Introduction to Object Oriented Concepts

Marwadi University

Department of Diploma Engineering

UNIT 1

Object Oriented Programming with C++ – 09CE2301

Prof. Meghnesh Jayswal

Objective

- Overview and Characteristics of Procedure-Oriented Programming
- Object-Oriented Programming – Definition
- Basic Concepts of Object-Oriented Programming
- Benefits of Object-Oriented Programming
- Application of OOP

Software Difficulties

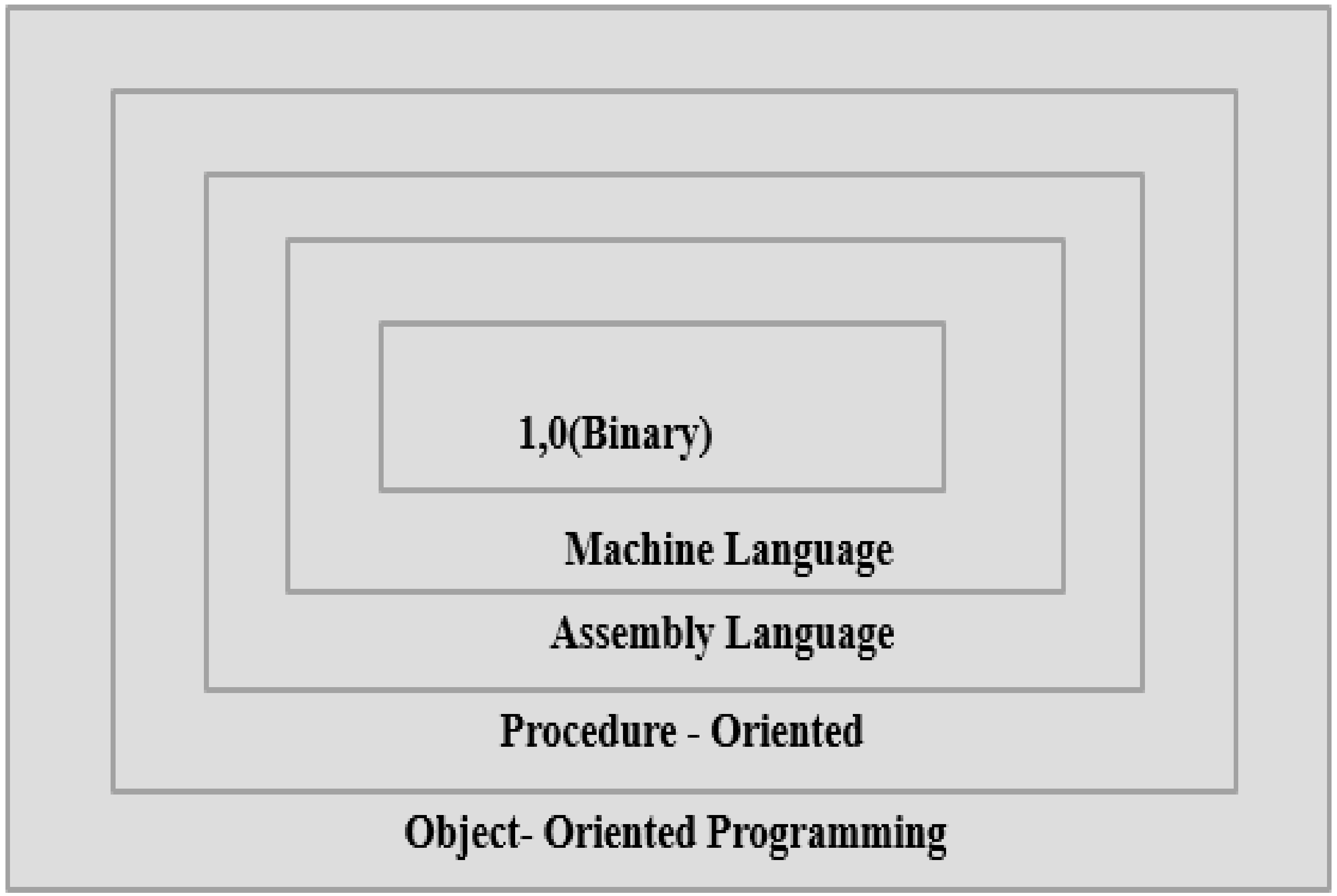
- Software technology is **dynamic** as continuous new approach to software design and development.
- Software product should be **evaluated carefully** for their quality before they are delivered and implemented.

- Some quality issues that considered as……
- 1. Correctness
- 2. Maintainability
- 3. Reusability
- 4. Openness and interoperability
- 5. Portability

6. Security
7. Integrity
8. User Friendliness

Software Evolution

- **Layers of computer software**



- S/w evolution has distinct phases or “layers” or growth. Each layer has improvement over previous one. Each layer work as functional.
- Modular Programming, top-down programming, bottom-up programming and structured programming are different techniques of programming.
- Structured Programming was powerful tool that enable programmers to write moderately complex programs fairly easily.
- OOP is new way of organizing and developing programs.

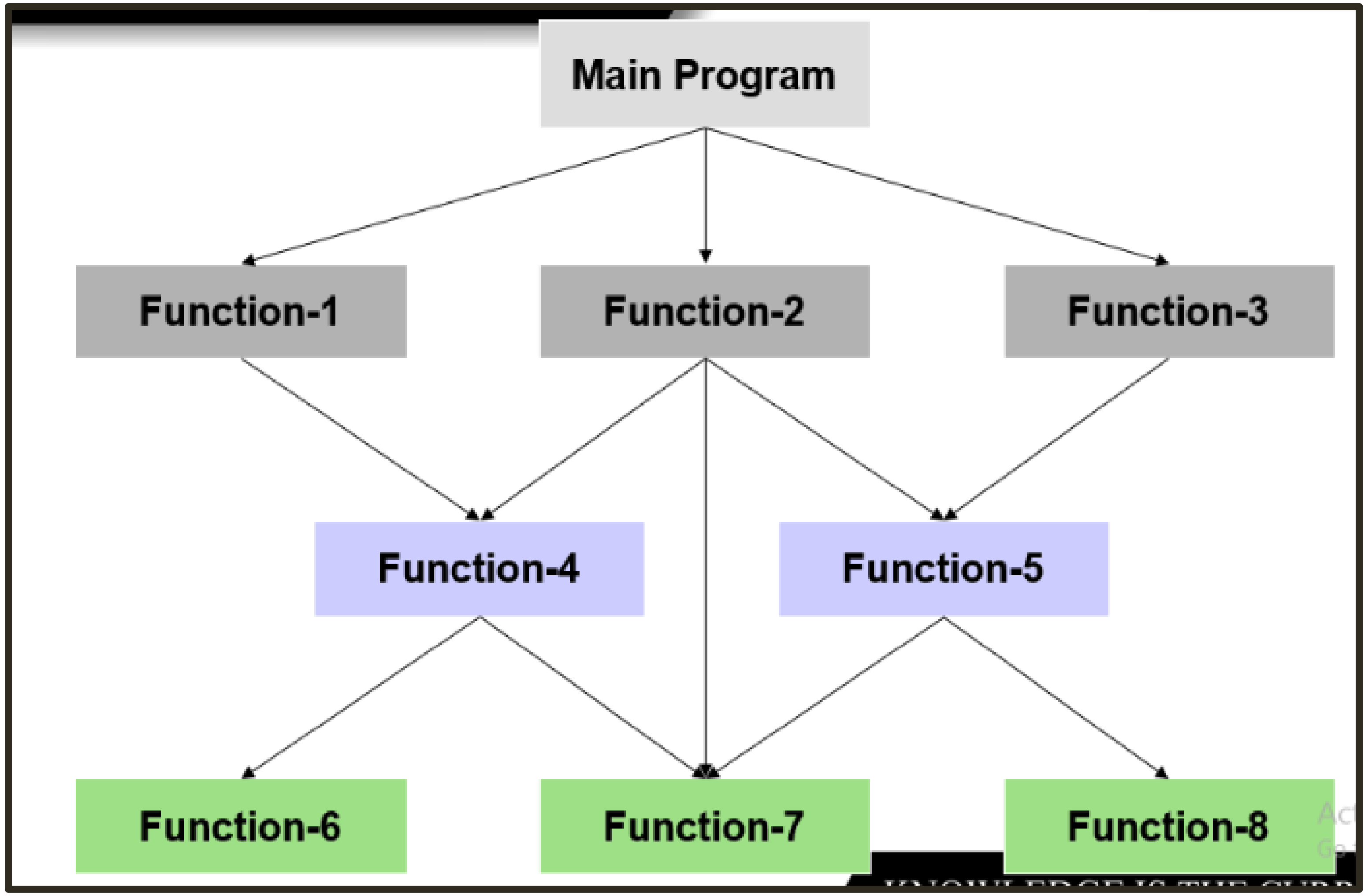
Object Oriented Programming

- **Object-Oriented Programming (OOP)** is the term used to describe a programming approach based on **objects** and **classes**. The object-oriented paradigm allows us to organize software as a collection of objects that consist of both data and behavior.

Procedure-Oriented Programming

- Traditional procedural language, such as assembly language or a high-level like COBOL, FORTRAN, C, etc.
- The problem is viewed as a sequence of things to be done.
- The primary focus is on functions.
- Procedure-oriented programming basically consists of writing a list of instructions for the computer to follow and organizing these instructions into groups known as functions.

Typical structure of procedure-oriented program

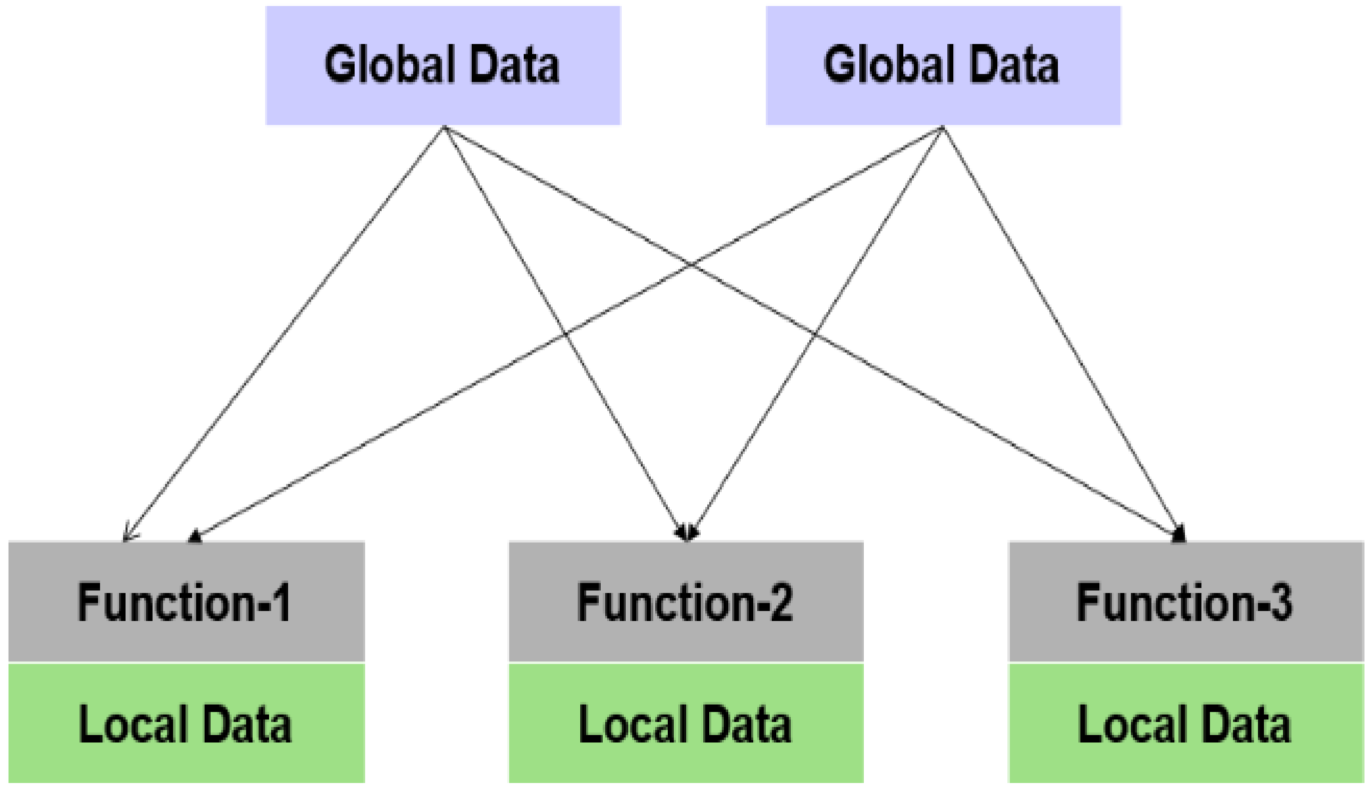


Procedure-Oriented Programming

- To revise an external data structure, we also need to revise all functions that access the data.
- This approach does not model real world problems. This is because functions are action-oriented and do not really correspond to the elements of the problem.
- In multi-function program, many important data items are placed as global so it may access by all function.

- **Drawback:- It** does not model real world problems very well. Due to function are action-oriented and do not really corresponds to elements of the problem.

Relationship of data and functions in procedural programming



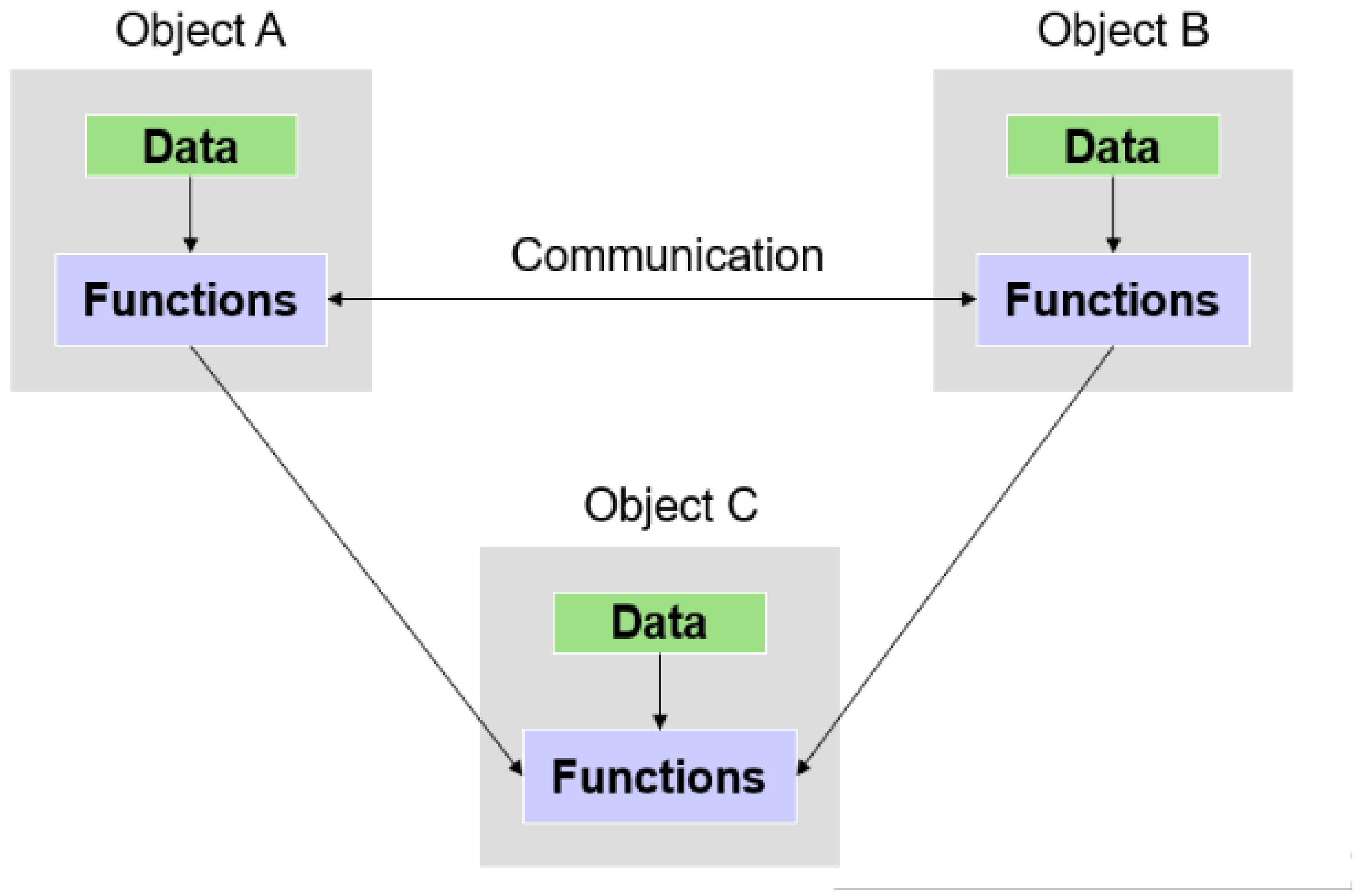
Object-Oriented Programming

- OOP treat 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 modification from outside functions.
- OOP allows decomposition of a problem into a number of entities called objects and then build data functions around these objects.

Object-Oriented Programming

- The data of an object can be accessed only by the functions associated with that object.
- Functions of one object can access the functions of another objects.
- Example: Class Fruit, that have two object mango(sweet and yellow) and apple(sweet and red color). In both have some functions are same and some not same.

Organization of data and functions in OOP



Object-Oriented Programming

- Definition:

  It is an approach that provides a way of modularizing programs by creating partitioned memory area for both data and functions that can be used as templates for creating copies of such modules on demand.

Thus the object is considered to be a partitioned area of computer memory that stores data and set of operations that can access that data.

Procedure Oriented V/s Object Oriented

POP	OOP
Emphasis is doing on things not on data, so it is procedure oriented.	Emphasis is on data, so it is object oriented.
Main focus is on function and procedures that operate on data.	Main focus is data that is being operated.
Top-Down approach in program design	Bottom-up approach in program design.
Larger program is divide in smaller parts known as function.	Large program is divided into classes and objects.
Most of function share global data.	Data is tied together with function in data structure.

Procedure Oriented V/s Object Oriented

Data moves openly from one function to another function.	Data is hidden and can’t be accessed external events.
Adding of function and data is difficult.	Adding of function and data is easy.
We can’t declare namespace directly.	We can use name space directly; eg. Using namespace std;
Inheritance, polymorphism, abstraction, access specified are not supported.	Inheritance, polymorphism, abstraction, access specified are supported.
Eg. FORTRAN, C, Pascal etc…	Eg. C++, Java, C# etc…

Basic Concepts of Object-Oriented Programming

- Objects
- Classes
- Data Abstraction and Encapsulation
- Inheritance
- Polymorphism
- Dynamic Binding
- Message Passing

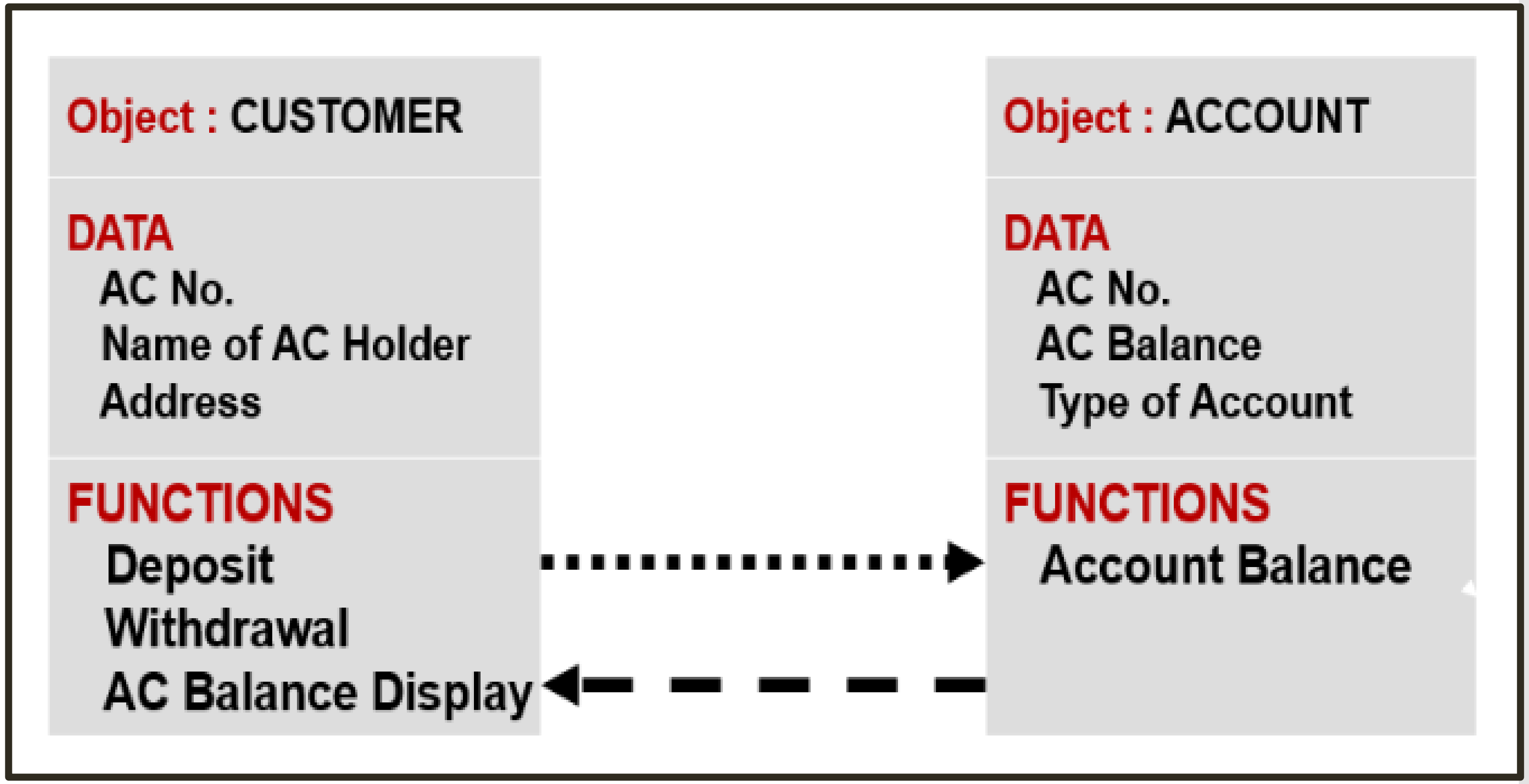
Basic Concepts of OOP

- Objects

  Objects are the basic(unit) run-time entities in an object-oriented system. They may represent a person, a place, a bank account, etc. Objects take up space in the memory and have an associated address like a structure in C.

When a program is executed, the objects interact by sending messages to one another.

Basic Concepts of OOP



Basic Concepts of OOP

- Classes

  Classes are user-defined data types.

  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 a class has been defined, we can create any number of objects belonging to that class. Each object is associated with the data of type class with which they are created.

A class is a collection of objects of similar type.

Basic Concepts of OOP

- Classes

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

fruit   mango ;

will create an object mango belonging to the class fruit.

Example: Class And Object

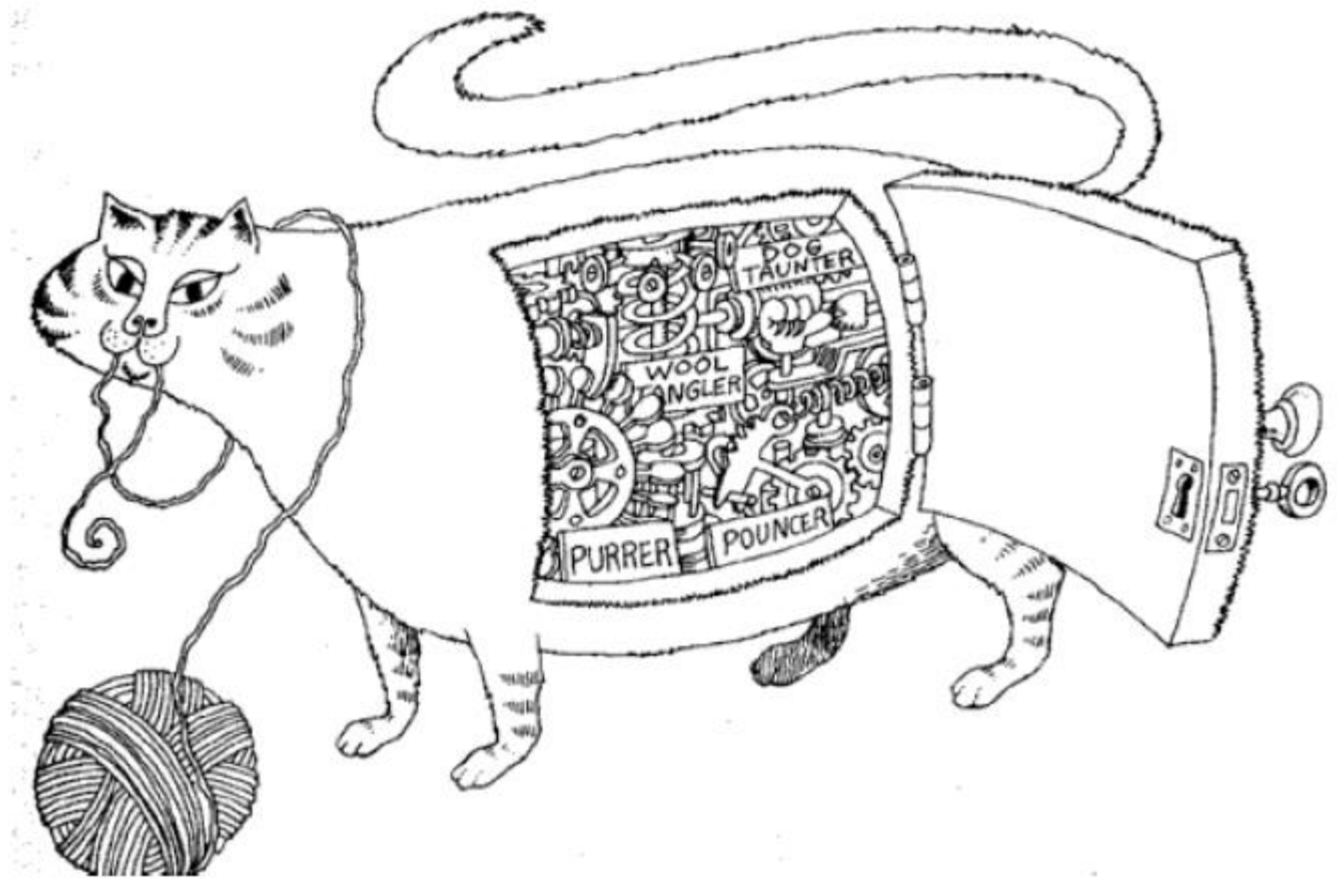
```
// Header Files
#include <iostream>
#include<conio.h>
using namespace std;
// Class Declaration
class person {
   //Access - Specifier
public:
   //Variable Declaration
   string name;
   int number;
};
```

```
//Main Function
int main() {
   // Object Creation For Class
   person obj;
   //Get Input Values For Object Varibales
   cout << "Enter the Name :";
   cin >> obj.name;
   cout << "Enter the Number :";
   cin >> obj.number;
   //Show the Output
   cout << obj.name << ": " << obj.number << endl;
   return 0;
}
```

Basic Concepts of OOP

- Data Abstraction and 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, and only those functions which are wrapped in the class 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.

Encapsulation



Encapsulation hides the details of the implementation of an object.

Basic Concepts of OOP

- Data Abstraction and Encapsulation

The attributes wrapped in the classes are called **data members** and **the functions** that operate on these data are called methods or member functions.

Since the classes use the concept of data abstraction, they are known as Abstracted Data Types (ADT).

Abstraction

Hiding details,
showing necessary features

Difference between Encapsulation and Abstraction

ABSTRACTION	ENCAPSULATION
Abstraction is the process or method of gaining the information.	While encapsulation is the process or method to contain the information.
In abstraction, problems are solved at the design or interface level.	While in encapsulation, problems are solved at the implementation level
Abstraction is the method of hiding the unwanted information.	Whereas encapsulation is a method to hide the data in a single entity or unit along with a method to protect information from outside.

Difference between Encapsulation : and Abstraction

ABSTRACTION	ENCAPSULATION
We can implement abstraction using abstract class and interfaces.	Whereas encapsulation can be implemented using by access modifier i.e private, protected and public.
In abstraction, implementation complexities are hidden using abstract classes and interfaces.	While in encapsulation, the data is hidden using methods of getters and setters.
The objects that help to perform abstraction are encapsulated	Whereas the objects that result in encapsulation need not be abstracted.

Basic Concepts of OOP

- Inheritance
  - Inheritance is the process by which objects of one class acquire the properties of objects of another class.
  - It supports the concept of hierarchical classification.
  - Each derived class shares common characteristics with the class from which it is derived.

Basic Concepts of OOP

- Inheritance
  - Inheritance provides the idea of reusability.
    - We can add additional features to an existing class without modifying it.
    - *(By deriving new class from existing one. The new class will have the combined features of both the classes.)*

Basic Concepts of OOP

- Polymorphism - *ability to take more than one form*
  - An operation may exhibit different behaviors in different instances.
  - The behavior depends upon the types of data used in the operation.
  - add(3, 5) gives 8
  - Add(“hello”, “-world”) gives “hello-world”

Basic Concepts of OOP

- One object,
  taking many forms….

Basic Concepts of OOP

- Polymorphism - *ability to take more than one form*
  - The process of making an operator to exhibit different behaviors in different instances is known as operator overloading.
  - << Insertion Operator
  - << Left-shift bit-wise operator
  - Using a single function name to perform different types of tasks is known as function overloading.
  - add(3, 5) gives 8
  - Add(“hello”, “-world”) gives “hello-world”

Basic Concepts of OOP

- Dynamic Binding
  - Binding refers to the linking of a procedure call to the code to be executed in response to the call.
  - Dynamic binding (late 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.

Basic Concepts of OOP

- Communicate using methods…

Basic Concepts of OOP

- Message Passing
  - An oop consists of a set of objects that communicate with each other.
  - Oop involves the following steps:
    - Creating classes that define objects and their behavior.
    - Creating objects from class definitions.
    - Establishing communication among objects.
  - Objects communicate with one another by sending and receiving information.

Basic Concepts of OOP

- Message Passing
  - A message for an object is a request for execution of a procedure.
  - The receiving object will invoke a function and generates results.
  - Message passing involves specifying:
    - The name of the Object.
    - The name of the Function.
    - The information to be send.

Modularity

- **Made up of small modules (components)**

Benefits of OOP

- Inheritance – eliminate redundant code and extend the use of existing classes.
- We can build programs from the standard working module, no need of starting from the scratch.
- Data hiding helps the programmer to build secure programs that can not be invented by code in other parts of the program.

Benefits of OOP

- Multiple instances of an objects can co-exists with out any interference.
- It is easy to partition the work in a project based on objects.
- Object-oriented system 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.

APPLICATION OF OOP

- Real time system
- Simulation and modeling
- Object-oriented Databases
- Hypertext, Hypermedia
- AI and Expert system
- Neural network and parallel programming
- Decision support and office automation system
- CIM/ CAM/CAD systems

Questions

1. POP Vs OOP OR C Vs C++ Or Structured programming and object oriented programing
2. Basic concept of OOP OR Features of OOP.
3. Applications and benefits of oop

Thank You