Object-Oriented Thinking (OOT) and Object-Oriented Programming (OOP) are two basic ideas in software development that provide a way to structure and organize code.

Object-Oriented Thinking (OOT):

Object-Oriented Thinking is a mindset or paradigm for designing and modeling software based on real-world entities and their interactions. It centers on the concept of “objects” which can be thought as instances of a real world or abstract entity.

These objects have both data (attributes or properties) and behavior (methods or functions) associated with them.

Some key principles behind Object-Oriented thinking include the following:

1. Encapsulation: This compiles together data (attributes) as well as methods that operate on the data into one unit i.e. an object. This helps in hiding an object’s internal details, but allows access only to those parts that are necessary.

2. Abstraction: The art of making very complicated systems seem simple through modeling classes based on their essential properties and behaviors while ignoring unimportant things. Thus, developers are allowed to concentrate on what an object does rather than how it does it.

3. Inheritance: In this case, new classes get created by obtaining attributes and behaviors from already existing ones. As it encourages reusing codes, it helps build a class hierarchy.

4. Polymorphism: Objects from different classes can be treated as objects of the same class allowing for extensibility and flexibility. Thus, Polymorphism makes it possible to represent

many forms using just one interface.

Object-Oriented Programming (OOP):

Object-Oriented Programming is a programming paradigm that implements the principles of Object-Oriented Thinking hence serves as a method of organizing code designing code using objects and classes. In OOP software is structured around objects each with its own set of data and behavior.

The important characteristics of Object Oriented Programming include:

1. Classes and Objects: Classes define the structure and behavior of an object. Objects, on the other hand, are instances of classes.

2. Encapsulation: The practice of wrapping up data (attributes) and methods that work on the data within a class so as to prevent its direct access and modification from outside the class.

3. Inheritance: This refers to a way of making a class have properties or behaviors that it's inherited from another class. For this reason, it is a means to promote using already developed codes and building classes in a hierarchy.

4. Polymorphism: This provides for utilizing one interface for representing different types of objects. In order to achieve this approach, both method overloading and method overriding can be used.

Object-Oriented Thinking is a conceptual approach to software design, emphasizing the modeling of entities and their interactions, while Object-Oriented Programming is the

practical implementation of these concepts using programming languages and tools.

Java, C++, Python, and others are object-oriented programming (OOP) languages which provide syntax as well as features for implementing OOP concepts. In designing modular, reusable, and maintainable code, the principles of OOT and OOP are applied by developers in using these languages.

Related Posts:

1. Abstraction and encapsulation
2. Object Oriented Programming & Methodolog Viva Voce
3. How to install compiler for code blocks
4. Object Oriented Programming
5. Differences between Procedural and Object Oriented Programming
6. Features of Object Oriented Paradigm
7. Inheritance in Object Oriented Programming
8. Object Oriented Programming
9. Difference Between Object-Oriented Programming (OOP) and Procedural Programming
10. features of Object oriented paradigm
11. Merits and demerits of Object Oriented methodology
12. Concept of Objects: State, Behavior & Identity of an object
13. Access modifiers
14. Static members of a Class
15. Instances in OOP
16. Message Passing in OOP
17. Construction and destruction of Objects