# CORE JAVA

MANUAL V8.3

**MODULE CODE:**

**ANUDIP FOUNDATION**


ICONS AND THEIR MEANING
HINTS:
Get ready for helpful insites on difficult topics and questions.
STUDENTS:
This icon symbolize important instrcutions and guides for the students.
TEACHERS/TRAINERS:
This icon symbolize important instrcutions and guides for the trainers.

## Module 2: Object Oriented Programming and Package

### Chapter 1

| | |
|---|---|
| **Objective:** After completing this lesson you will be able to :<br>* Gain an understanding regarding the advantages of OOPS<br>* Gain an understanding of class and objects in Java | **Materials Required:**<br>1. Computer<br>2. Internet access |
| **Theory Duration:** 80 minutes | **Practical Duration:** 40 minutes |
| **Total Duration:** 120 minutes | |

# Chapter 1

## 1.1 Advantages of OOPS

Java is an object-oriented programming (OOP) language. It is a computer programming process where data structure, data type and operations are defined by programmers. Using an object-based platform like Java is quite advantageous for programmers.

### OOP Advantages

1. **Modular:** An object generates a separate item that is disjointed from other system components.

2. **Customisable:** Performing method and data representation changes are quite easy in an object oriented programming environment. Programmers can make changes within a class without affecting other sections of a program.

3. **Extensible:** Implementing new features and adapting to new operating environments can be accomplished easily through creating new objects or changing current ones.

4. **Simple:** In OOP, programming objects represent real objects. Hence coding becomes very simple and program structure becomes simple.

5. 

6. **Re-usable:** OOP objects can be used across other programs.

7. **Maintenance ease:** Maintenance is easy as problem solving is convenient.

**Differences between pure object oriented programming and semi object oriented programming**

| Pure Object Oriented Programming | Semi Object Oriented Programming |
|---|---|
| A pure object oriented programming language considers all components within a programme as objects | A semi object oriented programming language has objects and primitive types within a programme |
| It does not support any primitive data type | It supports primitive data types such as int, float, long, bool and char |
| Operations are performed only by applying methods to objects | Operations involve objects and primitive data types |
| Example of pure object oriented programming language – SmallTalk | Examples of semi object oriented programming language – C++, Java |

**Why Java is not a pure object oriented programming language –**

* Primitive data types are not objects in Java. They are considered as non-objects.
* Static class members can be accessed in Java without creating objects.

## 1.2 Class and Objects

Classes and objects are two of the most fundamental components of Java, which is an Object-Oriented Programming language.

### i) Class

A Java class is a blueprint or design structure for the creation of objects. It symbolises the set of common methods and properties for similar object types. Classes can be considered as templates for creating objects, and to **define** object methods and data types.

**A typical class declaration in Java consists of –**

1. **Modifiers**: Access modifiers in Java can be of four types i.e private, public, default and protected. If an access modifier is not specified, the default modifier is used.

2. **Class name**: A Java class name is the name assigned to a Java class. It has to begin with a letter (generally a capitalized letter). A class name can be one or two words, but no underscore must be placed between words.

3. **Superclass**: A superclass is the name assigned to the parent of a Java class. It may or may not be used depending on whether it's needed.

4. **Interfaces**: An interface tells the class what action to perform. It is declared using the interface keyword. Multiple interfaces can be implemented by a class.

5. **Body**: The class body is the section of a class containing the definition of methods and variables. It is usually surrounded by braces { }.

Some class types include – Wrapper Class, Mutable Class, Abstract Class, System Class, Network Class, Final Class, Input-Output Class, String Class and Anonymous Class.

**Example of a Java class –**

MyClass.java

**Create a class named 'MyClass' with a variable x:**

```
public class TestClass {
  int x = 6;
}
```

**ii) Object**

An object in object-oriented programming is a representation of real-life objects. Java programs are capable of creating many objects that work by calling methods.

**An object consists of –**

1. **State**: The state of an object is symbolized by its attributes, and they represent the object's properties. Object states are also considered as variables containing data.
2. **Behavior**: Object behavior is represented by its methods. Behavior is also indicative of the response of one object with others.
3. **Identity**: An object's identity is the unique name assigned to it, and enables interaction with other objects.

Objects are related to real-world feature/function elements.

**Relationship between object and class**

A class functions as an outliner to define the common states, properties and behaviors common to multiple objects. An object is a class instance.

**Example of a Java object –**

**Create an object named myObj and print value of x:**

```
public class TestClass {
  int x = 6;
  public static void main(String[] args) {
          MyClass myObj = new MyClass();
           System.out.println(myObj.x);
  }

}
```

**Output:** 6

**Real world examples of classes and objects –**

A real world example can be looked at to understand the difference between classes and objects in an object oriented programming language like Java.

* The term 'vehicle' is a class or category containing all different vehicle types.

* The different types of 'vehicles' such as motorcycle, car, bus and lorry can be considered as subclasses.

* If we consider car as a subclass, all cars such as Tata Nexon, Swift Dzire and Hyundai i20 are objects. These objects belong to the subclass 'car', which itself is a part of the 'vehicle' class.

Class can be described as a collection or group of similar objects. One or more objects (of similar type) can be created within a class. An object contains methods and information that can be manipulated.

* A class is a blueprint containing methods, properties, constructors and variables. The components of a class help programmers interpret what is being represented by the class.

* An object is an instance of a class. An object pointing to a class has to be instantiated as the class cannot be used in a direct manner.

**Practical(40 minutes)**

See the example of a Java object programme below. Write a similar programme to print the value of y = 8. Rewrite again to print the value of z = 7. Show the resulting output of each programme.

```
Public class TestClass {
  int x = 6;
  public static void main(String[] args) {
```

```
        MyClass myObj = new MyClass();
        System.out.println(myObj.x);
  }
}
```

**Instructions: The progress of students will be assessed with the exercises mentioned below.**

**MCQ**

1. OOP stands for _____ Oriented Programming.

a) Obstruction

b) Oscillation

c) Object

d) None of the mentioned

2. Data _____ is defined by programmers in an OOP language like Java.

a) value

b) structure

c) string

d) None of the mentioned

3. OOP is advantageous due to its _____ nature.

a) mobile

b) modular

c) mechanical

d) None of the mentioned

4. Why is the customizability of Java as an OOP language advantageous?

a) class changes do not affect other program sections

b) class changes affect other program sections

c) object changes always affect other programs ections

d) None of the mentioned

5. Programming objects in OOP represent _____ objects.

a) imaginary

b) real

c) superficial

d) None of the mentioned

6. Two of the fundamental components of OOP are classes and _____.

a) files

b) subjects

c) objects

d) None of the mentioned

7. Classes are considered as object creation _____.

a) guidelines

b) directions

c) templates

d) All of the mentioned

8. What access modifier is used when access modifier is not defined?

a) denominator

b) default

c) decimal

d) None of the mentioned

9. A Java class name has to begin with a _____.

a) letter

b) number

c) asterisk

d) None of the mentioned

10. An object is an _____ of a class.

a) integer value

b) instance

c) interpreter

d) None of the mentioned