# Java Lab Programs (30 Programs)

## 1. Hello World Program

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, Java!");
    }
}
```

Output:

```
Hello, Java!
```

## 2. Variables & Data Types

```
public class VariablesDemo {
    public static void main(String[] args) {
        int a = 10;
        double b = 5.5;
        boolean c = true;
        System.out.println("a = " + a);
        System.out.println("b = " + b);
        System.out.println("c = " + c);
    }
}
```

Output:

```
a = 10
b = 5.5
c = true
```

## 3. Scope & Lifetime of Variables

```
public class ScopeDemo {
    public static void main(String[] args) {
        int x = 10;
        if (x > 5) {
            int y = 20;
            System.out.println("x = " + x + ", y = " + y);
        }
        // y is not accessible here
    }
}
```

Output:

```
x = 10, y = 20
```

## 4. If-Else Example

```
public class IfElseDemo {
    public static void main(String[] args) {
        int num = 7;
        if (num % 2 == 0)
            System.out.println(num + " is Even");
        else
            System.out.println(num + " is Odd");
    }
}
```

Output:

```
7 is Odd
```

## 5. Switch Case Example

```
public class SwitchDemo {
    public static void main(String[] args) {
        int day = 2;
        switch(day) {
            case 1: System.out.println("Monday"); break;
            case 2: System.out.println("Tuesday"); break;
            default: System.out.println("Other Day");
        }
    }
}
```

Output:

```
Tuesday
```

## 6. For Loop Example

```
public class ForLoopDemo {
    public static void main(String[] args) {
        for(int i = 1; i <= 5; i++)
            System.out.print(i + " ");
    }
}
```

Output:

```
1 2 3 4 5
```

## 7. While Loop Example

```
public class WhileLoopDemo {
    public static void main(String[] args) {
        int i = 1;
        while(i <= 5) {
            System.out.print(i + " ");
            i++;
        }
    }
}
```

Output:

```
1 2 3 4 5
```

## 8. Do-While Loop Example

```
public class DoWhileDemo {
    public static void main(String[] args) {
        int i = 1;
        do {
            System.out.print(i + " ");
            i++;
        } while(i <= 5);
    }
}
```

Output:

```
1 2 3 4 5
```

## 9. Array Example

```
public class ArrayDemo {
    public static void main(String[] args) {
        int[] arr = {10, 20, 30};
        for(int x : arr)
            System.out.print(x + " ");
    }
}
```

Output:

```
10 20 30
```

## 10. 2D Array Example

```
public class TwoDArrayDemo {
    public static void main(String[] args) {
        int[][] arr = {{1,2},{3,4}};
        for(int i=0; i<arr.length; i++) {
            for(int j=0; j<arr[i].length; j++) {
                System.out.print(arr[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

Output:

```
1 2
3 4
```

## 11. Class and Object

```
class Student {
    String name;
    int age;
}
public class StudentDemo {
    public static void main(String[] args) {
        Student s = new Student();
        s.name = "Gayathri";
        s.age = 21;
        System.out.println(s.name + " " + s.age);
    }
}
```

Output:

```
Gayathri 21
```

## 12. Constructor Example

```
class Car {
    String model;
    Car(String m) { model = m; }
}
public class CarDemo {
    public static void main(String[] args) {
```

```
        Car c = new Car("Tesla");
        System.out.println("Model: " + c.model);
    }
}
```

Output:

```
Model: Tesla
```

## 13. Method Example

```
class MathOp {
    int square(int n) { return n * n; }
}
public class MethodDemo {
    public static void main(String[] args) {
        MathOp m = new MathOp();
        System.out.println("Square of 5 = " + m.square(5));
    }
}
```

Output:

```
Square of 5 = 25
```

## 14. Access Control

```
class Demo {
    private int data = 40;
    public int getData() { return data; }
}
public class AccessDemo {
    public static void main(String[] args) {
        Demo d = new Demo();
        System.out.println(d.getData());
    }
}
```

Output:

```
40
```

## 15. This Keyword

```
class Demo {
    int x;
    Demo(int x) {
        this.x = x;
    }
    void display() {
        System.out.println("x = " + x);
    }
}
public class ThisDemo {
    public static void main(String[] args) {
        Demo d = new Demo(100);
        d.display();
    }
}
```

Output:

```
x = 100
```

## 16. Garbage Collection

```
public class GarbageDemo {
    public void finalize() {
        System.out.println("Object destroyed");
    }
    public static void main(String[] args) {
        GarbageDemo g = new GarbageDemo();
        g = null;
        System.gc();
    }
}
```

Output:

```
Object destroyed
```

## 17. Method Overloading

```
class Calculator {
    int add(int a, int b) { return a+b; }
    double add(double a, double b) { return a+b; }
}
public class OverloadDemo {
    public static void main(String[] args) {
        Calculator c = new Calculator();
        System.out.println(c.add(5,10));
        System.out.println(c.add(2.5,3.5));
    }
}
```

Output:

```
15
6.0
```

## 18. Constructor Overloading

```
class Box {
    int l, b;
    Box() { l=1; b=1; }
    Box(int x, int y) { l=x; b=y; }
    int area() { return l*b; }
}
public class OverloadConstructorDemo {
    public static void main(String[] args) {
        Box b1 = new Box();
        Box b2 = new Box(5,4);
        System.out.println(b1.area());
        System.out.println(b2.area());
    }
}
```

Output:

```
1
20
```

## 19. Method Binding

```
class A {
    void show() { System.out.println("Class A"); }
}
class B extends A {
```

```
    void show() { System.out.println("Class B"); }
}
public class BindingDemo {
    public static void main(String[] args) {
        A obj = new B();
        obj.show();
    }
}
```

Output:

```
Class B
```

## 20. Inheritance

```
class Animal {
    void eat() { System.out.println("Eating..."); }
}
class Dog extends Animal {
    void bark() { System.out.println("Barking..."); }
}
public class InheritanceDemo {
    public static void main(String[] args) {
        Dog d = new Dog();
        d.eat();
        d.bark();
    }
}
```

Output:

```
Eating...
Barking...
```

## 21. Overriding

```
class Parent {
    void show() { System.out.println("Parent"); }
}
class Child extends Parent {
    void show() { System.out.println("Child"); }
}
public class OverrideDemo {
    public static void main(String[] args) {
        Parent p = new Child();
        p.show();
    }
}
```

Output:

```
Child
```

## 22. Exception Handling

```
public class ExceptionDemo {
    public static void main(String[] args) {
        try {
            int a = 5/0;
        } catch(ArithmeticException e) {
            System.out.println("Cannot divide by zero");
        }
    }
}
```

Output:

```
Cannot divide by zero
```

## 23. Parameter Passing

```
class Demo {
    void change(int x) {
        x = x + 10;
    }
}
public class ParamDemo {
    public static void main(String[] args) {
        int a = 5;
        Demo d = new Demo();
        d.change(a);
        System.out.println("a = " + a);
    }
}
```

Output:

```
a = 5
```

## 24. Recursion (Factorial)

```
public class RecursionDemo {
    static int fact(int n) {
        if(n==0) return 1;
        return n*fact(n-1);
    }
    public static void main(String[] args) {
        System.out.println("Factorial of 5 = " + fact(5));
    }
}
```

Output:

```
Factorial of 5 = 120
```

## 25. Nested Class

```
class Outer {
    class Inner {
        void msg() { System.out.println("Hello from Inner"); }
    }
}
public class NestedDemo {
    public static void main(String[] args) {
        Outer.Inner obj = new Outer().new Inner();
        obj.msg();
    }
}
```

Output:

```
Hello from Inner
```

## 26. Inner Class (Anonymous)

```
abstract class Person {
    abstract void eat();
}
```

```
public class AnonymousInner {
    public static void main(String[] args) {
        Person p = new Person() {
            void eat() { System.out.println("Eating anonymously"); }
        };
        p.eat();
    }
}
```

Output:

```
Eating anonymously
```

## 27. String Class Methods

```
public class StringDemo {
    public static void main(String[] args) {
        String s = "Java Programming";
        System.out.println(s.length());
        System.out.println(s.toUpperCase());
        System.out.println(s.substring(0,4));
    }
}
```

Output:

```
16
JAVA PROGRAMMING
Java
```

## 28. Type Casting

```
public class CastingDemo {
    public static void main(String[] args) {
        double d = 10.5;
        int i = (int)d;
        System.out.println("d = " + d);
        System.out.println("i = " + i);
    }
}
```

Output:

```
d = 10.5
i = 10
```

## 29. Operators

```
public class OperatorDemo {
    public static void main(String[] args) {
        int a=10, b=3;
        System.out.println("a+b=" + (a+b));
        System.out.println("a-b=" + (a-b));
        System.out.println("a*b=" + (a*b));
        System.out.println("a/b=" + (a/b));
        System.out.println("a%b=" + (a%b));
    }
}
```

Output:

```
a+b=13
a-b=7
a*b=30
a/b=3
a%b=1
```

## 30. Final Program: Full OOP Concept

```
class Person {
    String name;
    Person(String name) { this.name = name; }
    void display() { System.out.println("Name: " + name); }
}
class Employee extends Person {
    int salary;
    Employee(String n, int s) { super(n); salary = s; }
    void display() {
        super.display();
        System.out.println("Salary: " + salary);
    }
}
public class FullDemo {
    public static void main(String[] args) {
        Employee e = new Employee("Nagasai", 50000);
        e.display();
    }
}
```

Output:

```
Name: Nagasai
Salary: 50000
```