore Institute of Technology
Chennai Campus

premanand.s@vit.ac.in

January 5, 2026

Module 1: Introduction to Java Fundamentals

- OOP Paradigm and Features of Java
- JVM, Bytecode, Java Program Structure
- Data Types, Variables, Naming Conventions
- Operators, Control and Looping Constructs
- One- and Multi-dimensional Arrays
- Enhanced for-loop
- Strings, StringBuffer, StringBuilder, Math Class
- Wrapper Classes

Control Constructs in Java

Control constructs decide **which path** a program should take based on conditions.

Control Constructs in Java

Control constructs decide **which path** a program should take based on conditions.

Decision Making Statements:

- `if`
- `if--else`
- `else--if` ladder
- `switch`

Control Constructs in Java

Control constructs decide **which path** a program should take based on conditions.

Decision Making Statements:

- `if`
- `if--else`
- `else--if` ladder
- `switch`

Purpose:

- Make programs intelligent
- Allow multiple execution paths
- Handle real-world conditions

Looping Constructs in Java

Looping constructs allow a block of code to **repeat** until a condition is satisfied.

Looping Constructs in Java

Looping constructs allow a block of code to **repeat** until a condition is satisfied.

Looping Statements:

- `for` loop
- `while` loop
- `do--while` loop
- Enhanced `for` loop (for–each)

Looping Constructs in Java

Looping constructs allow a block of code to **repeat** until a condition is satisfied.

Looping Statements:

- `for` loop
- `while` loop
- `do--while` loop
- Enhanced `for` loop (for–each)

Purpose:

- Avoid repetitive code
- Improve efficiency
- Handle large data and iterations

`if` Statement — Explanation

The `if` **statement** allows a program to **execute a block of code only when a condition is true**.

`if` Statement — Explanation

The `if` **statement** allows a program to **execute a block of code only when a condition is true**.
In simple words:

- A condition is checked
- If the condition is **true**, code is executed
- If the condition is **false**, code is skipped

`if` Statement — Explanation

The `if` **statement** allows a program to **execute a block of code only when a condition is true**.
In simple words:

- A condition is checked
- If the condition is **true**, code is executed
- If the condition is **false**, code is skipped

Why do we need `if`?

- To make decisions
- To control program flow
- To handle real-world logic

if Statement — Syntax (Java)

Syntax:

```
if (condition) {
 // statements to execute
}
```

`if` Statement — Syntax (Java)

Syntax:

```
if (condition) {
 // statements to execute
}
```

Important Rules:

- Condition must be a **boolean expression**
- Curly braces {} define the block
- Parentheses () are mandatory for condition

if Statement — Java vs Python

Python

```
marks = 75

if marks >= 50:
print("PASS")
```

- Indentation-based
- No braces
- Condition without ()

Java

```
int marks = 75;

if (marks >= 50) {
 System.out.println("PASS");
}
```

- Uses braces {}
- Condition inside ()
- Statements end with ;

if Statement

```
public class IfExample {
 public static void main(String[] args) {

  int marks = 65;

  if (marks >= 50) {
   System.out.println("Result: PASS");
  }

  System.out.println("End of program.");
 }
}
```

Discussion 1

Code:

```
int x = 5;

if (x > 10) {
 System.out.println("Hello");
}
```

Discussion 1

Code:

```
int x = 5;

if (x > 10) {
 System.out.println("Hello");
}
```

Answer:

- Program runs successfully
- Condition is **false**
- **No output is printed**

Discussion 1

Code:

```
int x = 5;

if (x > 10) {
 System.out.println("Hello");
}
```

Answer:

- Program runs successfully
- Condition is **false**
- **No output is printed**

Key Insight: An `if` block executes only when the condition is true.

Discussion 2

Code:

```
int x = 10;

if (x = 5) {
 System.out.println("Inside if");
}
```

Discussion 2

Code:

```
int x = 10;

if (x = 5) {
 System.out.println("Inside if");
}
```

Answer:

- compile-time error
- `x = 5` is an assignment, not a condition

Discussion 2

Code:

```
int x = 10;

if (x = 5) {
 System.out.println("Inside if");
}
```

Answer:

- compile-time error
- `x = 5` is an assignment, not a condition

Key Insight: Java requires the `if` condition to be **boolean**.

Discussion 3

Code:

```
int x = 4;

if (++x > 4) {
 System.out.println(x);
}
```

Discussion 3

Code:

```
int x = 4;

if (++x > 4) {
 System.out.println(x);
}
```

Answer:

- `x` becomes 5 before comparison
- Condition becomes `5 > 4` $\rightarrow$ true
- Output: **5**

Discussion 3

Code:

```
int x = 4;

if (++x > 4) {
 System.out.println(x);
}
```

Answer:

- `x` becomes 5 before comparison
- Condition becomes `5 > 4` $\rightarrow$ true
- Output: **5**

Key Insight: Pre-increment happens **before** condition check.

Discussion 4

Code:

```
int x = 15;

if (x > 10) {
 System.out.println("A");
}

if (x > 5) {
 System.out.println("B");
}
```

Discussion 4

Code:

```
int x = 15;

if (x > 10) {
 System.out.println("A");
}

if (x > 5) {
 System.out.println("B");
}
```

Answer:

- Both conditions are true
- Output: A and B both get printed

Discussion 4

Code:

```
int x = 15;

if (x > 10) {
 System.out.println("A");
}

if (x > 5) {
 System.out.println("B");
}
```

Answer:

- Both conditions are true
- Output: A and B both get printed

Key Insight: Each `if` is independent.

Discussion 5

Code:

```
int marks = 50;

if (marks > 50) {
 System.out.println("Pass");
}
```

Discussion 5

Code:

```
int marks = 50;

if (marks > 50) {
 System.out.println("Pass");
}
```

Answer:

- Condition is false
- No output is printed

Discussion 5

Code:

```
int marks = 50;

if (marks > 50) {
 System.out.println("Pass");
}
```

Answer:

- Condition is false
- No output is printed

Key Insight: > and >= make a big difference.

Discussion 6

Code:

```
boolean flag = true;

if (flag) {
 System.out.println("Condition satisfied");
}
```

Discussion 6

Code:

```
boolean flag = true;

if (flag) {
 System.out.println("Condition satisfied");
}
```

Answer:

- Valid Java code
- Output: **Condition satisfied**

Discussion 6

Code:

```
boolean flag = true;

if (flag) {
 System.out.println("Condition satisfied");
}
```

Answer:

- Valid Java code
- Output: **Condition satisfied**

Key Insight: A boolean variable itself can be a condition.

Discussion 7

Code:

```
int x = 10;

if (x > 5) {
}

System.out.println("Done");
```

Discussion 7

Code:

```
int x = 10;

if (x > 5) {
}

System.out.println("Done");
```

Answer:

- Empty `if` block is valid
- Output: **Done**

Discussion 7

Code:

```
int x = 10;

if (x > 5) {
}

System.out.println("Done");
```

Answer:

- Empty `if` block is valid
- Output: **Done**

Key Insight: Braces define a block, even if it is empty.

Discussion 8

Code:

```
int x = 10;

if (x > 5);
{
 System.out.println("Hello");
}
```

Discussion 8

Code:

```
int x = 10;

if (x > 5);
{
 System.out.println("Hello");
}
```

Answer:

- Semicolon ends the `if` statement
- Block executes unconditionally
- Output: **Hello**

Discussion 8

Code:

```
int x = 10;

if (x > 5);
{
 System.out.println("Hello");
}
```

Answer:

- Semicolon ends the `if` statement
- Block executes unconditionally
- Output: **Hello**

Key Insight: A misplaced semicolon breaks logic silently.

Discussion 9

Code:

```
if (true) {
 System.out.println("Always executes");
}
```

Discussion 9

Code:

```
if (true) {
 System.out.println("Always executes");
}
```

Answer:

- Condition is always true
- Output: **Always executes**

Discussion 9

Code:

```
if (true) {
 System.out.println("Always executes");
}
```

Answer:

- Condition is always true
- Output: **Always executes**

Key Insight: Conditions need not depend on variables.

Discussion 10

Code:

```
int x = 8;

if (x > 5) {
 if (x > 10) {
  System.out.println("Greater than 10");
 }
}
```

Discussion 10

Code:

```
int x = 8;

if (x > 5) {
 if (x > 10) {
  System.out.println("Greater than 10");
 }
}
```

Answer:

- Outer condition is true
- Inner condition is false
- No output is printed

Discussion 10

Code:

```
int x = 8;

if (x > 5) {
 if (x > 10) {
  System.out.println("Greater than 10");
 }
}
```

Answer:

- Outer condition is true
- Inner condition is false
- No output is printed

Key Insight: Inner `if` depends on outer `if`.

IF Questions

1. Write a Java program to check whether a number is positive. If positive, print `"Positive Number"`.

IF Questions

1. Write a Java program to check whether a number is positive. If positive, print `"Positive Number"`.
2. Given `marks = 72`, print `"Passed"` if marks ≥ 50.

IF Questions

1. Write a Java program to check whether a number is positive. If positive, print `"Positive Number"`.
2. Given `marks = 72`, print `"Passed"` if marks ≥ 50.
3. Check whether a given number is even. If yes, print `"Even Number"`.

IF Questions

1. Write a Java program to check whether a number is positive. If positive, print `"Positive Number"`.
2. Given `marks = 72`, print `"Passed"` if marks ≥ 50.
3. Check whether a given number is even. If yes, print `"Even Number"`.
4. Check whether a person is eligible to vote. Condition: Age ≥ 18. Print `"Eligible to Vote"`.

IF Questions

1. Given a number: Print `"Divisible by 3"` if divisible by 3. Print `"Divisible by 5"` if divisible by 5. (Both messages can appear.)

IF Questions

1. Given a number: Print `"Divisible by 3"` if divisible by 3. Print `"Divisible by 5"` if divisible by 5. (Both messages can appear.)
2. Given a character, check if it is an uppercase letter (A–Z). If yes, print `"Uppercase Letter"`.

IF Questions

1. Given a number: Print `"Divisible by 3"` if divisible by 3. Print `"Divisible by 5"` if divisible by 5. (Both messages can appear.)
2. Given a character, check if it is an uppercase letter (A–Z). If yes, print `"Uppercase Letter"`.
3. If the total purchase amount is greater than 5000, print `"Discount Applicable"`.

IF Questions

1. Given a number: Print `"Divisible by 3"` if divisible by 3. Print `"Divisible by 5"` if divisible by 5. (Both messages can appear.)
2. Given a character, check if it is an uppercase letter (A–Z). If yes, print `"Uppercase Letter"`.
3. If the total purchase amount is greater than 5000, print `"Discount Applicable"`.
4. If employee experience is greater than 5 years, print `"Bonus Granted"`.

IF Questions

1. Given a number:
   - If number > 0, print `"Positive"`
   - If number < 0, print `"Negative"`
   - If number $== 0$, print `"Zero"`

   (Use only `if`, no `else`)

IF Questions

1. Given a number:
   - If number > 0, print `"Positive"`
   - If number < 0, print `"Negative"`
   - If number $== 0$, print `"Zero"`

   (Use only `if`, no `else`)
2. Given a character: Print `"Digit"` if it is a digit. Print `"Alphabet"` if it is an alphabet.

IF Questions

1. Given a number:
   - If number > 0, print `"Positive"`
   - If number < 0, print `"Negative"`
   - If number $== 0$, print `"Zero"`

   (Use only `if`, no `else`)
2. Given a character: Print `"Digit"` if it is a digit. Print `"Alphabet"` if it is an alphabet.
3. If a number lies between 10 and 50 (inclusive), print `"In Range"`.

IF Questions

1. Given a number:
   - If number > 0, print `"Positive"`
   - If number < 0, print `"Negative"`
   - If number $== 0$, print `"Zero"`

   (Use only `if`, no `else`)
2. Given a character: Print `"Digit"` if it is a digit. Print `"Alphabet"` if it is an alphabet.
3. If a number lies between 10 and 50 (inclusive), print `"In Range"`.
4. If a year is divisible by 4, print `"Leap Year Candidate"`.

IF Questions

1. Write a program using `if` such that:
   - It prints nothing for some inputs
   - Prints something for other inputs

   Explain why.

IF Questions

1. Write a program using `if` such that:
   - It prints nothing for some inputs
   - Prints something for other inputs

   Explain why.
2. Given a number: If number $>$ 0 If number % 2 $==$ 0 Print `"Positive Even"` (No `else`).

IF Questions

1. Write a program using `if` such that:
   - It prints nothing for some inputs
   - Prints something for other inputs

   Explain why.
2. Given a number: If number $>$ 0 If number % 2 $==$ 0 Print `"Positive Even"` (No `else`).
3. Check whether a number is greater than 10. Test with input $=$ 10 and explain the output.

IF Questions

1. Write a program using `if` such that:
   - It prints nothing for some inputs
   - Prints something for other inputs

   Explain why.
2. Given a number: If number $>$ 0 If number % 2 $==$ 0 Print `"Positive Even"` (No `else`).
3. Check whether a number is greater than 10. Test with input $=$ 10 and explain the output.
4. Given:

   ```
   boolean isLoggedIn = false;
   ```

   Print `"Welcome User"` only if user is logged in.

IF Questions

1. Write a program where a condition exists but no output is printed. Explain why.

IF Questions

1. Write a program where a condition exists but no output is printed. Explain why.
2. Check whether the username length is greater than 5. If yes, print `"Valid Username"`.

IF Questions

1. Write a program where a condition exists but no output is printed. Explain why.
2. Check whether the username length is greater than 5. If yes, print `"Valid Username"`.
3. Predict the output without running:

```
int x = 5;
if (++x > 5) {
 System.out.println(x);
}
```

Thank You!

Stay Connected

Premanand S

Email: premanand.s@vit.ac.in
Phone: +91-7358679961

LinkedIn: linkedin.com/in/premsanand
Instagram: instagram.com/premsanand
WhatsApp Channel: anandsDataX

Google Scholar: Google Scholar Profile
GitHub: github.com/anandprems