**OOPS**

OO programming is designed so that real world concepts can be modeled in a computer program.

Objects are everywhere around us, while the class they belong to is not always obvious.

**Abstraction**

Abstraction is a design technique that focuses on the essential aspects of an entity and ignores less important aspects.

Abstraction is an important tool for simplifying a complex situation to a level where analysis, experimentation, or understanding can take place.

Abstraction is concerned with both the "attributes" and "behavior".

**Attributes** refer to the properties or characteristics associated with an entity. Attributes typically correspond to the data that is recorded for an object.

The "**behavior**" represents the set of actions that the object can perform. An action usually corresponds to an action that the real-world entity might perform.

**Encapsulation**

Encapsulation is the process of grouping methods and data together and hiding them behind a public interface. A class demonstrates good encapsulation by protecting variables with private or protected access, while providing more publicly accessible setter and/or getter methods.

**Inheritance**

Inheritance is an object-oriented concept that provides for the **reuse** and modification of an existing type in such a way that many types can be manipulated as a single type.

Inheritance is a simple concept which allows one class to extend another, or to inherit characteristics. When a class is extended a hierarchy is built up with the original class at the top and the classes which build on the class above it (the class at the top is known as the super-class) below. When a class is extended, to create a sub-class, all of the properties (variables and methods) of the original class still exist within the new class along with others which have been added.

In Java, inheritance is achieved with the ‘extends’ keyword.

**Polymorphism**

Polymorphism is the ability of objects to react differently when presented with different information, known as parameters.

In a functional programming language the only way to complete two different tasks is to have two functions with different names.

Object-oriented languages, which include Java, allow different methods to be run depending on what type of parameters are specified.

This ability of a method to react to different parameters is achieved by overriding, a number of methods are written (with the same name) but each one has a different set of input parameters.

Any Java object that can pass more than one IS-A test can be considered polymorphic.

Design Principles: Classes should be open for extension but closed for modification. [The Open-Close Principle (OCP)]

**Coupling**:

Coupling refers to the degree to which one class knows about or uses members of another class.

Loose coupling is the desirable state of having classes that are well encapsulated, minimize references to each other, and limit the breadth of API usage.

**Cohesion**:

The term cohesion is used to indicate the degree to which a class has a single, well-focused purpose.

Cohesion refers to the degree in which a class has a single, well-defined role or responsibility.

High cohesion is the desirable state of a class whose members support a single, well-focused role or responsibility.

OO Principles

Encapsulate what varies.

Code to interfaces rather to an implementation.

Each class in an application should have only one reason to change.

Classes are about behavior and functionality.

Use composition to assemble behaviors from other classes.