**Lecture: Java Control Statements | Control Flow in Java**

**Introduction to Control Statements**

* Control statements determine the flow of execution in a program.
* They allow us to:
  + Make decisions (if/else).
  + Loop through repetitive tasks (for, while).
  + Redirect program flow (break, continue).

**Types of Control Statements**

Java control statements are categorized into three main types:

1. **Decision-Making Statements**
   * if, if-else, if-else-if, switch
2. **Looping Statements**
   * for, while, do-while
3. **Jump Statements**
   * break, continue, return

**1. Decision-Making Statements**

**1.1 if Statement**

* Executes a block of code if the condition evaluates to true.

**Syntax**:

java

if (condition) {

// code to be executed if condition is true

}

**Example**:

java

int age = 18;

if (age >= 18) {

System.out.println("You are eligible to vote.");

}

**1.2 if-else Statement**

* Executes one block of code if the condition is **true**, another block if it’s false.

**Syntax**:

java

if (condition) {

// code if condition is true

} else {

// code if condition is false

}

**Example**:

java

int number = 10;

if (number % 2 == 0) {

System.out.println("Even number");

} else {

System.out.println("Odd number");

}

**1.3 if-else-if Ladder**

* Used for multiple conditions.

**Syntax**:

java

if (condition1) {

// code if condition1 is true

} else if (condition2) {

// code if condition2 is true

} else {

// code if none of the above conditions is true

}

**Example**:

java

int marks = 85;

if (marks >= 90) {

System.out.println("Grade: A");

} else if (marks >= 75) {

System.out.println("Grade: B");

} else {

System.out.println("Grade: C");

}

**1.4 switch Statement**

* A cleaner alternative to if-else for multiple choices.

**Syntax**:

java

**switch** (expression) {

case value1:

// code

break;

case value2:

// code

break;

default:

// code

}

**Example**:

java

int day = 3;

switch (day) {

case 1:

System.out.println("Monday");

break;

case 2:

System.out.println("Tuesday");

break;

case 3:

System.out.println("Wednesday");

break;

default:

System.out.println("Invalid day");

}

**2. Looping Statements**

**2.1 for Loop**

* Executes a block of code a specified number of times.

**Syntax**:

java

for (initialization; condition; increment/decrement) {

// code to be executed

}

**Example**:

java

for (int i = 1; i <= 5; i++) {

System.out.println("Iteration: " + i);

}

**2.2 while Loop**

* Executes a block of code as long as the condition is true.

**Syntax**:

java

while (condition) {

// code to be executed

}

**Example**:

java

int i = 1;

while (i <= 5) {

System.out.println("Iteration: " + i);

i++;

}

**2.3 do-while Loop**

* Executes the code at least once, then checks the condition.

**Syntax**:

java

do {

// code to be executed

} while (condition);

**Example**:

java

int i = 1;

do {

System.out.println("Iteration: " + i);

i++;

} while (i <= 5);

**3. Jump Statements**

**3.1 break**

* Exits the loop or switch statement.

**Example**:

java

for (int i = 1; i <= 5; i++) {

if (i == 3) {

break; // exit loop

}

System.out.println("Iteration: " + i);

}

**3.2 continue**

* Skips the current iteration and proceeds to the next.

**Example**:

java

for (int i = 1; i <= 5; i++) {

if (i == 3) {

continue; // skip this iteration

}

System.out.println("Iteration: " + i);

}

**3.3 return**

* Exits the method.

**Example**:

java

public void checkEligibility(int age) {

if (age < 18) {

System.out.println("Not eligible");

return; // exit the method

}

System.out.println("Eligible");

}

**Best Practices**

1. **Avoid infinite loops unless explicitly needed.**
2. Use switch for better readability over multiple if-else statements.
3. Use meaningful variable names in loop conditions and initialization.
4. Avoid hardcoding values in conditions.
5. Be cautious with break and continue, as they can make debugging harder.

**Hands-On Practice**

1. Write a program to calculate the factorial of a number using a for loop.
2. Create a menu-driven program using a switch statement.
3. Write a program to sum all even numbers from 1 to 50 using a while loop.