**Unit-3 Notes**

**Conditional Statements**

Conditional statements in Java allow you to perform different actions based on different conditions.

**1. if Statement**

The if statement executes a block of code if its condition evaluates to true.

**Syntax:**

if (condition) {

// code to be executed if condition is true

}

**Example:**

java

Copy code

int age = 20;

if (age >= 18) {

System.out.println("You are an adult.");

}

**2. if-else Statement**

The if-else statement executes one block of code if its condition is true, and another block if it is false.

**Syntax:**

if (condition) {

// code to be executed if condition is true

} else {

// code to be executed if condition is false

}

**Example:**

int age = 16;

if (age >= 18) {

System.out.println("You are an adult.");

} else {

System.out.println("You are not an adult.");

}

**3. if-else if-else Statement**

The if-else if-else statement allows you to test multiple conditions.

**Syntax:**

if (condition1) {

// code to be executed if condition1 is true

} else if (condition2) {

// code to be executed if condition2 is true

} else {

// code to be executed if all conditions are false

}

**Example:**

int score = 75;

if (score >= 90) {

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

} else if (score >= 80) {

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

} else if (score >= 70) {

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

} else if (score >= 60) {

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

} else {

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

}

**4. switch Statement**

The switch statement allows you to select one of many code blocks to be executed.

**Syntax:**

switch (expression) {

case value1:

// code to be executed if expression equals value1

break;

case value2:

// code to be executed if expression equals value2

break;

// you can have any number of case statements

default:

// code to be executed if expression doesn't match any case

}

**Example:**

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");

}

**Looping Statements**

Looping statements in Java are used to repeatedly execute a block of code.

**1. for Loop**

The for loop is used when you know in advance how many times you want to execute a statement or a block of statements.

**Syntax:**

for (initialization; condition; update) {

// code to be executed

}

**Example:**

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

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

}

**2. while Loop**

The while loop is used when you want to execute a block of code repeatedly as long as a condition is true.

**Syntax:**

while (condition) {

// code to be executed

}

**Example:**

int i = 1;

while (i <= 5) {

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

i++;

}

**3. do-while Loop**

The do-while loop is similar to the while loop, but it executes the block of code at least once before checking the condition.

**Syntax:**

java

Copy code

do {

// code to be executed

} while (condition);

**Example:**

int i = 1;

do {

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

i++;

} while (i <= 5);

**Jump Statements**

Jump statements in Java are used to transfer control to another part of the program.

**1. break Statement**

The break statement is used to terminate the loop or switch statement and transfer control to the statement immediately following the loop or switch.

**Example in a for loop:**

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

if (i == 3) {

break;

}

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

}

// Output: Iteration: 1, Iteration: 2

**Example in a switch statement:**

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");

break;

}

**2. continue Statement**

The continue statement is used to skip the current iteration of the loop and continue with the next iteration.

**Example:**

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

if (i == 3) {

continue;

}

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

}

// Output: Iteration: 1, Iteration: 2, Iteration: 4, Iteration: 5

**3. return Statement**

The return statement is used to exit from a method, with or without a value.

**Example:**

public class Main {

public static void main(String[] args) {

System.out.println(sum(10, 20));

}

public static int sum(int a, int b) {

return a + b;

}

}

// Output: 30