Fundamentals of Object-Oriented Programming (OOP)

Object-oriented programming (OOP) combines a group of variables (data types) and functions into a single unit called an object. This type of programming model is suited for large and complex programs. In today's changing atmosphere of technology, you need to update the program, and writing the same piece of code again will only overstuff the program.

OOP classifies the code into specific classes, data types, and functions as its objects.

OOP is popular today because of its four main concepts :-

1. Encapsulation
2. Abstraction
3. Inheritance
4. Polymorphism

Encapsulation

Encapsulation is the process of combining data types and functions into a single unit called a class. Consider a real-life example of an organization where there are different sections like account section, management section, sales and marketing section, finance section, etc.

The account section maintains records of the cash flow and transactions, make reports, and has some key roles like account receivable, account payable, etc. The sales and marketing section keeps all the records of sales.

Now, if the account section wants all the data about sales in a particular month. It will not have direct access to it and will need to contact the sales section requesting it to give sales details. Similarly, encapsulation covers the data members under a particular class by making it private and only a method can give its access to the reference object.

▪ Abstraction

Data abstraction is one of the most important features of OOP. Abstraction is providing the data to the outside world without going into its implementation details. For example, a man driving a car. He knows pressing the accelerator would increase the car speed but not its mechanism. It refers to abstraction.

▪ Abstraction using classes

A class can decide which data members to be made public and which are not.

▪ Abstraction using header files

One more type of abstraction can be header files. When we need to call any standard function, we can simply import the required class and call that function without going through its algorithm details.

▪ Inheritance

Using the inheritance concept, programmers can lessen the repetitive code. Inheritance allows a class to derive properties from its parent class. For example, elements of HTML code include a text box, checkbox, and select field have some properties in common so instead of redefining the properties and functions every time, you can define them once in a generic object and reduce unnecessary code.

▪ Polymorphism

Polymorphism is the ability of an object to perform a single action in different ways. There are two types of polymorphism :-

- **Dynamic Polymorphism**

This is also called “Run-time Polymorphism” and “Method-overriding”. It is a process in which a call to an overridden function is resolved at the run time.

If the same function is used in both parent and child class, the child class method overrides the parent class method when called with a reference (object) of the parent class.

- **Static Polymorphism**

Binding is static, private, and final methods always occur at the compile time as these methods cannot be overridden. It is also known as “Method-overloading”. Method-overloading can have more than one method with the same name but different arguments. Here is when the compiler calls a method depending upon the parameters.

Object-oriented programming allows maintaining and modifying the existing code easily. It divides the long program which is difficult for developers to modify into small modules called classes. It provides code-reusability reducing unnecessary code which is helpful to any developer in solving problems effectively.