# POLYMORPHISM

Polymorphism is a fundamental concept in object-oriented programming (OOP) that allows objects of different classes to be treated as objects of a common superclass. It enables the flexibility and extensibility of code by promoting code reusability and facilitating the creation of flexible, maintainable, and scalable software.

There are two main types of polymorphism: compile-time (static) polymorphism and runtime (dynamic) polymorphism.

Compile-time (Static) Polymorphism:

* Also known as method overloading or function overloading.
* Occurs during the compile-time phase of the program.
* Involves the creation of multiple methods in a class with the same name but different parameter lists.
* The appropriate method to be executed is determined by the number or type of arguments provided during the method call.

Example 1:

**class** Adder{

**static** **int** add(**int** a,**int** b){**return** a+b;}

**static** **int** add(**int** a,**int** b,**int** c){**return** a+b+c;}

}

**class** TestOverloading1{

**public** **static** **void** main(String[] args){

System.out.println(Adder.add(11,11));

System.out.println(Adder.add(11,11,11));

}}