CHAPTER 1: INTRODUCTION TO OBJECT ORIENTED PROGRAMMING

Define Object Oriented Programming

A type of programming in which programmers define not only the data type of a data structure, but also the types of operations (functions) that can be applied to the data structure. In this way, the data structure becomes an *object* that includes both data and functions. In addition, programmers can create relationships between one object and another. For example, objects can *inherit* characteristics from other objects.

Object-oriented programming (**OOP**) is a programming paradigm based on the concept of "objects", which are data structures that contain data, in the form of fields, often known as *attributes;* and code, in the form of procedures, often known as *methods.* A distinguishing feature of objects is that an object's procedures can access and often modify the data fields of the object with which they are associated (objects have a notion of "this"). In object-oriented programming, computer programs are designed by making them out of objects that interact with one another. There is significant diversity in object-oriented programming, but most popular languages are class-based, meaning that objects are instances of classes, which typically also determines their type.

Evolution of Object Oriented Programming

The object-oriented paradigm took its shape from the initial concept of a new programming approach, while the interest in design and analysis methods came much later.

- The first object–oriented language was Simula (Simulation of real systems) that was developed in 1960 by researchers at the Norwegian Computing Center.
- In 1970, Alan Kay and his research group at Xerox PARK created a personal computer named Dynabook and the first pure object-oriented programming language (OOPL) - Smalltalk, for programming the Dynabook.
- In the 1980s, Grady Booch published a paper titled Object Oriented Design that mainly presented a design for the programming language, Ada. In the ensuing editions, he extended his ideas to a complete object–oriented design method.
- In the 1990s, Coad incorporated behavioral ideas to object-oriented methods.

The other significant innovations were Object Modelling Techniques (OMT) by James Rumbaugh and Object-Oriented Software Engineering (OOSE) by Ivar Jacobson.

Programming paradigms

A **programming paradigm** is a fundamental style of computer **programming**, a way of building the structure and elements of computer programs.

The following are considered the main programming paradigms. There is inevitably some overlap in these paradigms but the main features or identifiable differences are summarized below:

- ✓ **Imperative programming** – defines computation as statements that change a program state
- ✓ **Procedural programming**, structured programming – specifies the steps the program must take to reach the desired state.
- ✓ **Structured programming** (sometimes known as *modular programming*) is a subset of procedural programming that enforces a logical structure on the program being written to make it more efficient and easier to understand and modify. Modular programming is the process of subdividing a computer program into separate sub-programs.
- ✓ **Declarative programming** – defines computation logic without defining its control flow.
- ✓ **Functional programming** – treats computation as the evaluation of mathematical functions and avoids state and mutable data
- ✓ **Object-oriented programming** (OOP) – organizes programs as *objects*: data structures consisting of datafields and methods together with their interactions.
- ✓ **Event-driven programming** – the flow of the program is determined by events, such as sensor outputs or user actions (mouse clicks, key presses) or messages from other programs or threads.
- ✓ **Automata-based programming** – a program, or part, is treated as a model of a finite state machine or any other formal automata.

Merits and demerits of OOP

Object-Oriented Programming has the following advantages over conventional approaches:

- ✓ OOP provides a clear modular structure for programs which makes it good for defining abstract datatypes where implementation details are hidden and the unit has a clearly defined interface.

- ✓ OOP makes it easy to maintain and modify existing code as new objects can be created with small differences to existing ones.

- ✓ OOP provides a good framework for code libraries where supplied software components can be easily adapted and modified by the programmer. This is particularly useful for developing graphical user interfaces.

Examples of object oriented languages

Many of the most widely used programming languages are multi-paradigm programming languages that support object-oriented programming to a greater or lesser degree, typically in combination with imperative, procedural programming. Significant object-oriented languages include C++, Objective-C, Smalltalk, Delphi, Java, C#, Perl, Python, Ruby and PHP.

Object Oriented Databases (OODBs)

An object-oriented database management system (OODBMS) is a database management system that supports the creation and modeling of data as objects. OODBMS also includes support for classes of objects and the inheritance of class properties, and incorporates methods, subclasses and their objects. Most of the object databases also offer some kind of query language,