**DAY-02**

**(06/05/2025)**

**Constructor:**

Constructor is used to initialize the object

for every class their will be default constructor whenever we create an object constructor will be called then object will be created.

* Constructor having class name and method properties

public class q2{

public static void main(String[] args){

Demo d1=new Demo();

Demo d2=new Demo();

}

}

class Demo{

static {

System.out.println("Static Block");

}

Demo(){

System.out.println("Construtor");

}

}

Output:

Static Block

Construtoe

Constructor  
  
  
**Static:**

Static block will be created with just static keyword.

It will call Automatically,no need to call neither object or class.

**Static method:**

For static method no need to call with object we can directly call with class name and here static will act as instance as objects

public class Main {

public static void main(String[] args) {

Person obj = new Person();

obj.setAge(24);

obj.setName("Bharath");

obj.setERP(2203031050187L); // long value

int ageResult = obj.getAge();

System.out.println("Age: " + ageResult);

String nameResult = obj.getName();

System.out.println("Name: " + nameResult);

long erpResult = obj.getERP(); // Corrected method name

System.out.println("ERP: " + erpResult);

}

}

class Person {

int age;

String name;

long ERP;

void setAge(int age1) {

this.age = age1;

}

int getAge() {

return age;

}

void setName(String name1) {

this.name = name1;

}

String getName() {

return name;

}

void setERP(long ERP1) { // fixed type to long

this.ERP = ERP1;

}

long getERP() { // fixed return type

return ERP;

}

}

**Output:**

Age: 24

Name: Bharath

ERP: 2203031050187

### ****Fibonacci Series (0 to 5 terms) using**** if-else****:****

public class Fibonacci {

public static void main(String[] args) {

int n = 5; // Total terms

int a = 0, b = 1;

System.out.println("Fibonacci Series from 0 to " + n + ":");

for (int i = 0; i <= n; i++) {

if (i == 0) {

System.out.print(a + " ");

} else if (i == 1) {

System.out.print(b + " ");

} else {

int next = a + b;

System.out.print(next + " ");

a = b;

b = next;

}

}

}

}

**Fibonacci Series from 0 to 5:**

**0 1 1 2 3 5**

**Conditional Statements:**

If

Else

Conditional statements are used to check the condition on to print the relevant block in constant time.

public class IfElseIfExample {

public static void main(String[] args) {

int marks = 85;

if (marks >= 90) {

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

}

else {

System.out.println("Fail");

}

}

}

**Control Statements:**

Entry control loop(example:for, while)

Exit control loop (example:do-while)

### ****Control Statements in Java****

Control statements are used to control the flow of execution in a program. There are two main types of loops based on **entry** and **exit** control:

### ****1. Entry Control Loops:****

An **entry control loop** is a type of loop where the **condition is checked before entering the loop body**. If the condition is **false** initially, the loop will not run even once.

#### ****Examples:**** for ****and**** while ****loops****

#### ****a.**** for ****loop****:

The for loop is commonly used when the number of iterations is known in advance. It consists of three parts:

**Initialization**

**Condition**

**Update/Increment**

##### ****Syntax of a**** for ****loop:****

for (initialization; condition; update) {

// code to be executed

}

##### ****Example: Print numbers from 1 to 5:****

public class ForLoopExample {

public static void main(String[] args) {

// Prints numbers from 1 to 5

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

System.out.println(i);

}

}

}

#### ****b.**** while ****loop****:

The while loop is used when the number of iterations is not known and depends on a condition.

##### ****Syntax of a**** while ****loop:****

while (condition) {

// code to be executed

}

##### ****Example: Print numbers from 1 to 5 using**** while ****loop:****

public class WhileLoopExample {

public static void main(String[] args) {

int i = 1;

while (i <= 5) {

System.out.println(i);

i++; // Increment

}

}

}

### ****2. Exit Control Loops:****

An **exit control loop** is a type of loop where the **condition is checked after the loop body** is executed. This ensures that the loop will always execute **at least once**, regardless of the condition.

#### ****Example:**** do-while ****loop.****

#### ****a.**** do-while ****loop****:

The do-while loop ensures that the loop's body is executed at least once because the condition is checked after the loop body.

##### ****Syntax of a**** do-while ****loop:****

do {

// code to be executed

} while (condition);

##### ****Example: Print numbers from 1 to 5 using**** do-while ****loop:****

public class DoWhileLoopExample {

public static void main(String[] args) {

int i = 1;

do {

System.out.println(i);

i++; // Increment

} while (i <= 5);

}

}

### ****Summary of Differences:****

| **Type of Loop** | **Entry Control** | **Exit Control** |
| --- | --- | --- |
| **Examples** | for, while | do-while |
| **Condition Check** | Before executing the loop body | After executing the loop body |
| **Guarantee of Execution** | May not execute if condition is false initially | Always executes at least once |

**Fibonacci Series using loops:**

public class FibonacciSeries {

public static void main(String[] args) {

int n = 10; // Number of terms in the Fibonacci series

int a = 0, b = 1;

System.out.println("Fibonacci Series up to " + n + " terms:");

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

System.out.print(a + " ");

int next = a + b;

a = b;

b = next;

}

}

}

**Multiplication using loops:**

public class IfElseIfExample {

public static void main(String[] args) {

int[] arr = {1, 2, 3, 4, 5};

System.out.println("\nMultiplication Table of 5:");

int j = 1;

while (j <= 10) {

System.out.println("5 \* " + j + " = " + (5 \* j));

j++;

}

}

}

**Reverse a number:**

public class ReverseNumber {

public static void main(String[] args) {

int num = 1234;

int rev = 0;

while (num > 0) {

int digit = num % 10; // Get last digit

rev = rev \* 10 + digit; // Build reversed number

num = num / 10; // Remove last digit

}

System.out.println("Reversed number: " + rev);

}

}