1 Lesson 1: Concepts of Object-Oriented Programming

1.1 Introduction

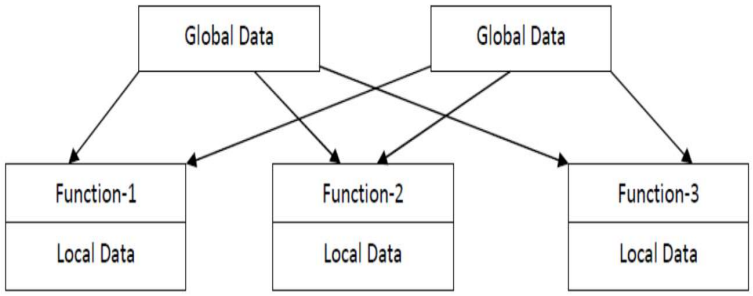
Object-Oriented Programming (OOP) is the term used to describe a programming approach based on objects and classes. The object-oriented paradigm allows the organization of software as a collection of objects that consist of both data and behaviour. This is in contrast to conventional functional programming practice / procedure-oriented programming (POP) that only loosely connects data and behaviour.

1.2 Procedure Oriented Programming

Conventional programming, using high level languages such as COBOL, FORTRAN, Pascal, and C, is commonly known as procedure-oriented programming (POP).

In the procedure-oriented approach, the problem is viewed as a sequence of things to be done such as reading, calculating and printing. A number of functions are written to accomplish these tasks.

- The primary focus is on functions (algorithm) rather than data
- Programs are divided into individual modules
- Modules are independent of each other and have their own local data
- Modules can work with its own data as well as with the data passed to it.

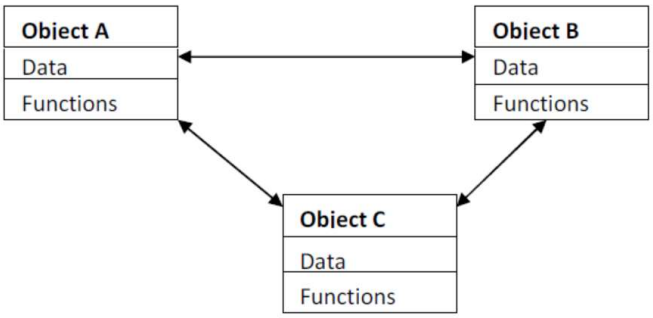


1.3 Object Oriented Programming (OOP)

OOP was developed to overcome the drawbacks of procedural and modular programming. It is now widely accepted that OOP is the most important and powerful way of creating software.

- Emphasis is on data rather than procedure (algorithm).
- Programs are divided into what are known as objects.
- Data is hidden and cannot be accessed externally.
- Objects may communicate with each other through functions.
- New objects can be easily added whenever necessary.

Examples of OOP Languages: C++, Java, C#, Python etc.



1.4 Concepts of Object-Oriented Programming

1.4.1 Objects

Objects are the basic runtime entities in an object-oriented system, that represents both data and associated functions in a single unit. An object may represent a person, a place, a bank account, a table of data or any item that the program has to handle.

1.4.2 Class

Object contains data, and code to manipulate that data. The entire set of data and code of an object can be made a user-defined data type with the help of a class. A class is a **user defined data type** that represents a group of similar objects.

1.4.3 Data Encapsulation

The wrapping up of data and functions into a single unit is known as encapsulation.

The data is not accessible to the outside world, only those function which are wrapped in the unit (object) can access it. These functions provide the interface between the object's data and the program.

This insulation of the data from direct access by the program is called data hiding or information hiding.

1.4.4 Data Abstraction

Abstraction refers to the act of representing essential features without including the background details or explanations. Since classes use the concept of data abstraction, they are known as Abstract Data Types (ADT).

1.4.5 Modularity

Modularity is designing a system that is divided into a set of functional units (named modules) that can be composed into a larger application.

1.4.6 Inheritance

Inheritance is the process by which objects of one class acquire the properties of objects of another class. It is the technique of building new classes (derived class) from an existing class. In OOP, the concept of inheritance provides the idea of reusability- which means a new class with additional features can be built based on an existing class without modifying the existing class, while avoiding building everything from scratch (Re-inventing the wheel). Inheritance is transitive in nature.

1.4.7 Polymorphism

Polymorphism, a Greek term means to ability to take more than one form. An operation may exhibit different behaviors in different instances. These behaviors depend upon the type of data used in the operation. The process of making a function to exhibit different behaviors in different instances is known function overloading.

For example, consider the operation of addition for two numbers; the operation will generate a sum. If the operands are of type string, then the operation would produce a third string by concatenating the first string with the second string.

The process of making an operator to exhibit different behavior in different instances is known operator overloading

1.5 Advantages of OOP

- Re-usability: A class can be written once and reused as many times
- Redundancy: Inheritance is the good feature for data redundancy. If you need a same functionality in multiple classes, you can write a common class for the same functionality and inherit that class to sub class.
- Easy Maintenance: It is easy to maintain and modify existing code as new objects can be created with small differences to existing ones.
- Security: Using data hiding and abstraction only necessary data will be exposed thus maintains the security of data.

1.6 Disadvantages of OOP

- Size: Object Oriented Programs are much larger than other programs.
- Effort: Object Oriented Programs require a lot of work to create.
- Speed: Object Oriented Programs are slower than other programs, because of their size.