Introduction of Object-Oriented Programming

Chapter 1
Session Contents

- Procedure Oriented Languages
- Definition of OOP
- Basic Concept of OOP
  - Object
  - Class
  - Data Encapsulation
  - Data Abstraction
  - Data hiding Member functions
  - Reusability
  - Inheritance
Session Contents

- Polymorphism
- Overloading
- Dynamic Binding and Message Passing
- Object Oriented Languages
- Application Of Object Oriented Languages
Conventional programming using high level languages such as COBOL, FORTRAN and C, is commonly known as procedure-oriented programming (POP).

In the procedure-oriented approach, the problem is viewed as a sequence of things to be done such as reading, calculating and printing.

A number of functions are written to accomplish these tasks.

The primary focus is on functions.
Typical Structure of Procedure-Oriented Programs

Main Program

Function 1

Function 2

Function 3

Function 4

Function 5
Drawbacks

- In multi-function program, many important data items are placed as global so that they can be accessed by all the function.
- Global Data is vulnerable to an inadvertent change by a function.
- Another serious drawback with procedural approach is that it does not model real world problems very well.
- This is because functions are action-oriented and do not correspond to the elements of the problem.
Relationship between data and functions in procedural programming

- Global data
- Function 1
  - Local Data
- Function 2
  - Local Data
- Function 3
  - Local Data

OOPS WITH C++

Sahaj Computer Solutions
Characteristics of POP

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

- Object Oriented Programming is an approach that provides a way of modularizing programs by creating partitioned memory areas for both data and functions that can be used as templates for creating copies of such modules on demand.
Organization of data and functions in OOP

Object A
- Data
- Functions

Object B
- Data
- Functions

Object C
- Function
- Data

Communication
Striking features of OOP

- 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 tied together in the data structure.
- Data is hidden and cannot be accessed by external functions
- Objects may communicate with each other through functions.
- New data and functions can be easily added whenever necessary.
- Follows bottom up approach in program design
Basic Concepts of OOP

- It is necessary to understand some of the concepts used extensively in object oriented programming. These include:
  - Object
  - Class
  - Data abstraction and encapsulation
  - Inheritance
  - Polymorphism
  - Dynamic binding
  - Message Passing
Objects

- Objects are the basic runtime entities in an object oriented system.
  - They may represent a person, a place, a bank account, a table of data or any item that the program has to handle. They may also represent user defined data such as vectors, time and lists.
- Programming problems is analyzed in terms of objects and the nature of communication between them.
- Program objects should be chosen such that they match closely with the real world objects.
Object

- When a program is executed the objects interact by sending messages to one another.
- For example, if “customer” and “account” are two objects in a program, then the customer object may send a message to the account object requesting for the bank balance.
- Objects interact without having to know details of each other’s data or code.
Classes

- We mentioned that objects contain data and code to manipulate that data.
- The entire set of data and code of an object can be made a user defined data type with the help of a class.
- Objects are variables of the type class.
- Once class has been defined we can create any number of objects belonging to that class.
- Class is collection of objects of similar types.
Data Abstraction and Encapsulation

- The wrapping up of data and functions into a single unit (called class) is known as encapsulation.
- Data encapsulation is most striking feature of a class. The data is not accessed to the outside world and only those functions which are wrapped in the class can access it.
- The insulation of data from direct access by the program is called data hiding or information hiding.
Data Abstraction and Encapsulation

- Abstraction refers to the act of representing essential features without including the background details or explanations.
- Since classes use the concept of data abstraction, they are known as Abstract Data Types (ADT).
Inheritance

- Inheritance is a process by which object a of one class acquire the properties of objects 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 inheritance is that each derived class shares common characteristics with the class from which it is derived.
- Inheritance provides the idea of reusability.
- It is the process of adding new features to the existing class without modifying it.
Polymorphism

- Polymorphism is another important OOP concept.
- Poly in Greek, means many and morph means forms. Polymorphism means the ability to take more than one form.
- For example, the + operator is used for addition of two numbers, also the same operator is used for concatenation of two strings.
- Polymorphism plays an important role in allowing objects having different internal structures to share the same external interface.
Polymorphism

- In polymorphism general class operations may be accessed in the same manner even though specific actions associated with each operations may differ. Polymorphism is used in implementing inheritance.
Dynamic Binding

- Binding refers to the linking of procedure call to the code to be executed in response to the call.
- Dynamic binding (also known as late binding) means that the code associated with a given procedure call is not known until the time of the call at runtime.
- It is associated with polymorphism and inheritance.
Message Passing

- An object-oriented program consists of objects that communicate with each other.
- The process of programming in an object-oriented language, therefore involves the following basic steps:
  - Creating classes that define objects and their behaviors
  - Creating objects from class definitions and
  - Establishing communication among objects.
Benefits of OOP

- Through inheritance we can eliminate redundant code and extend the use of existing classes.
- The principle of data hiding helps programmers to build secure code that cannot be accessed by other parts of the program.
- It is possible to map objects in the problem domain to those in the program.
- Object-oriented systems can easily upgrade from small to large systems.
- Software complexity can be easily managed.
# Object Oriented Languages

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Simula</th>
<th>Smalltalk</th>
<th>Objective C</th>
<th>C++</th>
<th>Ada</th>
<th>Object Pascal</th>
<th>Turbo Pascal</th>
<th>Eiffel</th>
<th>Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binding</td>
<td>Both</td>
<td>Late</td>
<td>Both</td>
<td>Both</td>
<td>Early</td>
<td>Late</td>
<td>Early</td>
<td>Early</td>
<td>Both</td>
</tr>
<tr>
<td>Polymorphism</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Data Hiding</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Concurrency</td>
<td>✓</td>
<td>Poor</td>
<td>Poor</td>
<td>Poor</td>
<td>Difficult</td>
<td>No</td>
<td>No</td>
<td>Promised</td>
<td>✓</td>
</tr>
<tr>
<td>Inheritance</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>No</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Multiple Inheritance</td>
<td>No</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>No</td>
<td>---</td>
<td>---</td>
<td>✓</td>
<td>No</td>
</tr>
<tr>
<td>Garbage Collection</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>No</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Persistence</td>
<td>No</td>
<td>Promised</td>
<td>No</td>
<td>No</td>
<td>Like 3GL</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Some Support</td>
</tr>
<tr>
<td>Genericity</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>✓</td>
<td>✓</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>✓</td>
</tr>
</tbody>
</table>

OOPS WITH C++  
Sahaj Computer Solutions
Applications Of OOP

- Real-time System
- Simulation and modeling
- Object-Oriented databases
- Hypertext, hypermedia and expertext
- AI and expert systems
- Neural Networks and Parallel programming
- Decision support and office automation systems
- CIM/CAM/CAD systems