**Group Assignment**

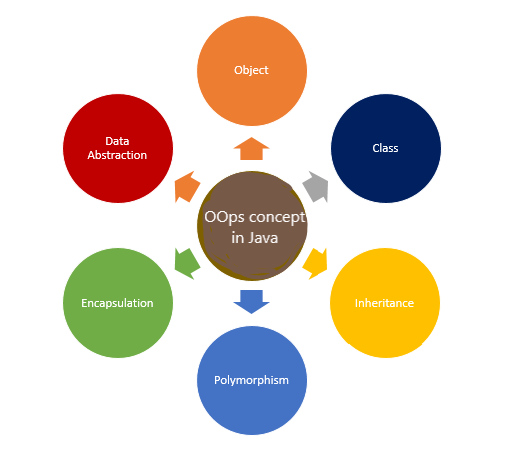
**OOPs concepts and advanced features**

Object-oriented programming is a core of [Java Programming](https://www.mygreatlearning.com/academy/learn-for-free/courses/java-programming?gl_blog_id=24671), which is used for designing a program using classes and objects. This can also be characterized as data controlling for accessing the code. In this type of approach, programmers define the [data type](https://www.mygreatlearning.com/blog/types-of-data/) of a [data structure](https://www.mygreatlearning.com/blog/data-structure-tutorial-for-beginners/) and the operations that are applied to the data structure.

OOps in java is to improve code readability and reusability by defining a Java program efficiently. The main principles of object-oriented programming are **abstraction, encapsulation, inheritance, and polymorphism**. These concepts aim to implement real-world entities in programs.

## **List of OOPs Concepts in Java**

* Objects
* Classes
* Object
* Class
* Abstraction
* Inheritance
* Polymorphism
* Encapsulation



**Objects:**

Objects are always called instances of a class which are created from class in java or any other language. They have states and behaviour.

These objects always correspond to things found in the real world, i.e., real entities. So, they are also called a run-time entity of the world. These are self–contained which consists of methods and properties which make data useful. Objects can be both physical and logical data. It contains addresses and takes up some space in memory. Some examples of objects are a dog, chair, tree etc.

**Classes:**

Classes are like object constructors for creating objects. The collection of objects is said to be a class. Classes are said to be logical quantities. Classes don’t consume any space in the memory. Class is also called a template of an object. Classes have members which can be fields, methods and constructors. A class has both static and instance initializers.

A class declaration consists of:

1. **Modifiers**: Can be public or default access.
2. **Class name:** Initial letter.
3. **Superclass:** A class can only extend (subclass) one parent.
4. **Interfaces:** A class can implement more than one interface.
5. **Body:** Body surrounded by braces, { }.

**Abstraction:**

Abstraction is a process which displays only the information needed and hides the unnecessary information. We can say that the main purpose of abstraction is data hiding. Abstraction means selecting data from a large number of data to show the information needed, which helps in reducing programming complexity and efforts.

**Advantages of Abstraction**

* It simplifies the process of seeing things in their entirety.
* Code duplication is avoided, and reusability is increased.
* Because just the most necessary information is shown to the user, it helps to enhance the security of an application or software.

**Inheritance:**

Inheritance is a method in which one object acquires/inherits another object’s properties, and inheritance also supports hierarchical classification. The idea behind this is that we can create new classes built on existing classes, i.e., when you inherit from an existing class, we can reuse methods and fields of the parent class. Inheritance represents the parent-child relationship.

**Single level**

**Multilevel**

**Hierarchical level**

**Hybrid inheritance**

**Polymorphism:**

Polymorphism refers to many forms, or it is a process that performs a single action in different ways. It occurs when we have many classes related to each other by inheritance. Polymorphism is of two different types, i.e., compile-time polymorphism and runtime polymorphism. One of the examples in Compile time polymorphism is that when we overload a static method in java. Run time polymorphism is also called a dynamic method dispatch is a method in which a call to an overridden method is resolved at run time rather than compile time. In this method, the overridden method is always called through the reference variable. By using method overloading and method overriding, we can perform polymorphism.

**Encapsulation:**

Encapsulation is one of the concepts in OOPs concepts; it is the process that binds together the data and code into a single unit and keeps both from being safe from outside interference and misuse. In this process, the data is hidden from other classes and can be accessed only through the current class’s methods. Hence, it is also known as data hiding. Encapsulation acts as a protective wrapper that prevents the code and data from being accessed by outsiders. These are controlled through a well-defined interface.

* **Data Hiding:** In this case, the user will be unaware of the class’s internal implementation. Even the user will have no idea how the class stores data in variables. He or she will only be aware that the values are sent to a setter method and that variables are initialised with that value.
* **Increased Flexibility:** Depending on our needs, we may make the variables of the class read-only or write-only.
* It also promotes**reusability** and makes it simple to alter to meet new needs.

**Constructors and Destructors:**

Constructors in most object-oriented languages have the same name as the class and are public. Constructors may be overloaded, which means that multiple argument lists can be used with the same name. The function Object() { [native code] } in PHP 5.0 is the function \_construct (). Normally, attribute values would be initialised in a function Object() { [native code] }. The \_destruct() method is optional, although it might be used to implement code that cleans up once an object is destroyed, such as shutting files or database connections.

**OOP Advantages**

* Complex things are modeled as repeatable, basic structures in OOP.
* Thus, OOP objects are reusable and may be utilized in several applications.
* Modularity for easier troubleshooting.
* Classes are easier to debug since they generally include all relevant information.
* Reuse of code through inheritance.

**Conclusion:**

Object-oriented programming necessitates planning and thinking about the program’s structure before starting to code and examining how to decompose the requirements into basic, reusable classes that you may utilize to create object instances. Overall, using OOP provides for more reusable data structures and saves time in the long run.