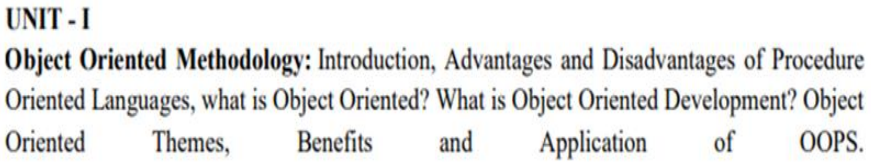
MARRI LAXMAN REDDY
Institute of Technology and Management
(Approved By AICTE, New Delhi & Affiliated to JNTU Hyderabad)
Dundigal,Medchal(M),Hyderabad,Telangana, India – 500 043.
(UGC - AUTONOMOUS)

OOPS THROUGH JAVA

UNIT-I-PART-I



UNIT - I

Object Oriented Methodology: Introduction, Advantages and Disadvantages of Procedure Oriented Languages, what is Object Oriented? What is Object Oriented Development? Object Oriented Themes, Benefits and Application of OOPS.

Object Oriented Methodology(OOM)

- Object-oriented methodology is a way of viewing software components and their relationships.
- Object-oriented methology.
- OOM on three characteristics that define object-oriented languages:
- Encapsulation,
- Polymorphism,
- Inheritance.

Why Object Oriented Methodology

- The main aim of Object Oriented Design (OOD) is to improve the quality and productivity of system analysis and design by making it more usable.
- In analysis phase, OO models are used to fill the gap between problem and solution.

Where is oops is used in Realtime

- OOP means Object Oriented programming
- OOP can also be used in manufacturing and designing applications as it allows people to reduce the efforts involved.

For instance, it can be used while designing blueprints and flowcharts.

Why it is called OOP?

- In basic terms, OOP is a programming pattern that is built around objects or entities, so it's called object-oriented programming.
- Object-Oriented Programming Languages each developer must know.
- It includes Python, Java, C, C#, Ruby, and so on

What is the most used OOP language?

- Java is one of the best and most widely used programming languages for OOP.
- Java has a large community with lots of resources and libraries, so it is easy for beginners to learn.
- Simula (1967) is generally accepted as being the first language with the primary features of an object-oriented language.

What is Object Oriented

As the name suggests, Object-Oriented Programming or OOPs refers to languages that use objects in programming.

Object-oriented programming aims to implement real-world entities like inheritance, hiding, polymorphism, etc in programming.

The main aim of OOP is to bind together the data and the functions that operate on them so that no other part of the code can access this data except that function.

OOPs Concepts:

- Class
- Objects
- Data Abstraction
- Encapsulation
- Inheritance
- Polymorphism
- Dynamic Binding
- Message Passing

What is Object Oriented Development

Object oriented development defines the life cycle of the system.

Object oriented development focuses on building a model of the software before starting the implementation and then go on adding detail to the model.

Why do we need Object oriented Programming

- To make the development and maintenance of projects more effortless.
- To provide the feature of data hiding that is good for security concerns.

- We can solve real-world problems if we are using object-oriented programming.
- It ensures code reusability.

Procedural Programming

Procedural Programming can be defined as a programming model which is derived from structured programming, based upon the concept of calling procedure.
Procedures, also known as routines, subroutines or functions, simply consist of a series of computational steps to be carried out.
During a program's execution, any given procedure might be called at any point, including by other procedures or itself.
Languages used in Procedural Programming:
FORTRAN, ALGOL, COBOL,

BASIC, Pascal and C.

Object-Oriented Programming

Object-oriented programming can be defined as a programming model which is based upon the concept of objects.
Objects contain data in the form of attributes and code in the form of methods. In object-oriented programming, computer programs are designed using the concept of objects that interact with the real world.
Object-oriented programming languages are various but the most popular ones are class-based, meaning that objects are instances of classes, which also determine their types.
Languages used in Object-Oriented Programming:
Java, C++, C#, Python,

PHP, JavaScript, Ruby, Perl,

Objective-C, Dart, Swift, Scala.

Procedural Programming vs Object-Oriented Programming

Below are some of the differences between procedural and object-oriented programming:

Procedural Oriented Programming	Object-Oriented Programming
In procedural programming, the program is divided into small parts called ***functions***.	In object-oriented programming, the program is divided into small parts called ***objects***.
Procedural programming follows a ***top-down approach***.	Object-oriented programming follows a ***bottom-up approach***.
There is no access specifier in procedural programming.	Object-oriented programming has access specifiers like private, public, protected, etc.
Adding new data and functions is not easy.	Adding new data and function is easy.
Procedural programming does not have any proper way of hiding data so it is ***less secure***.	Object-oriented programming provides data hiding so it is ***more secure***.
In procedural programming, overloading is not possible.	Overloading is p ossible in object-oriented programming.
In procedural programming, there is no concept of data hiding and inheritance.	In object-oriented programming, the concept of data hiding and inheritance is used.
In procedural programming, the function is more important than the data.	In object-oriented programming, data is more important than function.
Procedural programming is based on the ***unreal world***.	Object-oriented programming is based on the ***real world***.
Procedural programming is used for designing medium-sized programs.	Object-oriented programming is used for designing large and complex programs.
Procedural programming uses the concept of procedure abstraction.	Object-oriented programming uses the concept of data abstraction.

Procedural Oriented Programming	Object-Oriented Programming
Code reusability absent in procedural programming,	Code reusability present in object-oriented programming.
Examples: C, FORTRAN, Pascal, Basic, etc.	**Examples:** C++, Java, Python, C#, etc.

What is Object oriented Themes

Object Oriented: It means that we organize software as a collection of discrete objects that incorporate both data structure and behavior.

Benefits of OOPS

- ❖ OOP offers code reusability.
- ❖ Already created classes can be reused

without having to write them again.

- ❖ OOP facilitates the quick development of programs
- ❖ where parallel development of classes is possible.
- ❖ With OOP, programs are easier to test, manage and debug.

Applications of OOPS:-

Building complex systems and applications is made possible by OOPs, it enables modular, reusable, and maintainable programming.

1.Software development,

2.GUI development

3. Game development,

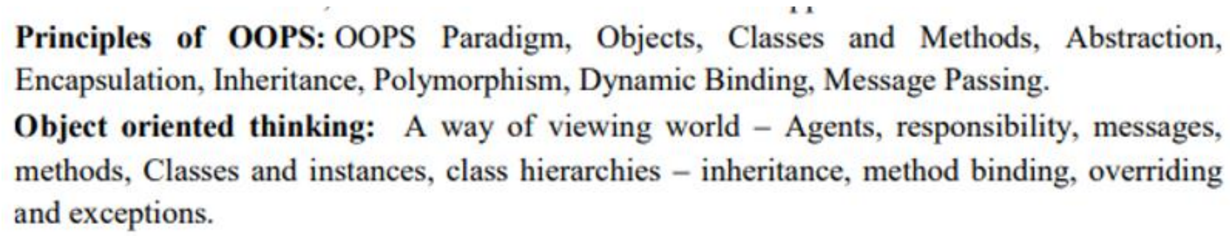
4.Mobile app development,

5. web development

6. database systems

7. AI/ML are some of the main applications of OOPs.

UNIT-I-PART-II



Principles of OOPS: OOPS Paradigm, Objects, Classes and Methods, Abstraction, Encapsulation, Inheritance, Polymorphism, Dynamic Binding, Message Passing.

Object oriented thinking: A way of viewing world – Agents, responsibility, messages, methods, Classes and instances, class hierarchies – inheritance, method binding, overriding and exceptions.

OOPs Concepts

Object-Oriented Programming is a paradigm that provides many concepts, such as **inheritance**, **data binding**, **polymorphism**, etc.

Simula is considered the first object-oriented programming language.

The programming paradigm where everything is represented as an object is known as a truly object-oriented programming language.

Smalltalk is considered the first truly object-oriented programming language.

The popular object-oriented languages are Java, C#, PHP, Python, C++, etc.

The main aim of object-oriented programming is to implement real-world entities, for example, object, classes, abstraction, inheritance, polymorphism, etc.

OOPs (Object-Oriented Programming System)

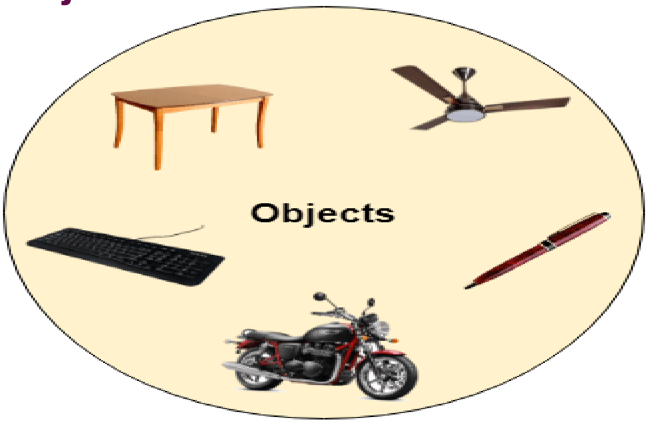
Object means a real-world entity such as a pen, chair, table, computer, watch, etc.

Object-Oriented Programming is a methodology or paradigm to design a program using classes and objects.

It simplifies software development and maintenance by providing some concepts:

- Object
- Class
- Inheritance
- Polymorphism
- Abstraction
- Encapsulation

Object



Any entity that has state and behavior is known as an object. For example, a chair, pen, table, keyboard, bike, etc. It can be physical or logical.

An Object can be defined as an instance of a class. An object contains an address and takes up some space in memory. Objects can communicate without knowing the details of each other's data or code. The only necessary thing is the type of message accepted and the type of response returned by the objects.

Example: A dog is an object because it has states like color, name, breed, etc. as well as behaviors like wagging the tail, barking, eating, etc.

Class

Collection of objects is called class. It is a logical entity.

A class can also be defined as a blueprint from which you can create an individual object. Class doesn't consume any space.

Inheritance

When one object acquires all the properties and behaviors of a parent object, it is known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism.

Polymorphism

If *one task is performed in different ways*, it is known as polymorphism.

For example: to convince the customer differently, to draw something, for example, shape, triangle, rectangle, etc.

In Java, we use method overloading and method overriding to achieve polymorphism.

Another example can be to speak something; for example, a cat speaks meow, dog barks woof, etc.

Abstraction

Hiding internal details and showing functionality is known as abstraction. For example phone call, we don't know the internal processing.

In Java, we use abstract class and interface to achieve abstraction.

Encapsulation

Binding (or wrapping) code and data together into a single unit are known as encapsulation. For example, a capsule, it is wrapped with different medicines.

Dynamic Binding:-In Dynamic binding compiler does not decide the method to be called.

Overriding is a perfect example of dynamic binding.
In overriding both parent and child classes have the same method.
Dynamic binding is also called **Late binding**.

Message Passing**:**
Message Passing in terms of computers is communication between processes.
It is a form of communication used in object-oriented programming as well as parallel programming.
Message passing in Java is like sending an object i.e. message from one thread to another thread.
It is used when threads do not have shared memory and are unable to share monitors or semaphores or any other shared variables to communicate.
Suppose we consider an example of producer and consumer, likewise what producer will produce, the consumer will be able to consume that only.
We mostly use Queue to implement communication between threads.

Let’s see the differences between Dynamic Binding and Message Passing:

Dynamic Binding	Message Passing
Binding refers to the linking of a procedure call to the code to be executed in response to the call.	The process of programming in which communication is involved is known as message passing.
It allows execution of different codes using the same object at runtime.	It allows developing communication between objects.
The usage of Dynamic binding to allow executing different codes using the same object at runtime.	It involves three basic steps. They are: • Creating classes • Creating objects • Establishing communication among objects.
It is the method of linking a procedure call to the relevant code that will be executed only at runtime.	Message passing is the method of exchanging message between objects in Object Oriented Programming.
The code associated with the procedure is not known until the program is executed. This process is known as late binding.	Message passing involves name of the objects, the name of the function and the information to be sent.
A function call associated with a polymorphic reference depends on the dynamic type of that reference.	Communication with an object is feasible as long as it is alive.
In dynamic binding, only at run time the code matching the object under current reference will be called.	A message for an object is a request for execution of a procedure, and therefore invoke a function in the receiving object that generates the desired result.
In short, dynamic binding occurs during runtime.	Message passing occurs between two processes.
Dynamic binding is also known as dynamic dispatch, late binding or	Message passing is also known as message exchanging.

Dynamic Binding	Message Passing
run-time binding.	
Object based programming does not support dynamic binding.	Object based programming support message passing.