Object Oriented Programming(OOP)

(Introduction)

Introduction

Quality Issues required for software product:

- Correctness
- User friendliness
- Maintainability
- Portability
- Security
- Openness and interoperability

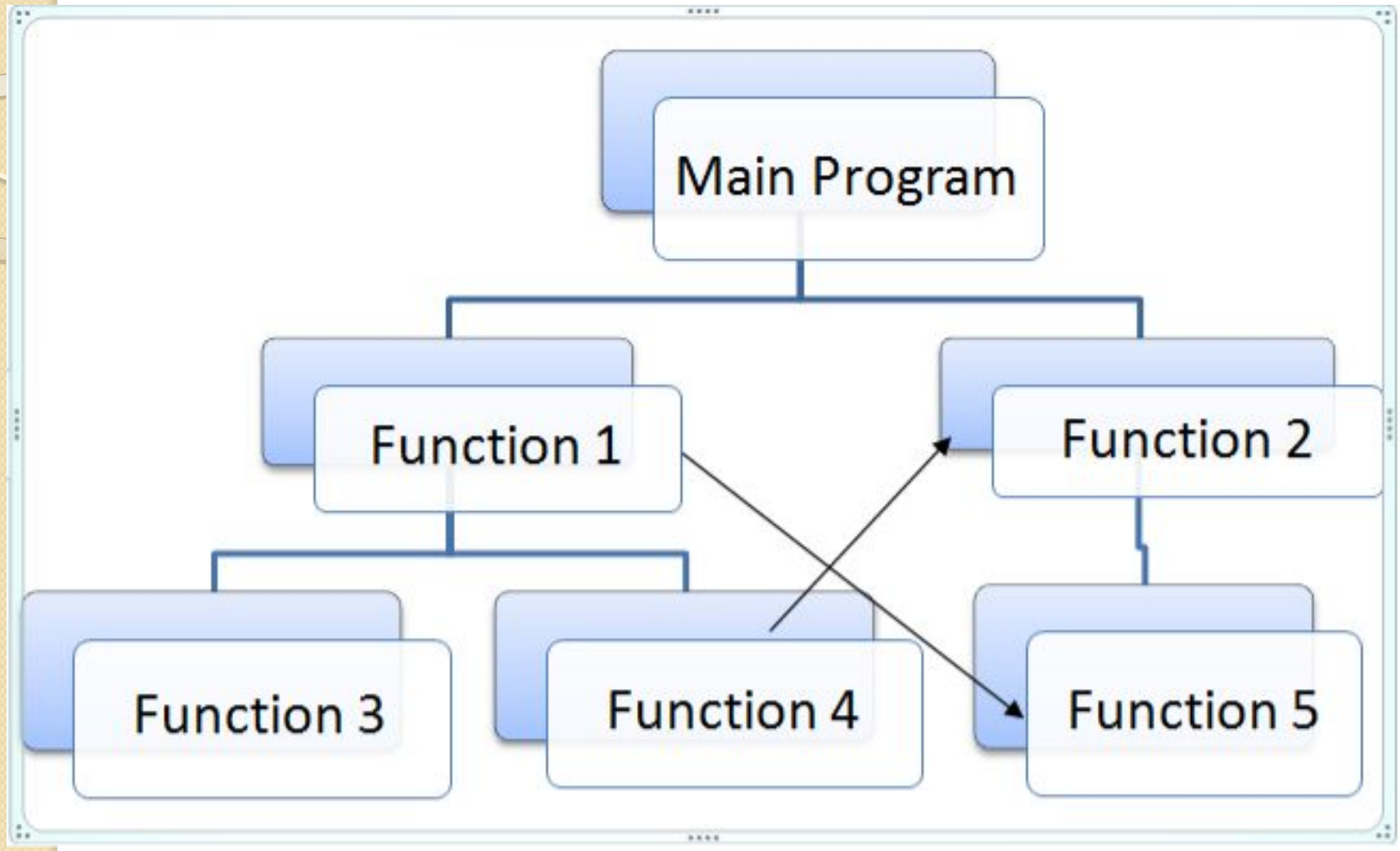
Procedure Oriented Programming

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

- The related instructions to accomplish these tasks are organized into a group called procedure.
- The primary focus is on functions .

Example

Conventional programming like COBOL,FORTRAN,C


Main Program
Function 1
Function 2
Function 3
Function 4
Function 5

Disadvantages:

- Importance is given to the operation/function rather than the data.
- Data is exposed to the whole program, so no security for data.

- Difficult to relate with real world objects.
- Difficult to create new data types reduces extensibility.

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 when ever necessary.
- Follows bottom-up approach in program design

C++

- C++ is a powerful and all-purpose programming tool developed by **Bjarne Stroustrup at Bell Labs**.
- This language is an extension of C and is by far one of the fastest object-oriented programming languages.
- C++ is super popular because of its high speed and compatibility.

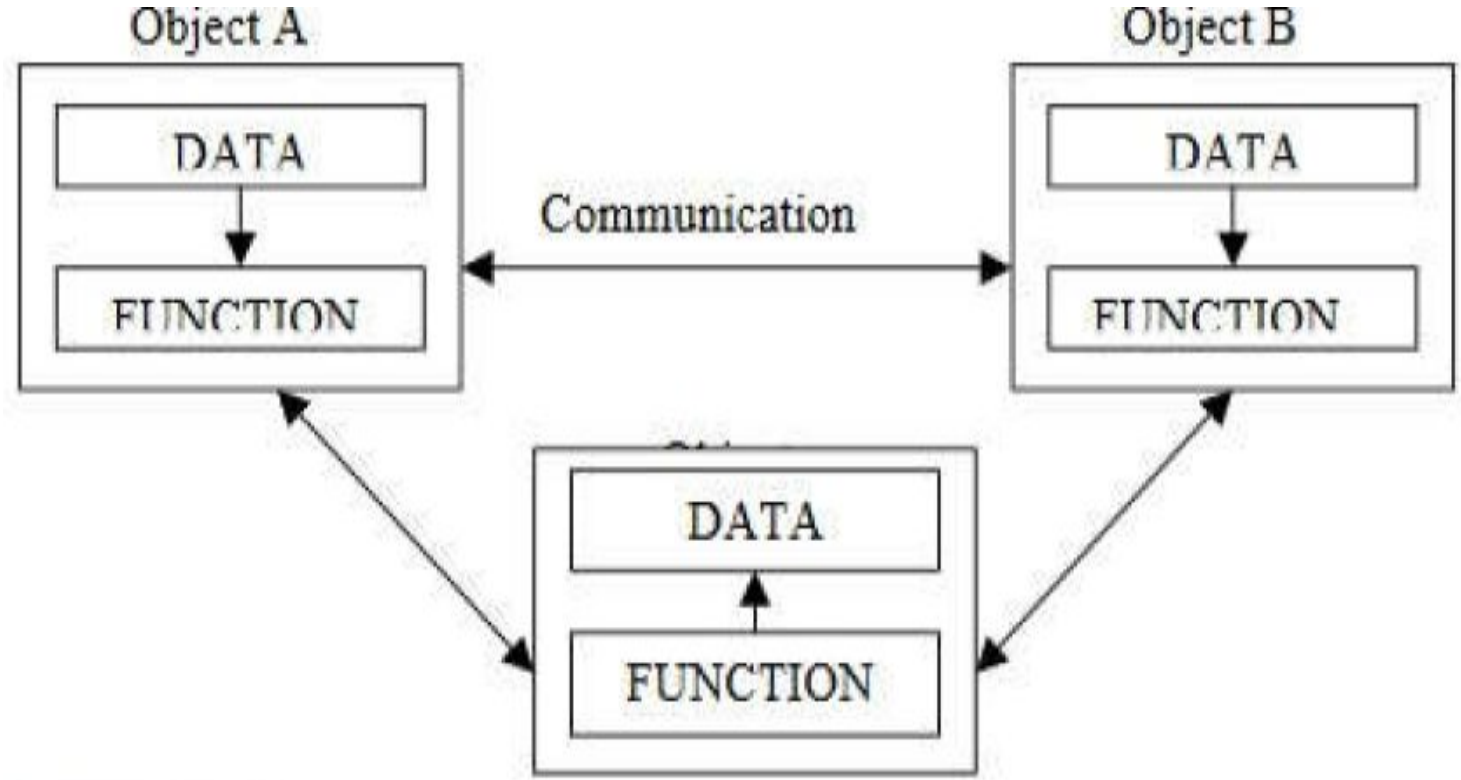
Basic concepts of OOP

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.
- OOP decompose the program into number of modules according to the entities/objects.
- Then it builds data and functions around these objects.
- The data of an object can be accessed only by the functions associated with that object.

Example:

object: STUDENT
DATA Name Regdno Marks
FUNCTIONS Total Average Display


Object A
DATA
FUNCTION
Communication
Object B
DATA
FUNCTION
DATA
FUNCTION

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 called class.
- Once a class has been defined, we can create any number of objects belonging to that class that means objects are variables of the type class.
- Thus a class is a collection of objects of similar type.
- When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated.
- Example:

Class: Fruit

Object: Mango

Fruit Mango;

Example:

```
class person
{
    char name[20];
    int id;
public:
    void getdetails(){}
};

int main()
{
   person p1; // p1 is a object
}
```

❑Abstraction

- **Hiding internal details and showing functionality** is known as abstraction.

For example: phone call, we don't know the internal processing.

Abstraction using Classes:

- ✔ We can implement Abstraction in C++ using classes.
- ✔ The class helps us to group data members and member functions using available access specifiers.
- ✔ A Class can decide which data member will be visible to the outside world and which is not.

Abstraction in Header files:

- ✔ One more type of abstraction in C++ can be header files.

  For example, consider the pow() method present in math.h header file. Whenever we need to calculate the power of a number, we simply call the function pow() present in the math.h header file and pass the numbers as arguments without knowing the underlying algorithm according to which the function is actually calculating the power of numbers.

Polymorphism

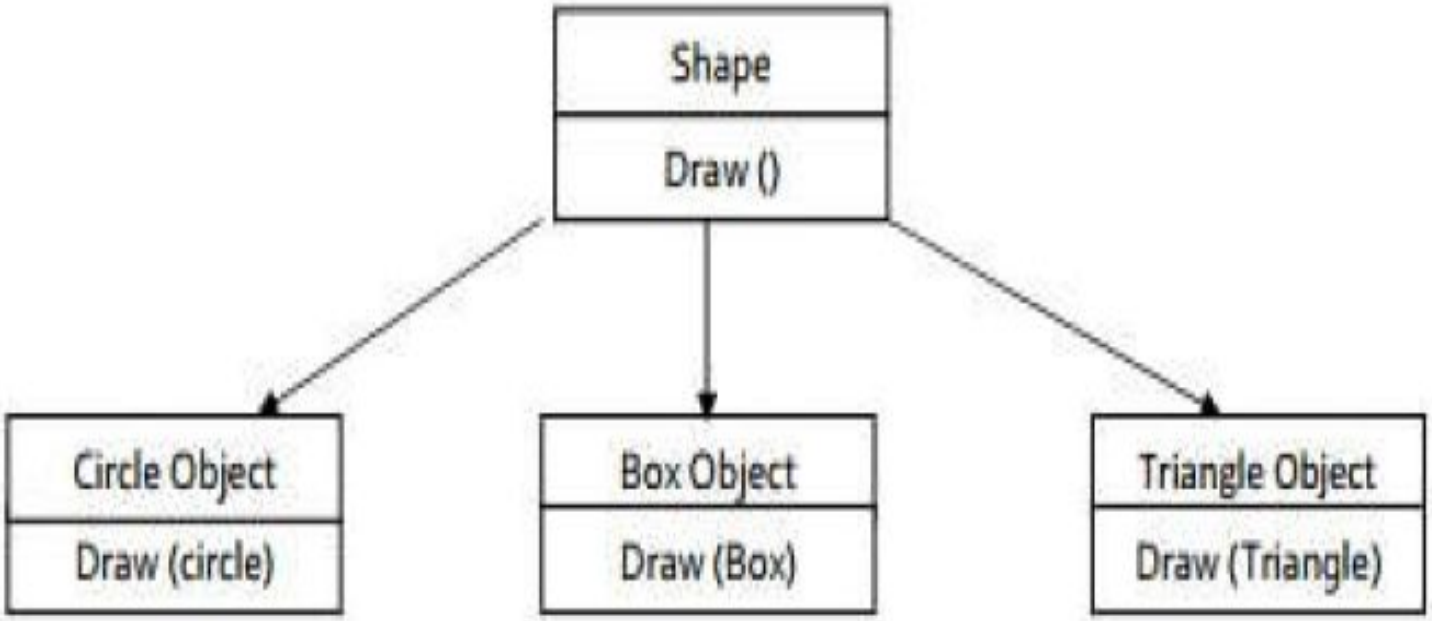
- polymorphism means having many forms.

Example:

A person at the same time can have different characteristic. Like a man at the same time is a father, a husband, an employee. So the same person posses different behaviour in different situations. This is called polymorphism.

Polymorphism in C++:

An operation and function may exhibit different behaviours in different instances. The behaviour depends upon the types of data used in the operation.


Shape
Draw ()
Circle Object
Draw (circle)
Box Object
Draw (Box)
Triangle Object
Draw (Triangle)

❑Inheritance

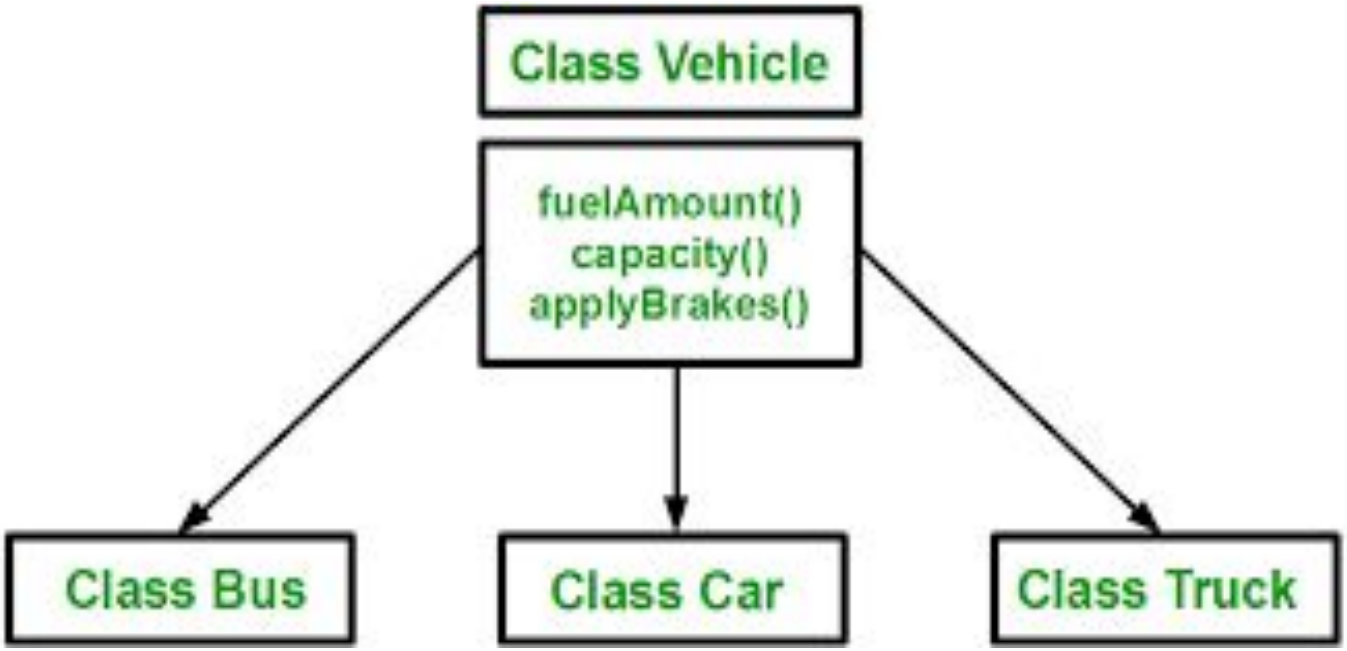
- The capability of a class to derive properties and characteristics from another class is called Inheritance.

Terms used in inheritance:

- **Sub Class**: The class that inherits properties from another class is called Sub class or Derived Class.
- **Super Class**:The class whose properties are inherited by sub class is called Base Class or Super class.
- **Reusability**: Inheritance supports the concept of "reusability", i.e. when we want to create a new class and there is already a class that includes some of the code that we want, we can derive our new class from the existing class. By doing this, we are reusing the fields and methods of the existing class.

- **Example**: Dog, Cat, Cow can be Derived Class of Animal Base Class.

Example


Class Vehicle
fuelAmount()
capacity()
applyBrakes()
Class Bus
Class Car
Class Truck

❑Encapsulation

- Encapsulation is defined as wrapping up of data and information under a single unit.
- In Object-Oriented Programming, Encapsulation is defined as binding together the data and the functions that manipulate them.
- Encapsulation also leads to *data abstraction or hiding*.

Dynamic binding

- Dynamic binding refers to linking a procedure call to that code that will execute only once.
- The code associated with the procedure is not known until the program is executed, which is also known as late binding.

❑Message Passing

- Objects communicate with one another by sending and receiving information to each other.
- A message for an object is a request for execution of a procedure and therefore will invoke a function in the receiving object that generates the desired results.
- Message passing involves specifying the name of the object, the name of the function and the information to be sent.

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
  - AI and expert systems
  - Neural networks and parallel programming
  - Decision support and office automation systems