OBJECT ORIENTED PROGRAMMING.

Definition

It is a programming approach that is used to solve real-world problems by perceiving all entities as objects bearing the identity (name), state (physical appearance) and characteristic (behavior.

Real life example

Object oriented programming can be easily understood from animation (cartoon) programming point of view. Consider, for instance, the development of a cartoon program like that of "Animal Kingdom." Here, the programmer will be perceiving everything existing in the program as objects. For example, the animal kingdom will contain such objects as Lion, Elephant, Rabbit, Trees and water amongst other.

Basic concepts used in object-oriented programming

1. **Object**

- ❖ The dictionary definition of an object is: "*something that is or is capable of being seen, touched of otherwise sensed*."
- ❖ A program is a set of objects telling each other what to do by sending messages.
- ❖ Everything in an object
- ❖ Common examples of objects in your environment include chair, table, phone, pen, book, etc.
- ❖ Each object has its own memory (made up by other objects)
- ❖ Every object has a type
- ❖ When designing application programs, such objects as employee, customer, instructor and student amongst others are commonly worked on

An object is an identifiable entity with 3 features i.e. ***identity***, ***state*** and ***behavior***.

- ***Identity*** (a unique reference) e.g. student admission number, employee number, etc.
- ***State*** is the physical appearance of the object, for example, Tall, black, Male or Female Elephant.
- State is also referred to as a characteristic
- ***Behavior*** is or are set of things that an object can do and that correspond to functions that act on the object's data (attributes).
- For example, a dog would bear such characteristics as walking, eating and parking.
- In OOP, behavior is also referred to as method

2. **Attribute(s)**

- ❖ An object always bear some characteristics known as attribute(s) that give more details about the object
- ❖ For example, the attributes of the object employee will include Employee-Name, Age and Sex
- ❖ Attribute(s) are, therefore, characteristics that give more details about an object.
- ❖ When the attributes of an object are worked on it yields something that is known as object instance.
- ❖ For example, the real name of an employee like "Peter" is the attribute instance.

3. **Class**

- A class is just a collection of data variables of different types which are combined with functions that act on them (data).
- A class is a blueprint/template that defines the form of an object.
- Class is also referred to as an Abstract Data Type (ADT)
- A class/ADT encloses/encapsulate/bundles, both the data variables and functions through a process known as encapsulations
- For example, when developing a calculator program, the programmer will put together all operands (data) and operators in one class. Such expression as $X + Y = Z$, could be expressed in a class a follows;

  class Addition { int x, y, z; int Add ( ); };

Here, variables ***x***, ***y*** and ***z*** are ***data members*** while ***Add*** is a ***method***

Characteristics/Building Blocks of OOP

1. Encapsulation

- Encapsulation is the process of binding/wrapping up of data and functions together into a single unit
- With encapsulation, hiding of information is achieved, hence ensuring that data structures and operators are used as intended and protect the code from outside interference and misuse

2. Data Abstraction

- Abstraction refers to the act of hiding the internal details of an object while presenting the essential features.
- **Abstraction** allows us to represent complex real world in simplest manner. It is process of identifying the relevant qualities and behaviors an object should possess, in other word represent the necessary feature without representing the back ground details.
- With abstraction, the hidden data and methods can be changed without affecting the "outside world"

3. Inheritance

- Inheritance is the mechanism by which one class (derived class) possesses the characteristics of another class (bass class).

4. Polymorphism

- Polymorphism is derived from two Latin words ***poly*** which means ***many*** and ***morphos*** that means ***forms***
- It is the ability of an object to exhibit different behaviors in different instances.
- For example, when developing a calculator program, a programmer can use the method ***calculate( )*** to come up with such operations as ***add( )***, ***substract( )***, ***multiply( )*** and ***divide( ).***

5. Dynamic Binding:

- Dynamic binding means that the code associated with a given procedure call is not known until the time of the call at run-time.

- ❖ Binding refers to the linking of a procedure call to the code to be executed in response to the call.

6. Message passing

- ❖ OOP objets are able to communicate with one another through message passing

Features of object-oriented programming are:

- ❖ Emphasis is on data rather than procedure.
- ❖ Programs are divided into what are known as objects.
- ❖ Data structures are designed such that they characterize the objects.
- ❖ Follows bottom-up approach in program design.

Advantages and Disadvantages of Object-Oriented Programming (OOP)

Advantages

1. Improved software maintainability: object oriented software is also easier to maintain since it can be updated in case of issues without a need to make large-scale
2. Faster development: Reuse enables faster development.
3. Lower cost of development due to code re-usability
4. Leads to development of high-quality software
5. Function and data both are tied together in a single unit.
6. Data is not accessible by external functions as it is hidden.
7. Objects may communicate with each other only through functions.
8. It is easy to add new data and functions whenever required.
9. It can model real world problems very well.

Disadvantages of object-oriented programming:

1. Steep learning curve: The thought process involved in object-oriented programming may not be natural for some people, and it can take time to get used to it.
2. Larger program size: Object-oriented programs typically involve more lines of code than procedural programs.
3. Slower programs: Object-oriented programs are slower as they typically require more instructions to be executed.
4. Not suitable for all types of problems

Types of OOP languages are:

- ❖ C++
- ❖ C#
- ❖ Java
- ❖ Python

- PHP
- Perl
- Ruby
- Smalltalk

History of OOP Languages

- The terms "objects" and "oriented" were introduced to programming field in between 1950s and early 1960s in Massachusetts Institute of Technology (MIT).
- In 1966, Simula-1 was introduced as the first object oriented programming language.
  Simula brought in such programming features as that if objects, classes and some form of inheritance
- In early 1970's, Smalltalk which turned out to be a trully O-O language was introduced
- Smalltalk supported inheritance and the concept of sending messages between objects.
- Smalltalk introduced other concepts such as browsers, windows, and pop-up menus.
- In 1982, Bjarne Stroustrup combined some features of C-language and Simula to come up with C++ language that was initially known as "C with Classes"