FACULTY OF ENGINEERING
Department of Telecommunications and Electronics Engineering

JAVA OOP

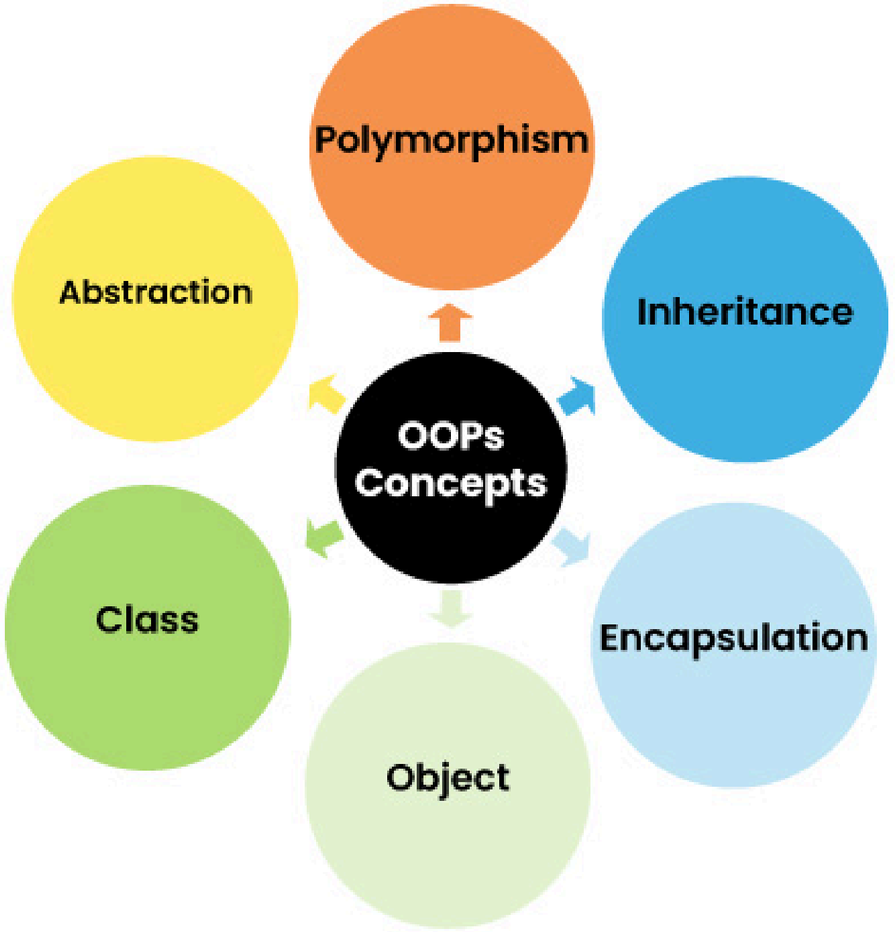
Basic Java Course

Advisor: Prof. Thap Thareoun

Present By : Ngouch HongCheng
: Saing LymChhun

INTRODUCTION

Object-Oriented Programming (OOP) in Java is a programming paradigm that uses objects and classes to structure software in a way that models real-world entities and relationships.



INTRODUCTION

Some Important uses of OOP in java

- **Modularity**: Encapsulation keeps data and methods together, making code more modular and secure.
- **Code Reusability**: Inheritance allows you to reuse and extend existing code easily.
- **Flexibility**: Polymorphism lets methods work differently based on the object, enhancing flexibility.
- **Maintainability**: The organized structure makes code easier to maintain and debug.
- **Collaboration**: Modular design allows team members to work independently on different parts of a project.

CLASS

A. DEFINITION

- A class in Java is defined using the class keyword, followed by the class name.
- The body of the class is enclosed in curly braces {}.

SYNTAX

```
public class Simulation {
//body of class will be put here
}
```

CLASS

B. ACCESS MODIFIER

- Access modifiers in Java control the visibility and accessibility of classes, fields, methods, and constructors.
- They play a crucial role in encapsulation, which is a fundamental principle of object-oriented programming (OOP).

CLASS

B. ACCESS MODIFIER - PUBLIC

- Classes: A class declared as public can be accessed from any other class.
- Fields/Methods/Constructors: Members declared as public can be accessed from any other class.

CLASS

B. ACCESS MODIFIER - PRIVATE

- Classes: The private modifier cannot be used with top-level classes. It is used with inner classes.
- Fields/Methods/Constructors: Members declared as private are accessible only within the same class.

CLASS

B. ACCESS MODIFIER - PROTECTED

- Classes: The protected modifier cannot be used with top-level classes.
- Fields/Methods/Constructors: Members declared as protected are accessible within the same package and by subclasses, even if they are in different packages.

CLASS

B. ACCESS MODIFIER - DEFAULT (PACKAGE-PRIVATE)

- Classes: A class with no access modifier (default) is accessible only within its own package.
- Fields/Methods/Constructors: Members with no access modifier (default) are accessible only within the same package.

CLASS

B. ACCESS MODIFIER

Access Modifier	Same Class	Same Package	Subclass	Other Packages
public	Yes	Yes	Yes	Yes
private	Yes	No	No	No
protected	Yes	Yes	Yes	No
default	Yes	Yes	No	No

END