Oops Concepts

———————————————————

What is Oops ?

- OOP Definition : Programming languages that use objects as a primary source to implement code.
- OOP Goal : Implement real-world entities like inheritance, hiding, and polymorphism in programming.
- OOP Benefit : Binds data and functions together, preventing access from other parts of the code.

———————————————————

Class

- Class Definition : A user-defined blueprint for creating objects with shared properties and methods.
- Class Purpose : Enables efficient object creation by defining common attributes and behaviours.
- Class Structure : Consists of modifiers (e.g., public), a capitalised class name, and a body enclosed in braces.

———————————————————

Object

- Object Definition: A basic unit in Object-Oriented Programming representing real-life entities.
- Object Components: Consists of state (attributes and properties), behaviour (methods and responses), and identity (unique name for interaction).
- Method Functionality: A collection of statements performing a specific task, potentially returning a result, and enabling code reuse.

———————————————————

Method n Method Parsing

- Method Definition: A collection of statements that perform a specific task and return a result.
- Method Parameters: Can be declared with or without arguments.

- Method Functionality: Takes input values, performs operations, and returns a result.

———————————————————

-> PILLAR OF OOPS

———————————————————

Abstraction

- Data Abstraction Definition: Showing only the important details of an object, hiding the unimportant details.
- Data Abstraction Example: A car is seen as a whole, not as its individual parts.
- Data Abstraction in Programming: Focusing on an object's important features and ignoring the details of how it works.

———————————————————

Encapsulation

- Encapsulation Definition: Wrapping up data under a single unit, binding code and data, and protecting data from external access.
- Encapsulation Mechanism: Hides variables or data in a class from other classes, accessible only through member functions.
- Encapsulation Implementation: Achieved by declaring variables as private and using public methods to set and get variable values.

———————————————————

Inheritance

- Inheritance in Java: Allows one class to inherit features (fields and methods) from another class using the 'extends' keyword.
- Superclass and Subclass: The class being inherited from is the superclass (base or parent class), while the inheriting class is the subclass (derived, extended, or child class).
- Reusability: Inheritance enables code reuse by allowing a subclass to inherit and build upon the existing code in a superclass.

———————————————————

Polymorphism

- Polymorphism Definition: The ability of object-oriented programming languages to differentiate between entities with the same name efficiently.

- Java Implementation: Java uses signatures and declarations to differentiate between entities with the same name.

- Polymorphism Benefit: Allows entities to appear in many forms.

———————————————————

-> METHOD OVERLOADING & OVERRIDING

———————————————————

Method Overloading

- Method Overloading Definition: Method overloading is a concept in programming where multiple methods with the same name but different parameter lists exist within a class.

- Method Overloading Purpose: Allows for code readability and flexibility by providing different implementations for the same method name based on parameter types.

- Method Overloading Example: In Java, you can have a method named “add” that takes two integers as parameters and returns an integer, and another “add” method that takes two doubles as parameters and returns a double.

———————————————————

Method Overriding

- Method Overriding Definition: A concept in object-oriented programming where a subclass provides a specific implementation for a method that is already defined in its superclass.

- Method Overriding Characteristics: The overridden method in the subclass must have the same name, return type, and parameter list as the method in the superclass.

- Method Overriding Purpose: Allows for different implementations of the same method based on the specific class in which it is called, enabling runtime polymorphism.

———————————————————

-> ADVANTAGES OF OOPS

———————————————————

- Code Reusability: OOP promotes code reusability through objects and classes, reducing duplication and enhancing efficiency.

- Code Organisation: OOP provides a clear and logical structure, making code easier to understand, maintain, and debug.

- Faster Development: OOP enables faster development by reusing existing code and creating modular components.

_____

CONCOLUSION

- OOPs Concept in Java: A powerful way to organise and write code using classes, objects, inheritance, polymorphism, encapsulation, and abstraction.

- Benefits of Java OOPs: Builds complex applications more efficiently, making code easier to manage, understand, and modify.

- Java OOPs Impact: Creates robust and scalable software solutions.