UNIT 1
INTRODUCTION TO OBJECT ORIENTED PROGRAMMING

LH – 3HRS

PRESENTED BY
ER. SHARAT MAHARJAN
OBJECT ORIENTED PROGRAMMING (OOP)

CONTENTS (LH – 3HRS)

1.1 Overview of structured programming approach; Problems with structured programming

1.2 Object oriented programming approach

1.3 Characteristics of object oriented languages: objects, classes, data abstraction and encapsulation, polymorphism and overloading, inheritance

A BRIEF HISTORY OF C++

- C++ was developed by Bjarne Stroustrup at Bell Laboratories over a period starting in 1979. Since C++ is an attempt to add object-oriented features to C, earlier it was called as “C with Objects”. As the language developed, Stroustrup named it as C++ in 1983. The name C++ suggests “C incremented” (the ++ is an increment operator of C).
- C++ was made available outside Bell Laboratories in 1985. The first commercial C++ compiler, Cfront, was released in 1985. It was only a front-end compiler for C. The American National Standard Institute (ANSI) formed a committee for C++, in 1989.

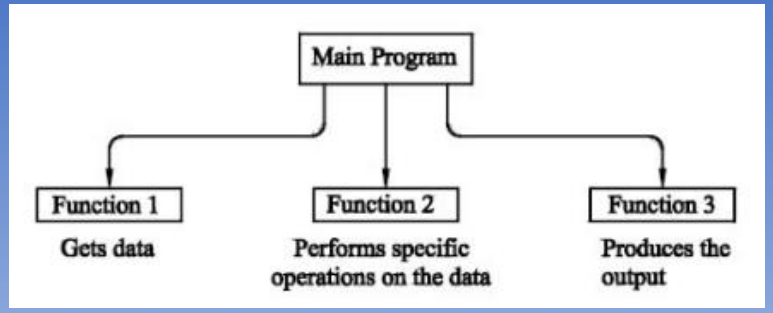
1.1 Overview of Structured Programming approach

1. **Unstructured Programming:**

- This is the programming approach where programmers put all their code in one place or say in one program.
- There is no harm as such but when line of code increases, situation starts getting unmanageable.
- For example, if the same statement sequence is needed at different locations within the program the sequence must be copied.
- A programmer can face so many problems like:
  - Duplicacy and redundancy of code
  - Code maintenance problem
  - Readability and many more
- Thus, this approach is totally useless if working on a large project. It can be only helpful while testing one or two features etc.

2. Structured Programming:

- To overcome above situation it was needed to come up with a Structured programming approach to make some code separation keeping reusability in mind.
- In this approach a new idea came up and a set of execution code was kept in a place and it was called function or procedure. (add function)
- The main program is responsible to pass data to the individual calls.
- The data is processed by the procedures and once the program has finished the resulting data is presented.
- The basic feature of procedural-oriented programming is to reuse the same code at different places in the program without copying it.



Characteristics of Structured Programming:

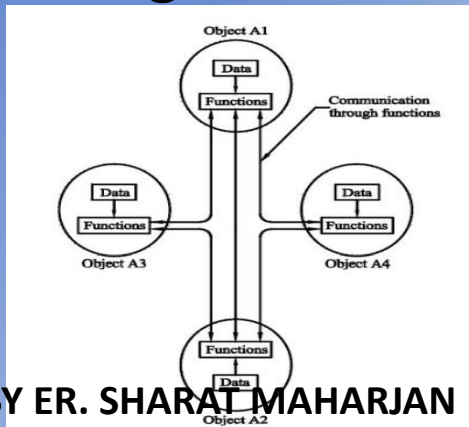
- Large programs are divided into smaller programs known as functions.
- Most of the functions share global data.
- Data is undervalued and moves openly around the system from function to function.

Problems with Structured Programming:

- The complexity of managing large projects is high.
- Functions and data structures don’t model the real world very well.
- It is difficult to create new data types with procedural languages.

1.2 Object-Oriented Programming (OOP) approach

- Object-oriented programming (OOP) developed to overcome the limitations of the procedure-oriented programming is an improved technique for developing the programs.
- In case of OOP, the data is treated as the most critical element and the primary focus is on the data and not on the procedures.
- In this technique, the data is grouped together with the functions that operate on it.
- A problem is divided into entities known as objects.
- Each object maintains its own copy of data and functions. The data cannot be accessed directly by the other objects of the program. It can only be accessed through a proper interface such as functions, as shown in figure below:
- Examples of OOP languages include Simula, C++, Python, Java etc.



IMP

Characteristics/Features of OOP:

There are certain features that have made object-oriented programming very popular. These features are as follows:

1. Objects
2. Classes
3. Encapsulation
4. Abstraction
5. Inheritance
6. Polymorphism

1. Objects:

- In object-oriented programming, a problem is divided into certain basic entities called objects. The objects can be used to represent real life objects such as people, car, and so on.
- In this type of programming, all communication is carried out between the objects. When a program is executed, the objects interact with each other by sending messages.
- The objects contain the data and the functions that can be used to manipulate the data. Each object maintains its own copy of the data and methods, which can communicate with each other through a proper channel or interface.

2. Classes:

- A class is a user-defined data type which is used to group the data and the functions together.
- The objects are the instances of a class.
- A class can also contain important members such as a constructor to create objects.
- The objects that belong to same class must have certain properties in common.
- For example, a table and a chair are the objects of the furniture class. Both the objects have certain properties in common. For example, both are made of wood and so on.

3. Encapsulation:

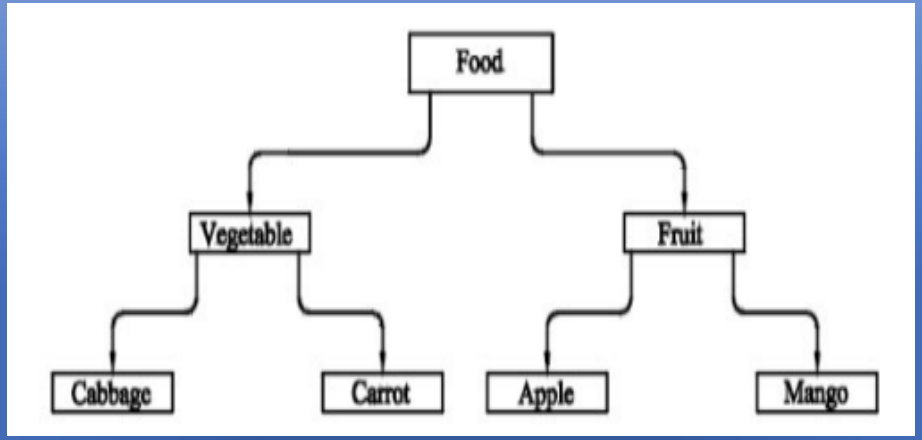
- Encapsulation is a mechanism used for wrapping up the data along with the functions that can operate on the data directly.
- This mechanism is used to keep the data safe from outside interferences.
- It hides the internal data and only provides the external interface (functions) through which it can be accessed.
- Encapsulation also provides the concept of data hiding or information hiding so that it cannot be accessed directly.

4. Abstraction:

- Data abstraction is one of the most essential and important feature of object oriented programming in C++.
- Abstraction means displaying only essential information(result) and hiding the details(pow() in math.h or add() defined by user).
- Data abstraction refers to providing only essential information about the data to the outside world, hiding the background details or implementation.
- Consider a real life example of a man driving a car. The man only knows that pressing the accelerators will increase the speed of car or applying brakes will stop the car but he does not know about how on pressing accelerator the speed is actually increasing, he does not know about the inner mechanism of the car or the implementation of accelerator, brakes etc in the car. This is what abstraction is.

5. Inheritance:

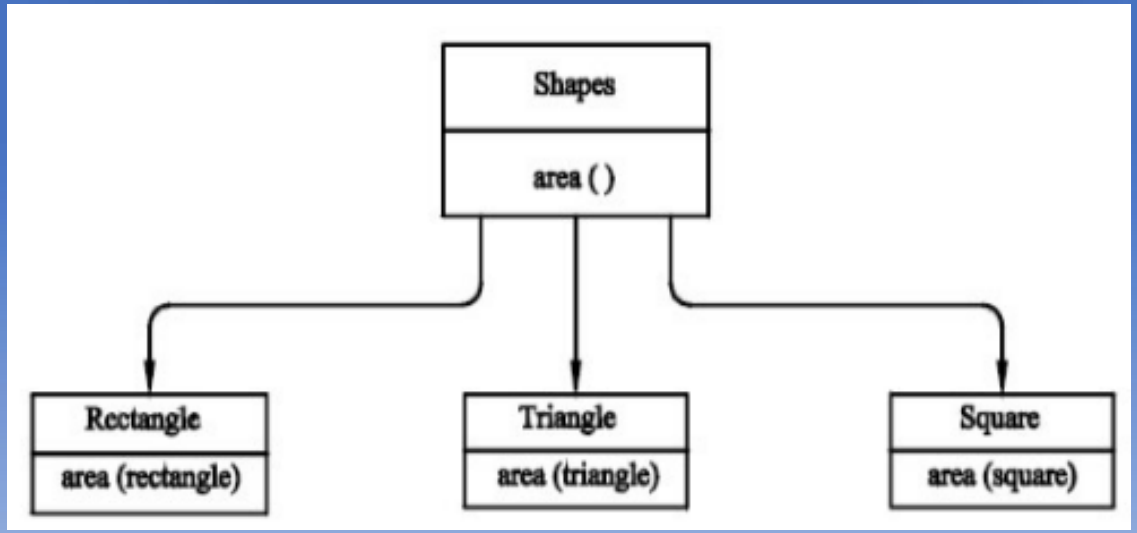
- Different kinds of objects often have certain amount of properties in common with each other.
- Inheritance is the process by which one object can acquire the properties of another object. This is called the reusability of code.
- In inheritance, whenever a new object is created, it can only define those properties that makes it unique from the other objects. Inheritance supports the concept of the hierarchical classifications.
- For example, the fruit 'Apple' is a part of the class 'Fruits', which is again a part of the class 'Food', as shown in figure below:



- In the above figure, the two classes Fruits and Vegetables, apart from inheriting certain properties from the class Food, add some properties of their own.
- The classes Apple and Mango inherit the properties of the Fruits class while the classes Cabbage and Carrot inherit the properties of the Vegetables class.

6. Polymorphism:

- Polymorphism is another feature of object-oriented programming.
- Polymorphism is a Greek term that consists of two words, poly and morph. Poly means many and morph means forms. So, polymorphism means 'one name many forms'.
- In polymorphism, the internal structure (functioning) of the operation is different but the external interface (name) is the same. When a single function name is used to perform different operations, it is known as function overloading, as shown in figure below:



IMP

Benefits of OOP:

The principle advantages are:

- By using inheritance, we can eliminate redundant code and extend the use of existing classes.
- We can build the programs from standard working modules that communicate with one another, rather than having to start writing the code from scratch which leads to saving of development time and higher productivity.
- The new technology promises greater programmer productivity, better quality of software and lesser maintenance cost.
- OOP systems can be easily upgraded from small to large systems.
- It is possible that multiple instances of objects co-exist without any interference.
- It is very easy to partition the work in a project based on objects.
- The principle of data hiding helps the programmer to build secure programs which cannot be invaded by the code in other parts of the program.

THANK YOU FOR ATTENTION