Introduction to Object Oriented Programming

The 1960s gave birth to structured programming. This is the method of programming championed by languages such as C. With the advent of languages such as c, structured programming became very popular and was the main technique of the 1980's. Structured programming was a powerful tool that enabled programmers to write moderately complex programs fairly easily. However, as the programs grew larger, even the structured approach failed to show the desired result in terms of bug-free, easy-to- maintain, and reusable programs. To solve these problems, a new way to program was invented, called *object-oriented programming* (OOP).

Object-oriented programming (OOP) is a computer programming model that organizes software design around data, or objects, rather than functions and logic.

Object Oriented programming (OOP) is a programming paradigm that relies on the concept of **classes** and **objects**. It is used to structure a software program into simple, reusable pieces of code blueprints (usually called classes), which are used to create individual instances of objects. There are many object-oriented programming languages including JavaScript, C++, Java, and Python.

Object Oriented Programming (*OOP*) is an approach to program organization and development that attempts to eliminate some of the pitfalls of conventional programming methods by incorporating the best of structured programming features with several powerful new concepts. It is a new way of organizing and developing programs and has nothing to do with any particular language. However, not all languages are suitable to implement the OOP concepts easily.

Procedural Oriented Programming Vs Object Oriented Programming

All computer programs consist of two elements: code and data. Furthermore, a program can be conceptually organized around its code or around its data. That is, some programs are written around — *"what is happening"* and others are written around —*"who is being affected"*. These are the two paradigms that govern how a program is constructed. The first way is called the *process- oriented* model or *procedural* language. This approach characterizes a program as a series of linear steps (i.e. code). The process-oriented model can be thought of as code acting on data. Procedural languages such as C employ this model to considerable success.

To manage increasing complexity, the second approach, called *object-oriented* programming, was designed. Object-oriented programming organizes a program around its data (i.e. objects) and a set of well-defined interfaces to that data. An object-oriented program can be characterized as data controlling access to code.

Difference between POP and OOP

	Procedure Oriented Programming	Object Oriented Programming
Divided Into	Program is divided into small parts called ***functions*.**	Program is divided into parts called ***objects.***
Importance	Importance is given to functions as well as sequence of actions to be done.	Importance is given to the data rather than procedures or functions because it works as a real world.
Approach	It follows Top-down approach.	It follows Bottom-up approach.
Data Moving	Data can move freely from function to function in the system.	Objects can communicate with each other through member functions.
Expansion	To add new data and function in POP is not so easy.	It provides an easy way to add new data and function.
Data Access	Most function uses Global data for sharing that can be accessed freely from function to function in the system.	Data cannot move easily from function to function, it can be kept public or private so we can control the access of data.
Data Hiding	It does not have any proper way for hiding data so it is less secure.	It supports data hiding so provides more security.
Overloading	In POP, Overloading is not possible.	In OOP, overloading is possible in the form of Function Overloading and Operator Overloading.
Inheritance	No such concept of inheritance in POP	Inheritance is allowed in OOP
Access Specifiers	It does not have any access specifiers.	It has access specifiers named Public, Private, Protected.
Examples	C, BASIC, FORTRAN, Pascal, COBOL.	C++, JAVA, C#, Smalltalk, Action Script.

Principles of OOP

It is necessary to understand some of the concepts used extensively in object-oriented programming. These include:

- Object
- Class
- Encapsulation
- Data abstraction
- Inheritance
- Polymorphism
- Dynamic binding
- Message passing

Object

Object is the basic run time entity in an object-oriented system. It may represent a person, a place, a bank account, a table of data, vectors, time and lists. Objects have states and behaviors. Example: A dog has states - color, name, breed as well as behaviors – wagging the tail, barking, eating. An object is an instance of a class.

In OOP, A problem is analyzed in term of objects and the nature of communication between them. Objects should be chosen such that they match closely with the real-world objects. Objects take up space in the memory and have an associated address like a *structure* in
C. When a program is executed, the objects interact by sending messages to one another.

For example, if "customer" and "account" are to object in a program, then the *customer* object may send a message to the *account* object requesting for the bank balance. Each object contain data, and code to manipulate data. Below figure shows two notations that are popularly used in object oriented analysis and design.

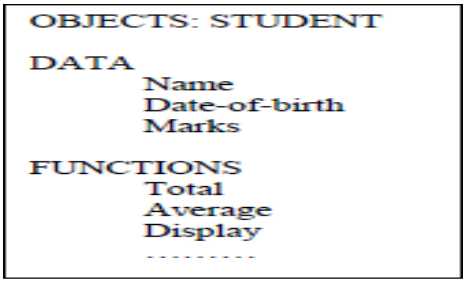


Fig. representing an object

Class

A class is a collection of similar objects that share the same properties, methods, relationships and semantics. A class can be defined as a template/blueprint that describes the behavior/state of the object.

Encapsulation

Definition: The process binding (or wrapping) code and data together into a single unit is known as Encapsulation. For example: capsule, it is wrapped with different medicines.

Capsule

- Java supports the feature of Encapsulation using *classes*.
- The data in the class is not accessed by outside world.
- The functions that are defined along with the data within the same class are allowed to access the data.
- 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*.
- A class defines the structure and behavior (data and code) that will be shared by a set of objects. Each object of a given class contains the same structure and behavior defined by the class.
- The objects are referred to as instances of a class. Thus, a class is a logical construct; an object is physical reality.

Data abstraction

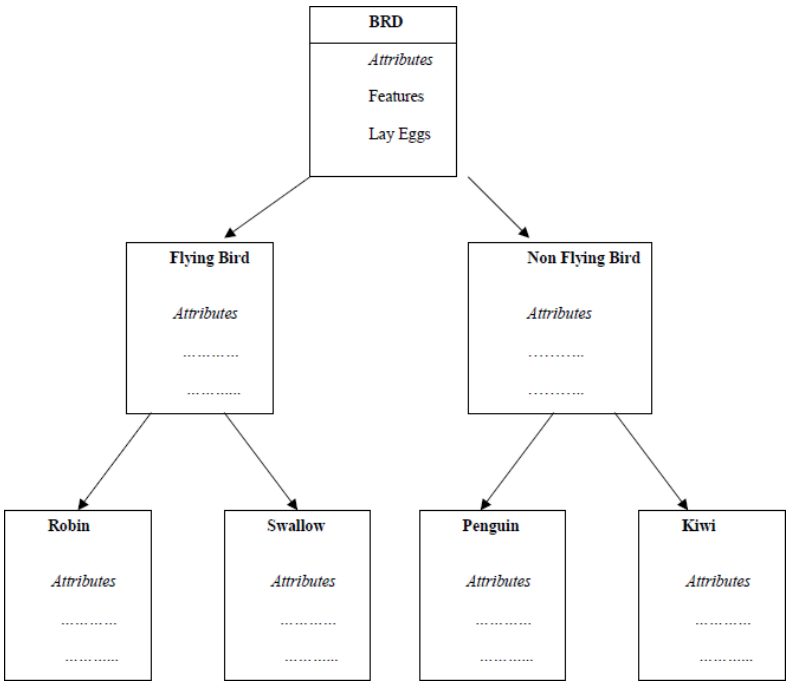
Definition: It a process of providing essential features without providing the background or implementation details. For example: It is not important for the user how TV is working internally, and different components are interconnected. The essential features required to the user are how to start, how to control volume, and change channels.
In java, we use *abstract class* and *interface* to achieve abstraction.

Inheritance

It a process by which an object of one class acquires the properties and methods of another class. It supports the concept of *hierarchical classification*. For example, the bird, 'robin' is a part of class 'flying bird' which is again a part of the class 'bird'. The principal behind this sort of division is that each derived class shares common characteristics with the class from which it is derived as illustrated in below figure.

In OOP, the concept of inheritance provides the idea of *reusability*. This means that we can add additional features to an existing class without modifying it. This is possible by deriving a new class from the existing one. The new class will have the combined feature of both the classes.

The Class from which the properties are acquired is called *super class*, and the class that acquires the properties is called *subclass*. This is mainly used for *Method Overloading* and *Code reusability*.



Polymorphism

Polymorphism is another important OOP concept. Polymorphism, a Greek term, means the ability to take more than one form. An operation may exhibit different behavior is different instances. The behavior depends upon the types of data used in the operation.
In Java, we use *method overloading* and *method overriding* to achieve polymorphism.

For example can be to speak something e.g. cat speaks meow, dog barks woof , duck speaks quack, etc.

Advantages of OOP

OOP offers several benefits to both the program designer and the user. Object-Orientation contributes to the solution of many problems associated with the development and quality of software products. The new technology promises greater programmer productivity, better quality of software and lesser maintenance cost. The principal advantages are:

- It presents a simple, clear and easy to maintain structure.
- Software complexity can be easily managed.
- It enhances program modularity since each object exists independently.
- New features can be easily added without disturbing the existing one.
- Objects can be reused in other program.
- It allows designing and developing safe programs using the data hiding.
- Through inheritance, we can eliminate redundant code extend the use of existing classes.
- It is easy to partition the work in a project based on objects.
- Object-oriented system can be easily upgraded from small to large system.
- Message passing techniques for communication between objects makes to interface descriptions with external systems much simpler.

Applications of OOP

Applications of OOP are beginning to gain importance in many areas. The most popular application of object-oriented programming, up to now, has been in the area of user interface design such as window. Hundreds of windowing systems have been developed, using the OOP techniques. The promising areas of application of OOP include:

- Real-time systems
- Simulation and modeling
- Object-oriented data bases
- Hypertext, Hypermedia, and Expertext
- AI and expert systems
- Neural networks and Parallel programming
- Decision support and Office automation systems
- CAM/CAD Systems