Introduction to Object Oriented Programming

SE E&TC

Academic Year 2017-18 Sem – II

Prof. Dipak R Raut

International Institute of Information Technology

Pune

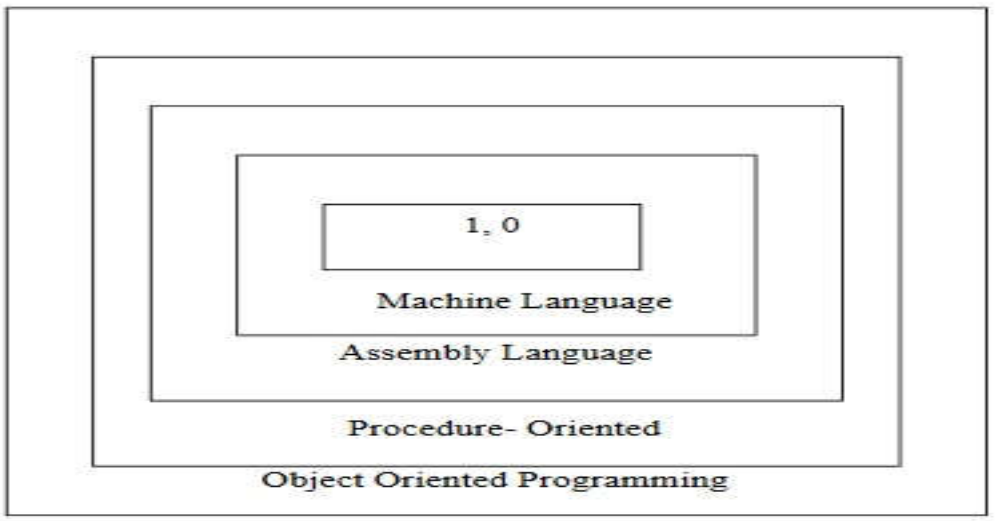
1.1 Software Crisis

- Developments in software technology continue to be dynamic.
- New tools and techniques are announced in quick succession.
- This has forced the software engineers and industry to continuously look for new approaches to software design and development.
- These rapid advances appear to have created a situation of crisis within the industry
- The following issued need to be addressed to face the crisis:

  - ✓ How to represent real-life entities of problems in system design?
  - ✓ How to design system with open interfaces?
  - ✓ How to improve the quality of software?
  - ✓ How to manage time schedules?
  - ✓ How to ensure reusability and extensibility of modules?
  - ✓ How to develop modules that are tolerant of any changes in future?
  - ✓ How to improve software productivity and decrease software cost?

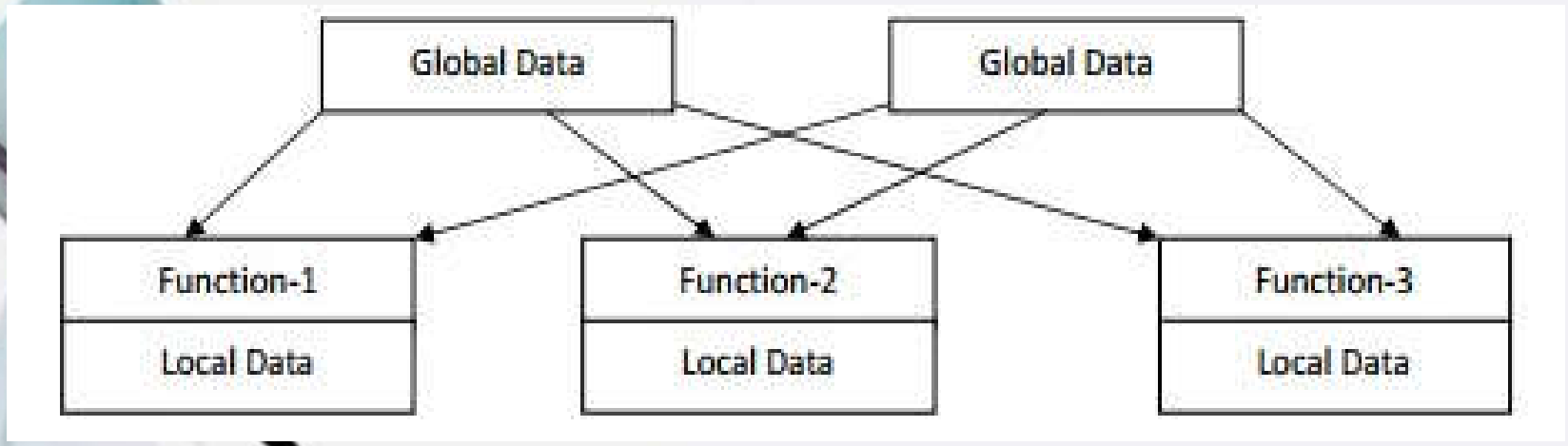
1.2 Software Evolution

- Ernest Tello, A well known writer in the field of artificial intelligence, compared the evolution of software technology to the growth of the tree.
- Alan Kay, one of the promoters of the object-oriented paradigm and the principal designer of Smalltalk, has said: “As complexity increases, architecture dominates the basic materials”.



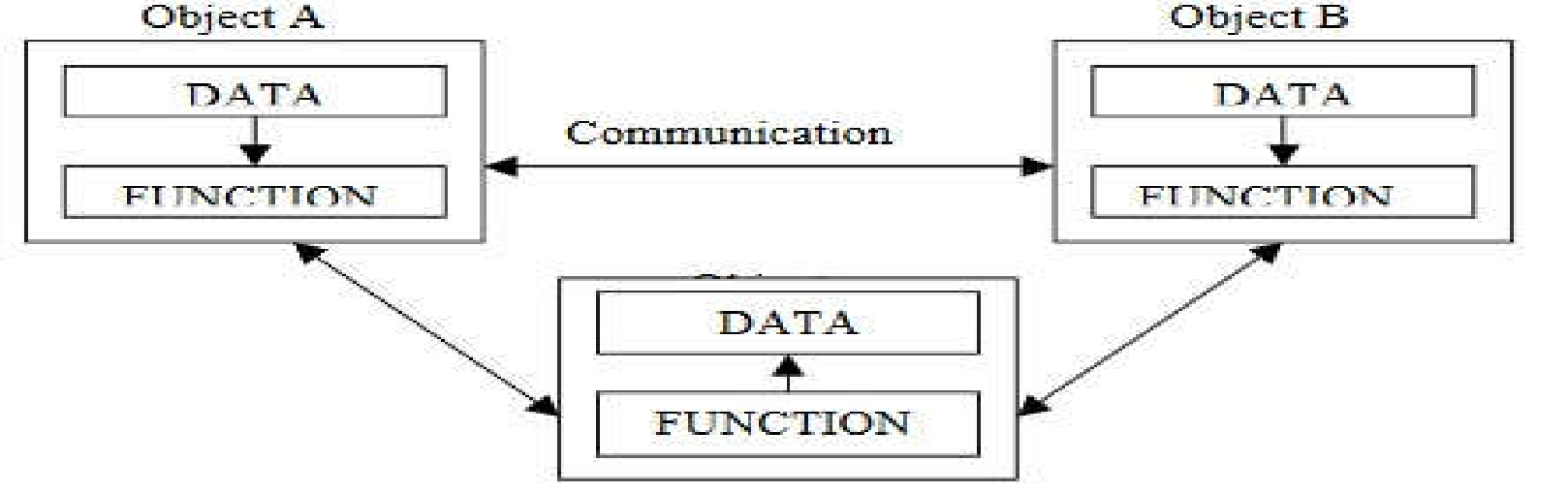
1.3 Procedure/ Structure Oriented Programming

- 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 :



1.4 Object Oriented Programming

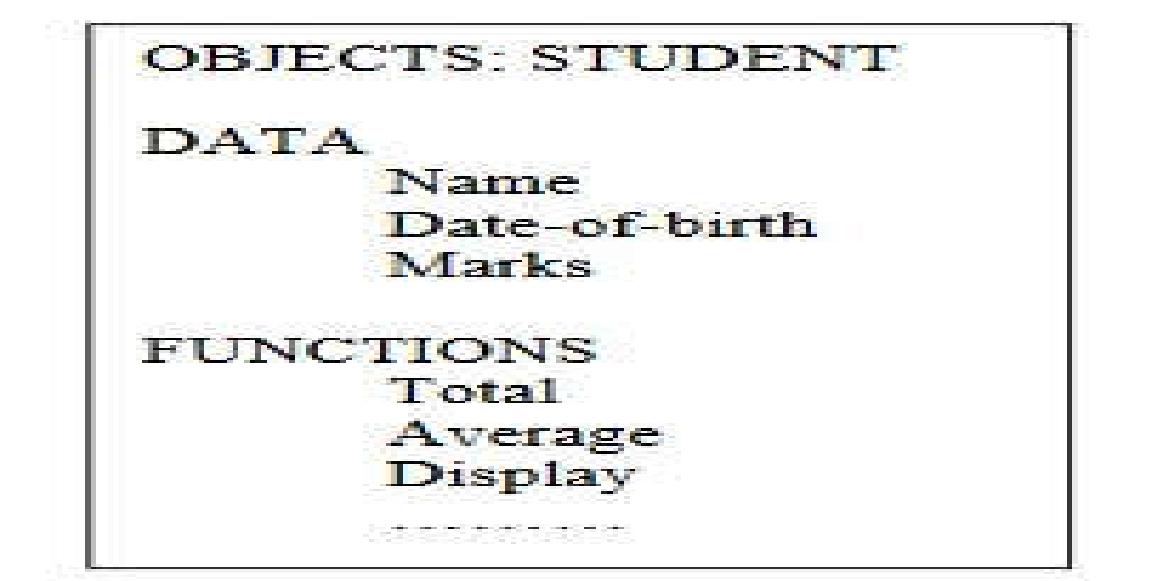
- Emphasis is on data rather than procedure.
- Programs are divided into what are known as objects.
- Data is hidden and cannot be accessed by external functions.
- Objects may communicate with each other through functions.
- New data and functions can be easily added whenever necessary.
- *Follows bottom-up* approach in program design



1.5 Basic Concepts of OOP

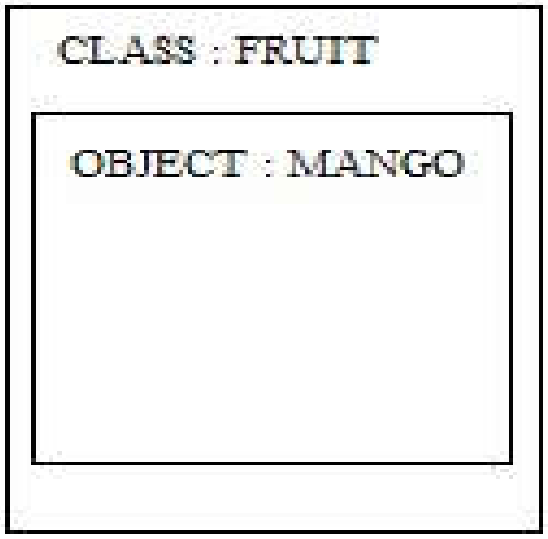
1.5.1 Objects :

- 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



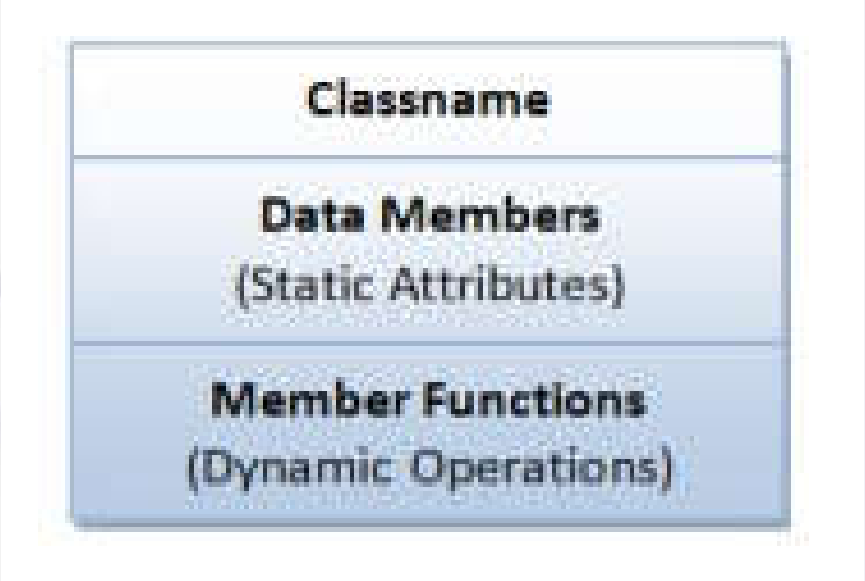
1.5.2 Class

- Object contains 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



- A Class is a 3-Compartment Box encapsulating Data and Functions

1. **Classname** (or identifier): identifies the class.
2. **Data Members** or Variables (or attributes, states, fields): contains the static attributes of the class.
3. **Member Functions** (or methods, behaviors, operations): contains the dynamic operations of the class.



1.5.3 Data Encapsulation

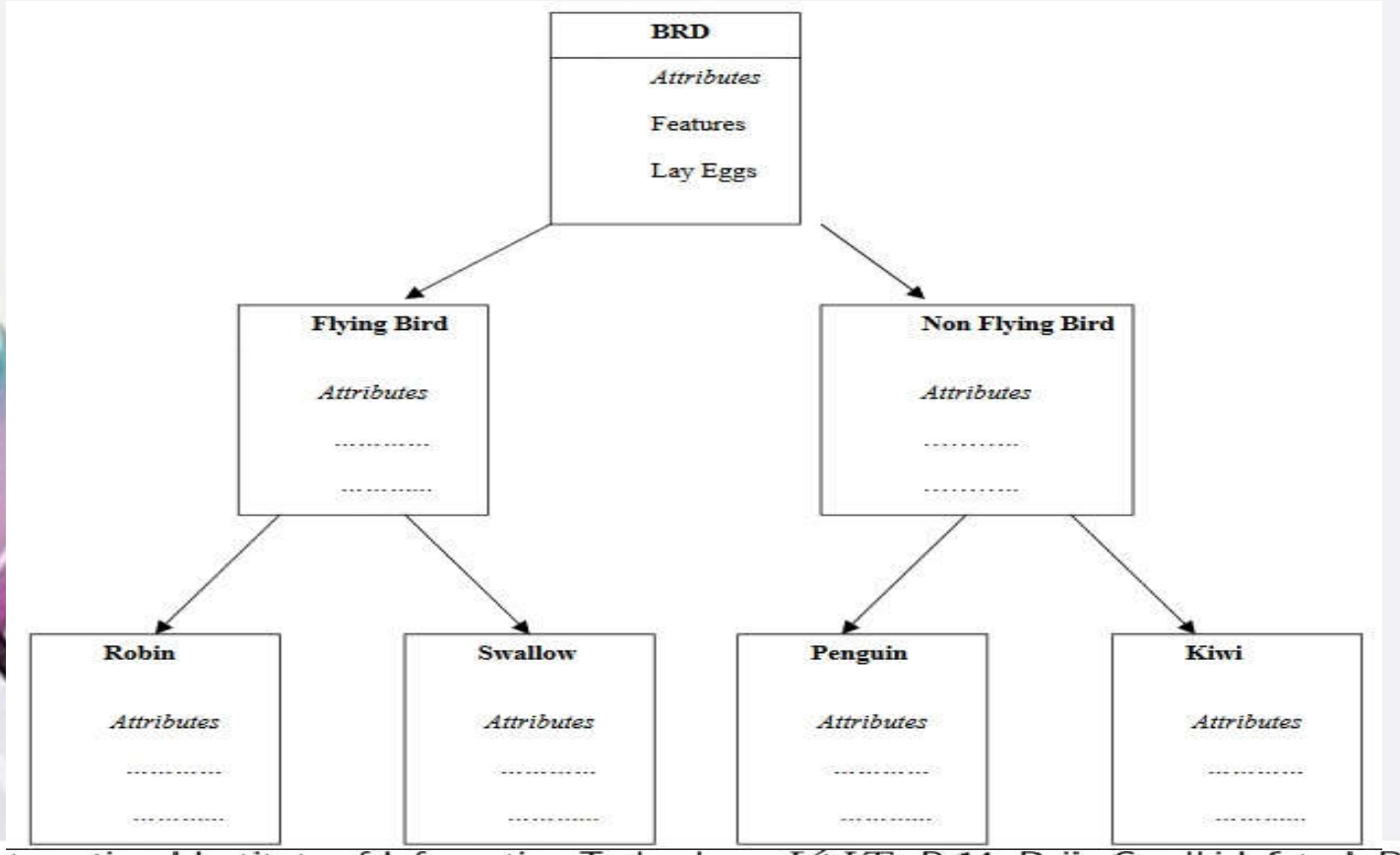
- The wrapping up of data and functions into a single unit is known as encapsulation.
- The data is not accessible to the outside world, only those function which are wrapped in the can access it.
- These functions provide the interface between the object's data and the program.
- This insulation of the data from direct access by the program is called data hiding or information hiding.

1.5.4 Data Abstraction

- 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)**

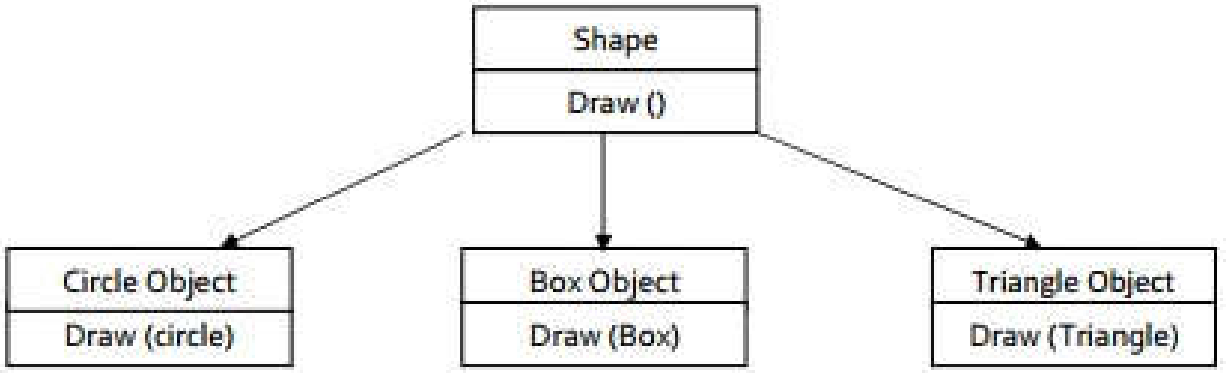
1.5.5 Inheritance

- Inheritance is the process by which objects of one class acquire the properties of objects of another class.
- In OOP, the concept of inheritance provides the idea of reusability. This means we can add additional features to an existing class without modifying it.



1.5.6 Polymorphism

- **Polymorphism,** a Greek term means to ability to take more than one form.
- An operation may exhibits different behaviors in different instances. The behavior depends upon the type of data used in the operation.
- For example consider the operation of addition for two numbers; the operation will generate a sum. If the operands are string then the operation would produce a third string by concatenation.
- The process of making an operator to exhibit different behavior in different instances is known operator overloading



1.6 Benefits of OOP

OOP offers several benefits to both the program designer & the user

- Through inheritance, we can eliminate redundant code and extend the use of existing class
- We can build programs from the standard working module the communicate with one another, rather than having to start writing code from scratch. This leads to saving of development time & higher productivity.
- 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 map objects in the problem domain to those in the program
- It is easy to partition the work in a project based on objects
- The data-centered design approach enables us to capture more details of a model in implementable form.

.

Benefits of OOP

- Object-oriented systems can be easily upgraded from small to large systems.
- Message passing techniques for communication between objects makes the interface descriptions with external systems much simpler.
- Software complexity can be easily managed .

While it is possible to incorporate all these features in an object-oriented system, their importance depends on the type of the project and the preference of the programmer. There are a number of issues that need to be tackled to reap some of the benefits stated above. For instance, object libraries must be available for reuse. The technology is still developing and current products may be superseded quickly. Strict controls and protocols need to be developed if reuse is not to be compromised.

1.7 Application of OOP

Real-business systems are often much more complex and contain many more objects with complicated attributes and methods. OOP is useful in these types of applications because it can simplify a complex problem. The promising areas for application of OOP include:

- Real-time systems
- Simulation and modeling
- Object-oriented databases
- Hypertext, hypermedia and expertext
- Al and expert systems
- Neural networks and parallel programming
- Decision support and office automation systems
- CIM/CAM/CAD systems

References

- Object Oriented Programming With C++, Fourth Edition, E BALAGURUSAMY.

Thank You

- For further information please contact

Prof. Dipak R Raut

Department of Electronics & Telecommunication Engineering

Hope Foundation's

International Institute of Information Technology, (I^2IT)
Hinjawadi, Pune 411057

Phone - +91202293341

www.isquareit.edu.in | dipakr@isquareit.edu.in