Introduction to Object Oriented Programming

Language Paradigms:

– Imperatives – Procedural Programming : C, Pascal, COBOL, FORTRAN etc.
– Applicative – Functional Programming – LISP, ML.
– Rule-based – Logic Programming :– PROLOG
– Object-Oriented Programming: – C++, JAVA, SMALLTALK

Our focus here is on procedural and Object oriented Programming approach.

Procedural programming Language

In procedural programming programs are organized in the form of subroutines. The subroutines do not let code duplication. This technique is only suitable for medium sized software applications. Conventional programming, using HLL e.g. COBOL, FORTRAN, C etc is commonly known as procedural programming. A program in Procedural Language is a list of instructions each statement in the language tells the computer to do something involving – reading, calculating, writing output. A number of functions are written to accomplish such tasks. Program become larger, it is broken into smaller units – functions. The primary focus of procedural oriented programming is on functions rather than data.

Procedure oriented programming basically consists of writing a list of instructions for computer to follow, and organize these instructions into groups known as functions.

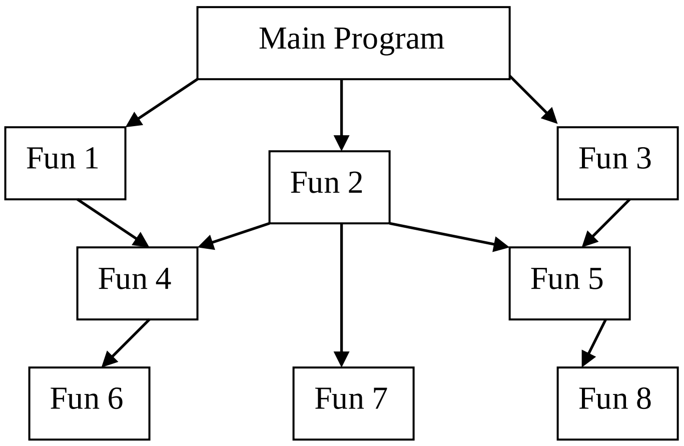
In procedural approach,

- A program is a list of instruction.
- When program in PL become larger, they are divided into functions(subroutines, sub-programs, procedures).
- Functions are grouped into modules.
-

Dividing the program into functions and modules is a task in structured programming. In structured programming there are certain structures as

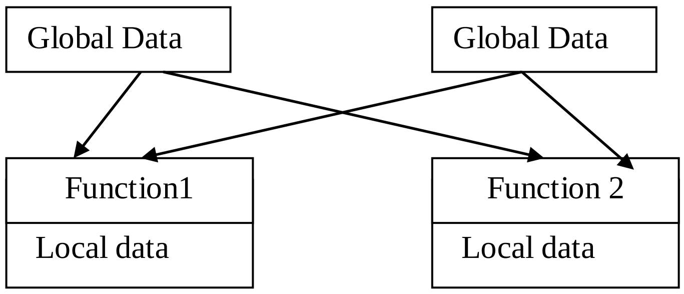
Sequences control
Selection Structures
Iteration
Functions
Modules

A pictorial views



– In a multi function program, two types of date are used: local and global.

- Data items are placed as global so that they can be accessed by all the functions freely.
- Each function may have their own data also called as local data.



Characteristics of Procedural approach:

- Emphasis is on doing things, (Algorithms)
- Large programs are divided into smaller program – function
- Most of functions share global data.
- Data more openly around system from function to function.
- Function transform data from one form to another
- Employs top-down approach for program design

Limitation of Procedural language

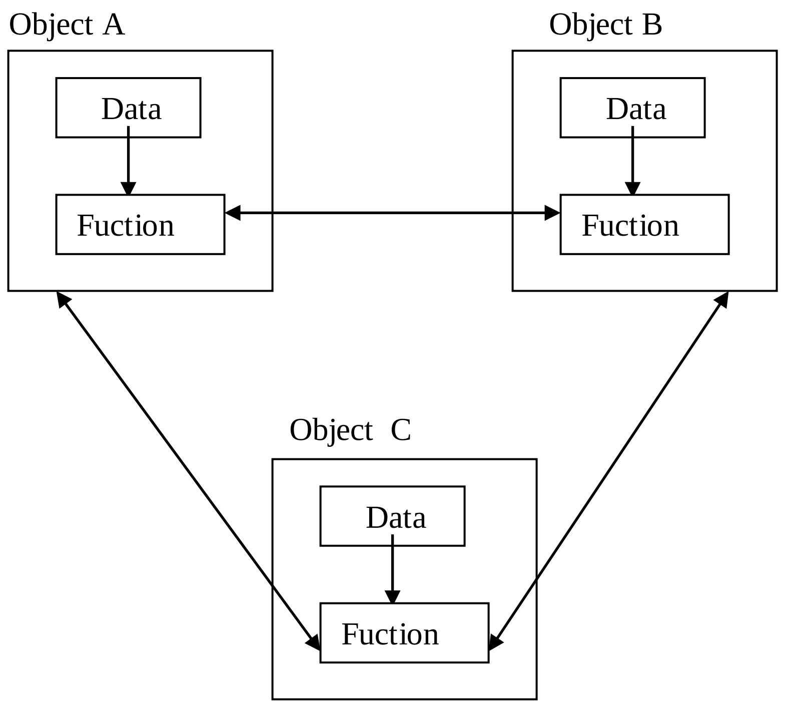
- In large program, it is difficult to identify which data is used for which function.
- Global variable overcome the local variable.
- To revise an external data structure, all functions that access the data should also be revised.
- Maintaining and enhancing program code is still difficult because of global data.
- Focus on functions rather than data.
- It does not model real world problem very well. Since functions are action oriented and do not really correspond to the elements of problem.

The Object Oriented Approach:

Object-Oriented programming is a programming methodology that associates data structures with a set of operators which act upon it. In OOP, an instance of such an entity is known as object. In other words, OOP is a method of implementation in which programs are organized as co-operative collections of objects, each of which represents an instance of some class and whose classes are all members of a hierarchy of classes united through the property called inheritance.

In Object Oriented programming

- Emphasis is on data rather than procedures.
- Programs are divided into objects.
- Data structures are designed such that they characterize the objects .
- Functions & data are tied together in the data structures so that data abstraction is introduced in addition to procedural abstraction.
- Data is hidden & can’t be accessed by external functions.
- Object can communicate with each other through function.
- New data & functions can be easily added.
- Follows Bottom up approach.



Features of Object Oriented Language:

1. **Objects**: Objects are the entities in an object oriented system through which we perceive the world around us. We naturally see our environment as being composed of things which have recognizable identities & behavior. The entities are then represented as objects in the program. They may represent a person, a place, a bank account, or any item that the program must handle. For example Automobiles are objects as they have size, weight, color etc as attributes (ie data) and starting, pressing the brake, turning the wheel, pressing accelerator pedal etc as operation (that is functions).

Object: Student
Data Name Date of birth Marks --------- ---------

Functions
Total( )
Average( )
Display( )
------------ 	

Object: Account
Data Account number Account Type Name balance ---------

Functions
deposit( )
withdraw( )
enquire( )
------------ 	

.

Example of objects:

Physical Objects:

- – Automobiles in traffic flow. simulation
- – Countries in Economic model
- – Air craft in traffic – control system .

Computer user environment objects.

-Window, menus, icons etc

Data storage constructs.

-Stacks, Trees etc

Human entities:

-employees, student, teacher etc.

Geometric objects:

-point, line, Triangle etc.

Objects mainly serve the following purposes:

- Understanding the real world and a practical base for designers
- Decomposition of a problem into objects depends on the nature of problem.

2. **Classes** : A class is a collection of objects of similar type. For example manager, peon, secretary, clerk are member of the class employee and class vehicle includes objects car, bus, etc.

It defines a data type, much like a **struct** in C programming language and built in data type(int char, float etc). It specifies what data and functions will be included in objects of that class. Defining class doesn't create an object but class is the description of object's attributes and behaviors. Person Class : Attributes: Name, Age, Sex etc.

Behaviors: Speak(), Listen(), Walk()

Vehicle Class: Attributes: Name, model, color, height etc

Behaviors: Start(), Stop(), Accelerate() etc.

When class is defined, objects are created as

```
<classname> <objectname> ;
```

If employee has been defined as a class, then the statement

employee manager;

Will create an object manager belonging to the class employee.

Each class describes a possibly infinite set of individual objects, each object is said to be an instance of its class and each instance of the class has its own value for each attribute but shares the attribute name and operations with other instances of the class. The following points gives the idea of class:

- A class is a template that unites data and operations.
- A class is an abstraction of the real world entities with similar properties.
- Ideally, the class is an implementation of abstract data type.

3. Encapsulation and Data Abstraction:

The wrapping up of data and function into a single unit is called encapsulation. Encapsulation is most striking feature of a class. The data is not accessible from outside of class. Only member function can access data on that class. The insulation of data from direct access by the program is called data hiding. That is data can be hidden making them private so that it is safe from accidental alteration.

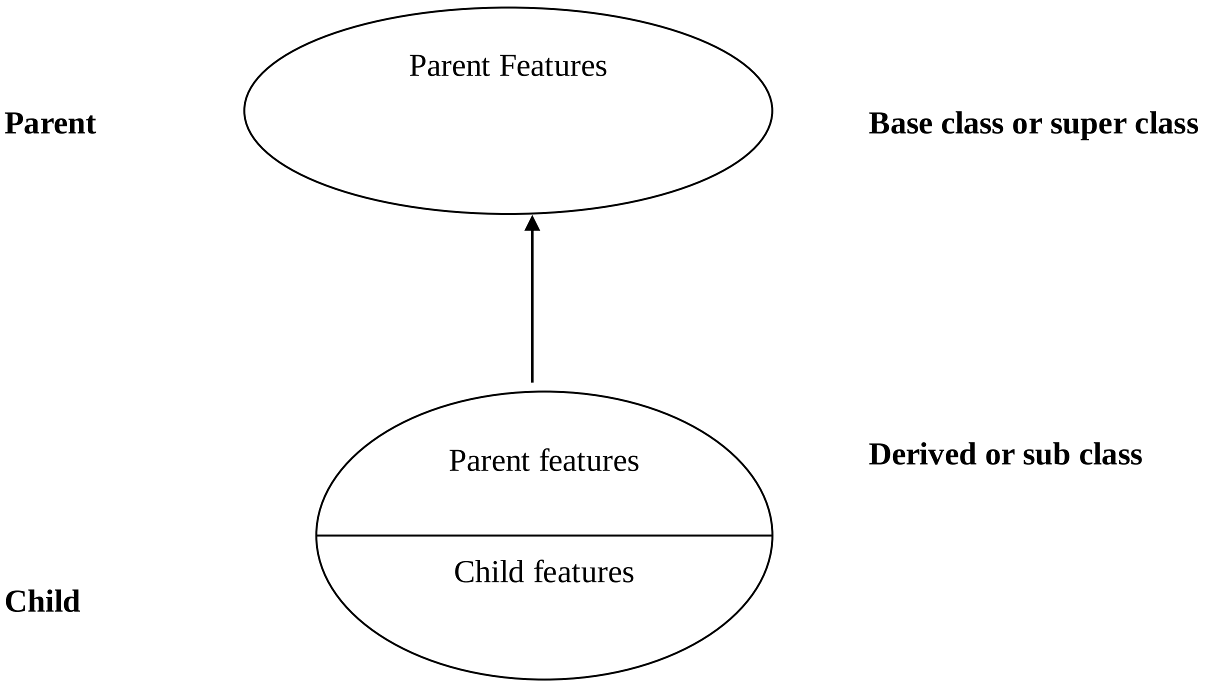
Abstraction is representing essential features of an object without including the background details or explanation. It focuses the outside view of an object, separating its essential behavior from its implementation.

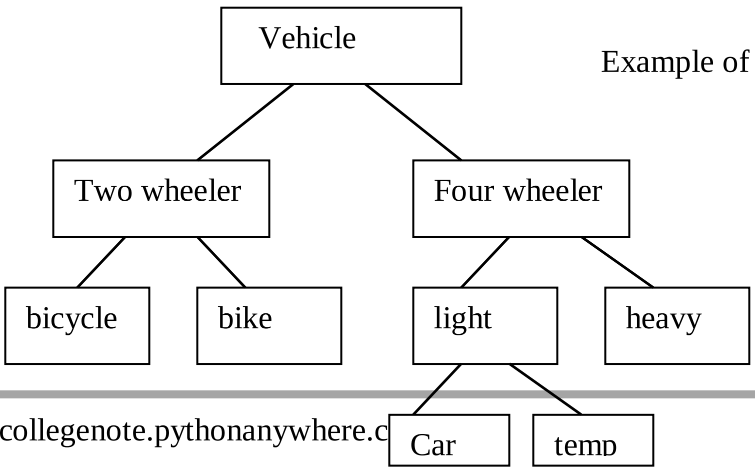
The class is a construct in C++ for creating user-defined data types called Abstract Data Types (ADT)

4. Inheritance:

Inheritance is the process by which objects of one class acquire the characteristics of object of another class. In OOP, the concept of inheritance provides the idea of **reusability**. We can use additional features to an existing class without modifying it. This is possible by deriving a new class (derived class) from the existing one (base class).This process of deriving a new class from the existing base class is called inheritance .

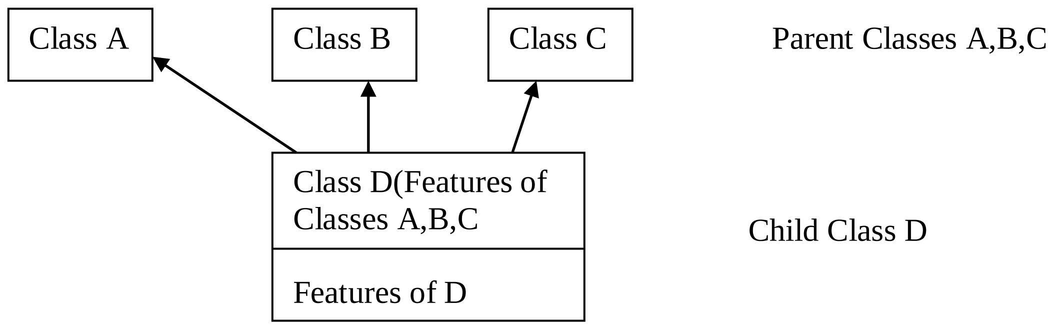
It supports the concept of hierarchical classification. It allows the extension and reuse of existing code without having to rewrite the code.





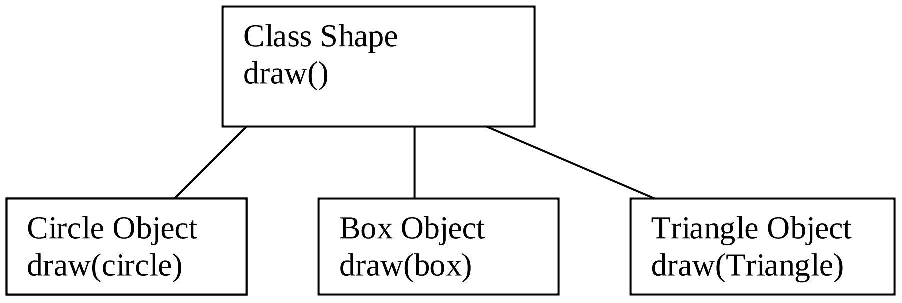
Example of Inheritance class hierarchy

Multiple Inheritance: If derived class inherits the features of more than one base class it is called multiple inheritance.



Multiple Inheritance

5. Polymorphism:
Polymorphism means "having many forms". The polymorphism allows different objects to respond to the same message in different ways, the response specific to the type of object. Polymorphism is important when object oriented programs dynamically creating and destroying the objects in runtime. Example of polymorphism in OOP is operator overloading, function overloading.
For example operator symbol '+' is used for arithmetic operation between two numbers, however by overloading (means given additional job) it can be used over Complex Object like currency that has Rs and Paisa as its attributes, complex number that has real part and imaginary part as attributes. By overloading same operator '+' can be used for different purpose like concatenation of strings.



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

6. **Message Passing**: An Object-Oriented program consists of set of objects that communicate with each other. Object communicates with each other by sending and receiving message (information). A message for an object is a request for execution of a procedure and therefore will invoke a function or procedure in receiving object that generate the desired result. Message passing involves specifying the name of the object name of the function (message) and the information to be sent .

e.g
$$\begin{tikzcd}
\text{employee} \arrow[d] & . & \text{salary} \arrow[d] & (\text{name}) ; \arrow[d] \\
\text{object} & & \text{message} & \text{information}
\end{tikzcd}$$

Advantages of OOPs

Object oriented programming contributes greater programmer productivity, better quality of software and lesser maintenance cost. The main advantages are:

- Making the use of inheritance, redundant code is eliminated and the existing class is extended.
- Through data hiding, programmer can build secure programs that cannot be invaded by code in other pats of the program.
- It is possible to have multiple instances of an object to co-exist without any interference.
- System can be easily upgraded from small to large systems.
- Software complexity can be easily managed.
- Message passing technique for communication between objects makes the interface description with external system much simpler.
- Aids trapping in an existing pattern of human thought into programming.
- Code reusability is much easier than conventional programming languages.

Disadvantages of OOPs

- Compiler and runtime overhead. Object oriented program required greater processing overhead – demands more resources.
- An object's natural environment is in RAM as a dynamic entity but traditional data storage in files or databases
- Re-orientation of software developer to object-oriented thinking.
- Requires the mastery in software engineering and programming methodology.
- Benefits only in long run while managing large software projects.
- The message passing between many objects in a complex application can be difficult to trace & debug.

Object oriented Languages:

The language should support several of OOPs concepts to claim that they are Object-Oriented. Depending upon the features they support, they can be classified as

1. Object–Based Languages
2. Object-Oriented Languages

Object Based Language: Supports encapsulation & object identity without supporting the important features of OOP languages such as Inheritance and Dynamic Bindings. Major features are

– Data Encapsulation
– Data hiding & access mechanisms
– Automatic initialization & clear-up of objects
– Operator overloading
– Don't support inheritance & Dynamic binding . e.g Ada.

Object-Based Language : Encapsulation + Object Identity

Object – Oriented language : Incorporates all features of object-based language plus inheritance and dynamic bindings

Object-Oriented L : Object based feature + inheritance + dynamic bindings
e.g SMALLTALK , C++ , JAVA,EIFFEL etc.

Application of OOP

Applications of OOP are beginning to gain importance in many areas.

– The most popular application of OOP, up to now. Has been area of user interface design such as Windows .
– Real Business systems are often much more complex attributes & behaviors .
– OOP can simplify such complex problem. The areas of application of OOP include
– Real time systems
– Simulation & modeling
– Object-Oriented databases
– Hypertext, hypermedia
– AI & expert system
– Neural Networks & parallel programming
– Decision support & Office automation system
– CAM/CAD systems .
– Computer Based training and Educational Systems

Object-Oriented Programming Languages :

SmallTalk :

– Developed by Alen Kay at Xerox Palo Alto Research center (PARC) in 1970's
– 100% OO Language
– The syntax is very unusual and this leads to learning difficulties for programmers who are used to conventional language syntax .
– Language of this type allocate memory space for object on the heap and dynamic garbage collector required .

Eiffel :

- Eiffel was designed by a Frenchman named Bertrand Meyer in the late 1980's .
– Syntax is very elegant & simple, fewer reserved world than Pascal .
– Compiler normally generates "C source which is than compiled using c compiler which can lead long compile time.
– All Eiffel object are created on the heap storage
– Pure object oriented
– Not very popular as a language for mainstream application development .

Java

- Designed by SUN(Stanford University Net) Microsystems, released in 1996 and is a pure O-O language.
- Syntax is taken from C++ but many differences .
- All object are represented by references and there is are pointer type.
- The compiler generates platform independent byte code which is executed at run time by interpreter.
- A very large library of classes for creating event driven GUI is included with JAVA compiler
- JAVA is a logical successor to C++ can also be called as C++--++(C-Plus-Plus-Minus-Minus-Plus-Plus i.e. remove some difficult to use features of C++ and Add some good features)
-

C++

- Developed by Bjarne Stroustrup at Bell Lab in New jersey in early 1980 s as extension of C
- Employs the basic syntax of the earlier C language which was developed in Bell Lab by Kernigan & Ritchie.
- One of most popular language used for s/w development .
- By default, c++ creates objects on the systems stack in the same way as the fundamental data type.
-

Unlike Java, SmallTalk and Eiffel, C++ is not pure O-O language. i.e it can be written in a conventional ‘C’.