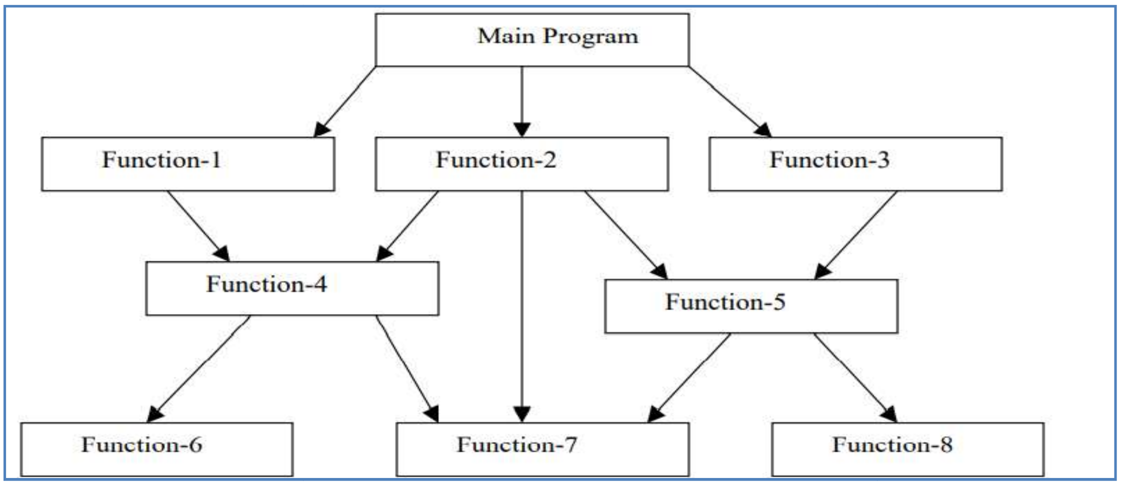
Week-2: Introduction to OOP

Introduction-Fundamentals of Object Oriented Programming:

Procedure Oriented Approach:

- In the POA, the problem is viewed by a sequence of things to be done, such as reading, calculating, printing such as COBOL, FORTRAN and C. A number of functions are written to accomplish these tasks. In POA primary focus is on functions.
- POP basically consists of writing a list of instructions for the computer to follow and organizing these instructions into groups known as functions.
- While we concentrate on the development of functions, a very little attention is given to the data that are being used by various functions. In a multifunction program many important data items are placed as global, so that they may be accessed by all the functions.
- Each function may have its own set of local data's. Global data are more vulnerable to inadvertent changes by a function. In a large program, it is very difficult to identify what data is used by which function.



- In case we need to revise an external data structure, we should also revise all the functions that access that particular data. This results in increased number of errors in the program.
- Another drawback of POA is that it does not model real world problems very well. This is because in POA functions are action oriented & do not really corresponding to the elements of the problem.

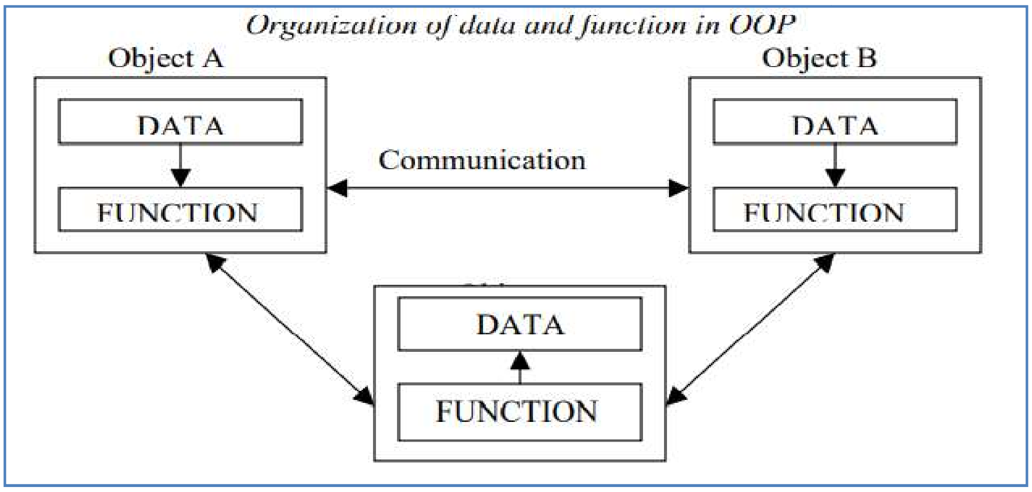
Some Characteristics exhibited by procedure-oriented programming are:

- Emphasis is on doing things (algorithms).
- Large programs are divided into smaller programs known as functions.
- Most of the functions share global data.
- Data move openly around the system from function to function.
- Functions transform data from one form to another.
- Employs top-down approach in program design.

Object Oriented Approach:

- OOP treats data as a critical element in the program development and does not allow it to flow freely around the system. It ties data more closely to the functions that operate on it and protects it from accidental modifications by the external functions.

- OOP allows us to decompose a problem into a number of entities called objects and then build data and functions around these entities.
- The data of an object can be accessed only by the functions associated with that object. However functions of one object can access the functions of other object.



Object-Oriented Programming:

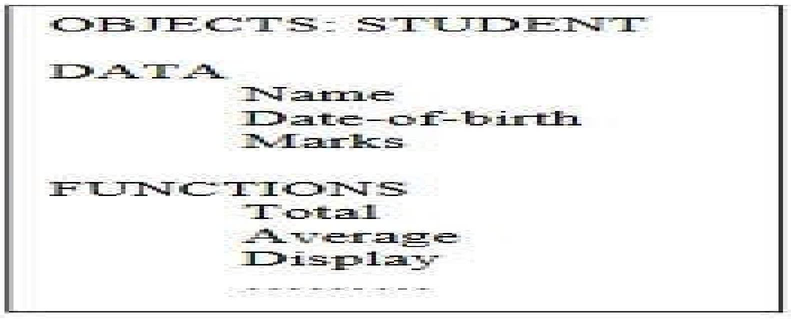
- **Object** means a real word entity such as pen, chair, table etc. **Object-Oriented Programming** is a methodology or paradigm to design a program using classes and objects. It simplifies the software development and maintenance by providing some concepts:
- Object-oriented programming is an approach to designing modular, reusable software systems.
- The goals of object-oriented programming are:
    - Increased understanding.
    - Ease of maintenance
    - Ease of evolution.
    - Object orientation eases maintenance by the use of encapsulation and information hiding.
- Some of the 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.
    - Functions that operate on the data of an object are ties together in the data structure.
    - Data is hidden and cannot be accessed by external function.
    - Objects may communicate with each other through function.
    - New data and functions can be easily added whenever necessary.
    - Follows bottom-up approach in program design.

Basic concepts of OOP:

- Objects
- Classes
- Data abstraction
- Data encapsulation
- Inheritance
- Polymorphism
- Dynamic binding
- Message Communication

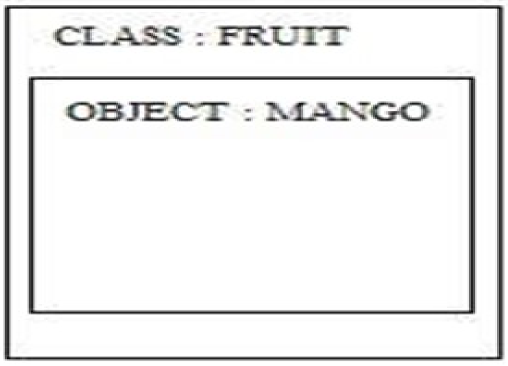
Objects:

- Objects are the basic run time entities in an object oriented system. They represent a person, a place or a bank account. Program objects should be chosen such that they match closely with the real world objects.



Classes:

- The entire set of data and code of an object can be made as user defined data type with the help of a class.
- Class is the mechanism that is used to create the objects. Objects are variables of type class. Thus a class is a collection of objects of similar type.
  Ex:-1) Mango, orange, banana are the objects of class fruit.
  2) 5Star,kitkat, dairymilk are the objects of class chocolate.
- Classes are user defined data types and we can apply all the operations which are performed on built in types on classes also.



Data abstraction:

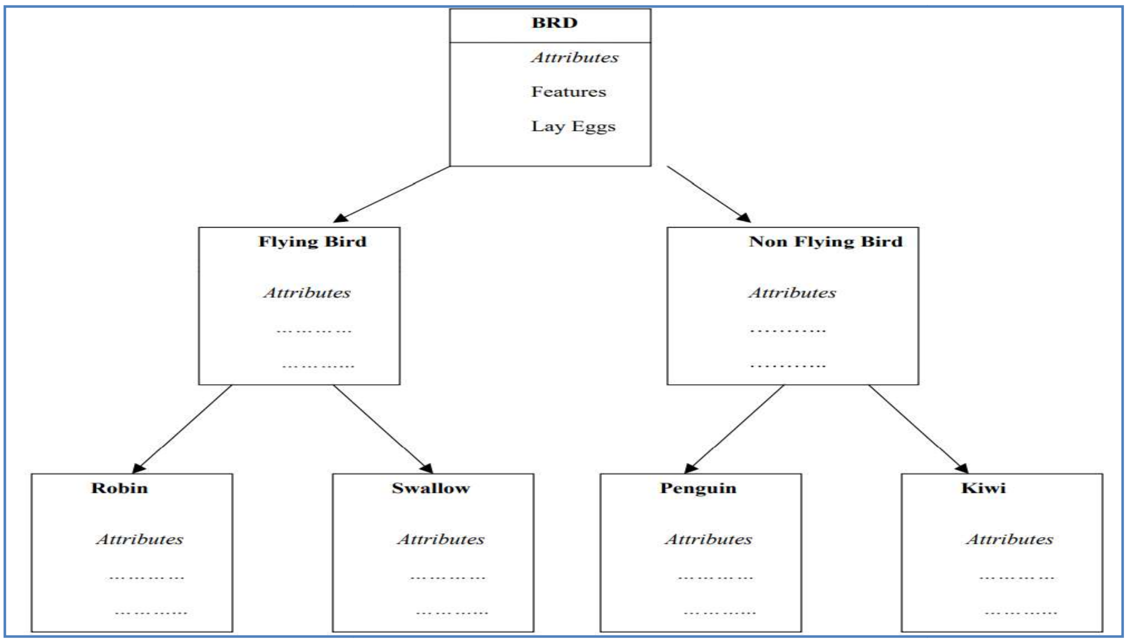
- Abstraction means ignoring those aspects of a subject that are not relevant to the current purpose in order to concentrate more fully on those that are relevant.
- **Hiding internal details and showing functionality** is known as abstraction. For example: phone call, we don't know the internal processing.
- Thus the abstraction refers to the act of representing essential features without including the background details or explanations. In java, we use abstract class and interface to achieve abstraction.

Data encapsulation:

- **Binding (or wrapping) code and data together into a single unit is known as encapsulation**. For example: capsule, it is wrapped with different medicines.
- A java class is the example of encapsulation. Javabean is the fully encapsulated class because all the data members are private here.

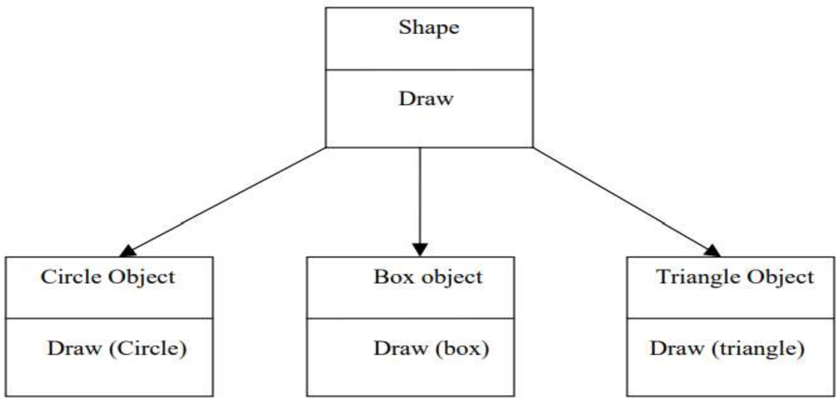
Inheritance:

- Inheritance is the process by which objects of one class can acquire the properties of objects of another class.
- 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 features of both the classes.
- The class from which a new class is derived is called the base class or a parent class. The new class which is derived is called the derived class or a child class.



Polymorphism:

- It is the quality that allows one name to be used for two or more related but technically different purposes.
- When **one task is performed by different ways** i.e. known as polymorphism. For example: to convince the customer differently, to draw something e.g. shape or rectangle etc. In java, we use method overloading and method overriding to achieve polymorphism.

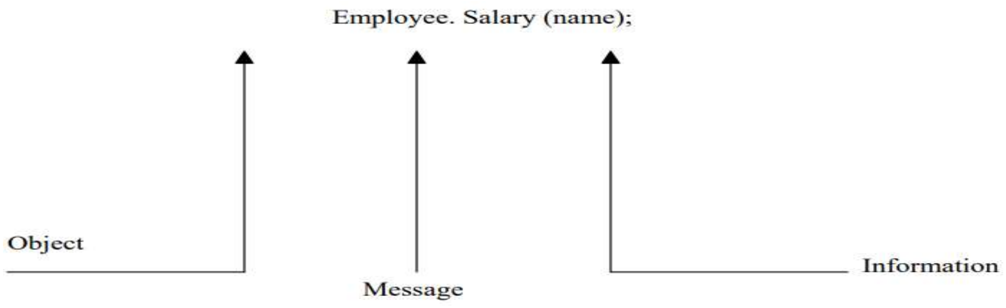


Dynamic Binding

- Binding refers to the linking of a procedure call to the code to be executed in response to the call.
- Dynamic binding means that the code associated with a given procedure call is not known until the time of the call at run time.
- It is associated with polymorphism and inheritance.

Message Communication:

- An object-oriented program consists of a set of objects that communicate with each other.
- The process of programming in an object-oriented language, involves the following basic steps:
  1. Creating classes that define object and their behaviour,
  2. Creating objects from class definitions, and
  3. Establishing communication among objects.
- Objects communicate with one another by sending and receiving information much the same way as people pass messages to one another.
- A Message for an object is a request for execution of a procedure, and therefore will invoke a function (procedure) in the receiving object that generates the desired results.
- Message passing involves specifying the name of object, the name of the function (message) and the information to be sent. Example:



Benefits of OOP:

The main advantages of OOP are,

- Through inheritance, we can eliminate redundant of code and extend the use of existing classes.
- The principle of data hiding helps the programmer to build secure programs that cannot be invaded by code in other parts of the program.
- It is possible to have multiple instances of an object to co-exist without any interference.
- It is easy to partition the work in a project which is based on objects.
- OOA approaches us to capture more details of a model in implementable form.
- System can be easily upgraded from small to large systems.
- There will be a simpler interface description with external system due to message passing technique employed in OOP.
- Software complexity can be easily managed.

Applications of OOP:

- In the area of user interface design such as Windows
- Real time systems.
- Simulation & Modeling.
- Object oriented data bases.
- Hypertext, Hypermedia and expert text.
- Neural network and parallel programming.
- Decision support and office automation system.
- CAD CAM systems.

Variables:

- ✓ A variable is an identifier that denotes the storage location.
- ✓ Variable is fundamental unit of storage in java.
- ✓ They are used in combination with identifier, data types, operators.
- ✓ Variables must be declared before its use.
- ✓ It refers the “value” that will change during the execution of program.

- **Rules for variable names**
  - It must start with a letter or underscore (-)
  - They must not begin with digit.
  - Space or any special characters are not allowed.
  - Uppercase & lowercase letters are distinct.
  - It can be of any length

    Example: height, total_height, classStrength, average

Scope of Variable:

The area of a program where the variable is accessible is called its scope. Java variables are classified into 3 types

- **Instance Variable**
  - Instance and class variables are declared inside a class.
  - Instance variable are created when the object are instantiated & therefore these are associated with the object.

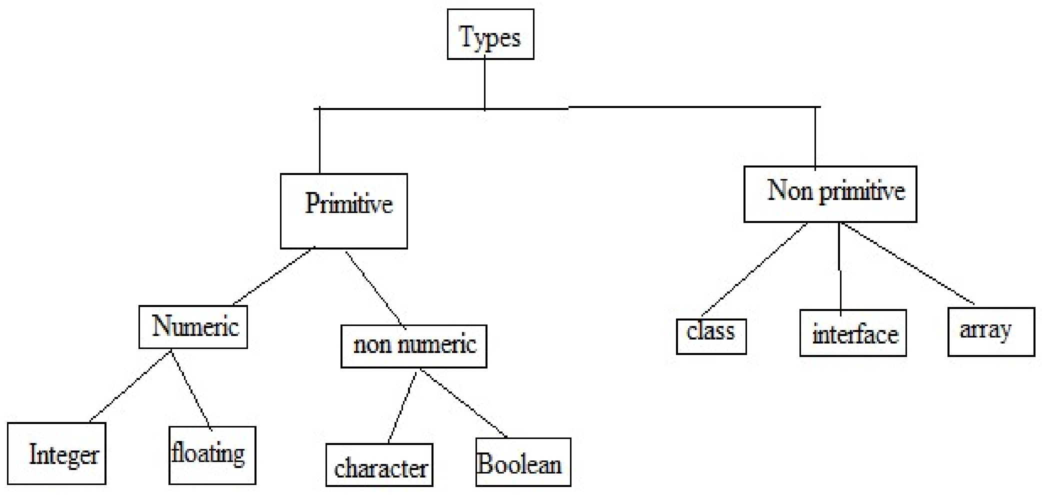
- They take different values for each object

- **Class Variable**
  - Class variables are declared inside a class
  - Class variables are global to a class &
  - Class variables belong to the entire set of objects that classcreates.
  - Only one memory is created for each class variable

- **Local variable**
  - Variables declared and used inside methods are called local variables.
  - They are not available for use outside the method definition.
  - They are visible to the program only within the block in which they are defined.
  - Local variables can also be declared inside program blocks that are defined between an opening brace "{" and close "}".
  - The program blocks can be nested. One block can be defined within another block.

Data Types:

Data types specify the size & type of values that can be stored. Various data types used in java are byte, short, int, long, char, float, double and Boolean.



✓ Primitive data types

Primitive types are built in types. They are as follows

i) **Integer:** This is the most commonly used data type for defining the numerical data that can hold whole number. The size of this data type is 4 byte having a range -2,147,483,648 to 2,147,483,647. Integer types are

- **Byte:** This is smallest integer type of data type. Its width is of 1 byte with the range -128 to 127. The variable can be declared as byte type as **byte i , j;**
- **Short:** This data type is also used for defining the signed numerical variables with a width of 2bytes and having a range from -32768 to 32,767. The variable can be declared as short type as **short i;**
- **Int**: This data type is used for defining the numerical variables with a width of 4 bytes.
- **Long:** Sometimes when int is not sufficient for declaring some data then long is used. Its width is of 8 byte.

  Ex: **long int x, y;**

ii) Floating point type: It is used to represent the real number .Two kinds of Floating point types are

- **float :** It is used to represent single precision real numbers.
- **double:** It is used to represent double precision real numbers. Its width is 8 bytes having the range 1.7e-308 to 1.7e+308.

iii) Char: This data type is used to represent the character type of data. The width of this data type is 2 byte but it can hold only a single character.

iv) Boolean: Boolean is a simple data type used to test particular condition during execution of the program .There are two values .Boolean type takes either true or false.

✓ Nonprimitive data types

Non primitive types are derived types or reference types. They are as follows

- **Array:** array is a defined data type to store homogeneous data.
- **Class:** user defined datatype.
- **Interface:** is like a class used to implement concept of multiple inheritance.