Object-Oriented Programming (OOP) in Java is a fundamental programming paradigm that organizes software design around data, or objects, rather than functions and logic. Java is inherently object-oriented, which means everything in Java is associated with classes and objects. The core concepts of OOP in Java include encapsulation, inheritance, polymorphism, and abstraction. Encapsulation allows developers to wrap data (variables) and code (methods) together as a single unit and restrict direct access to some of the object's components. This ensures better data integrity and security. Inheritance enables a class to inherit properties and behaviors from another class, promoting code reusability and method overriding for dynamic behavior.

Polymorphism in Java allows objects to take on many forms, most commonly through method overloading (compile-time) and method overriding (runtime), enabling flexibility and scalability in code. Abstraction, on the other hand, helps in hiding the internal implementation details and showing only the essential features of an object, which is achieved using abstract classes and interfaces. Java’s strong support for OOP makes it easier to manage large codebases, maintain software, and build modular, reusable, and robust applications. With tools like constructors, access modifiers, and interfaces, Java provides a structured and scalable way to apply OOP principles effectively in software development.