19PMAE06

PROGRAMMING WITH C++

M.Sc. MATHEMATICS

$\underline{\text{III}}$ - SEMESTER

M. POONGUZHALI

GUEST LECTURER

DEPARTMENT OF MATHEMATICS

GOVERNMENT ARTS AND SCIENCE COLLEGE

KOMARAPALAYAM - 638183

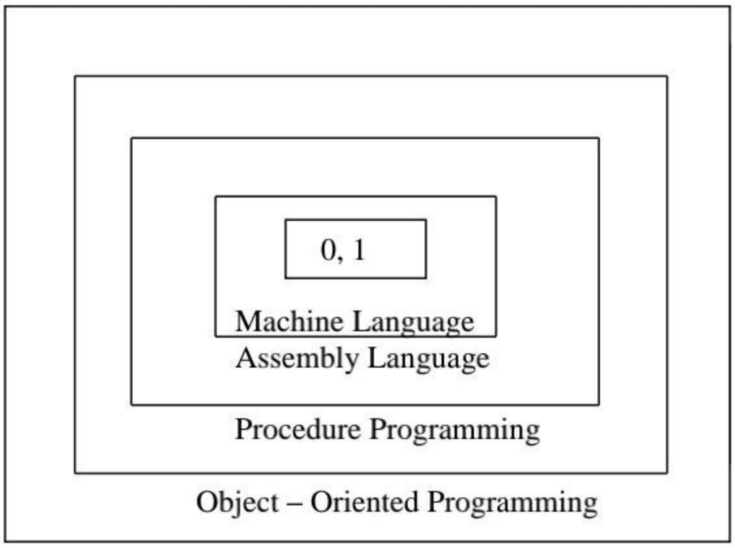
NAMAKKAL (DT).

Unit I: Software Evolution – Procedure oriented Programming – Object oriented programming paradigm – Basic concepts of object oriented programming – Benefits of oops – Object oriented Languages – Application of OOP – Beginning with C++ - what is C++ - Application of C++ - A simple C++ Program – More C++ Statements – An Example with class – Structure of C++ Program.

UNIT - I

SOFTWARE EVOLUTION

- The software evolution has had distance phases or "layers" of growth. These layers were built up one by one over the last five decades, with each layer representing an improvement over the previous one.
- However, the analogy fails if we consider the life of these layers. In software systems, each of the layers continues to be functional, whereas in the case of trees, only the uppermost layer is functional.
- To build today's complex software it is just not enough to put together a sequence of programming statements and sets of procedures and modules; we need to incorporate sound construction techniques and program structures that are easy to comprehend, implement and modify.



- These including techniques such as modular programming, top-down programming, bottom up programming and structured programming.
- The primary motivation in each has been the concern to handle the increasing complexity of programs that are reliable and maintainable.
- These techniques have become popular among programmers over the last two decades. With the advent of languages such as C, structured programming became vet popular and was the main technique of the 1980s.
- Structured programming was a powerful tool that enabled programmers to write moderately complex programs fairly easily.

- However, as the programs grew larger, even the structured approach failed to show the desired results in terms of bug-free, easy-to-maintain, and reusable programs.
- Object-Oriented Programming (OOP) is an approach to program organization and development that attempts to eliminate some of the pitfalls of conventional programming methods by incorporating the best of structured programming features with several powerful new concepts.
- It is a new way of organizing and developing programs and has nothing to do with any particular language. However, not all languages are suitable to implant the OOP concepts easily."

CHARACTERISTICS OF PROCEDURE-ORIENTED PROGRAMMING

Following are the striking features of Procedure-Oriented Programming:

- Emphasis is on doing things.
- Large programs are divided into smaller programs known as functions.
- Most of the functions share global data.
- Data move openly around the system form functions to function.
- Functions transform data form one form to another.
- Employs top-down approach in program design.

STRIKING FEATURES OF OBJECT-ORIENTED PROGRAMMING

Following are the striking features of Object-Oriented Programming:

- 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 OOP. They are:

- Classes
- Objects
- Data abstraction and encapsulation.
- Inheritance

- Dynamic binding
- Message passing

Classes

- ➢ The entire set of data and code of an object can be made a user-defined data type with the help of a 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 thus a collection of objects of similar type.

Objects

- ➢ Objects are the basic run-time entities in an object-oriented system. They may represent a person, a place, a blank account.
- ➢ Objects are variables of the type class.

Data Abstraction and Encapsulation

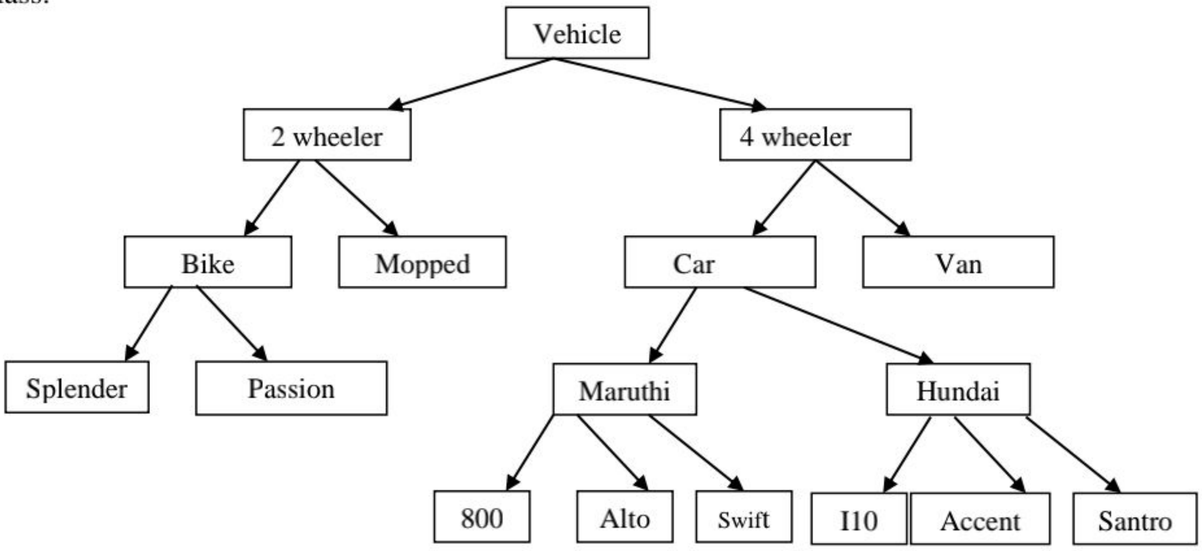
- ➢ The wrapping up of data and functions into a single unit is known as encapsulation.

Class = Data Members + Member Functions (Methods)

- ➢ The data is not accessible to the outside world, and only those functions which are wrapped in the class can access it.
- ➢ Abstraction refers to the act of representing essential features without including the background details or Explanations.

Inheritance

- ➢ Inheritance is the process by which objects of one class acquire the properties of objects of another class.

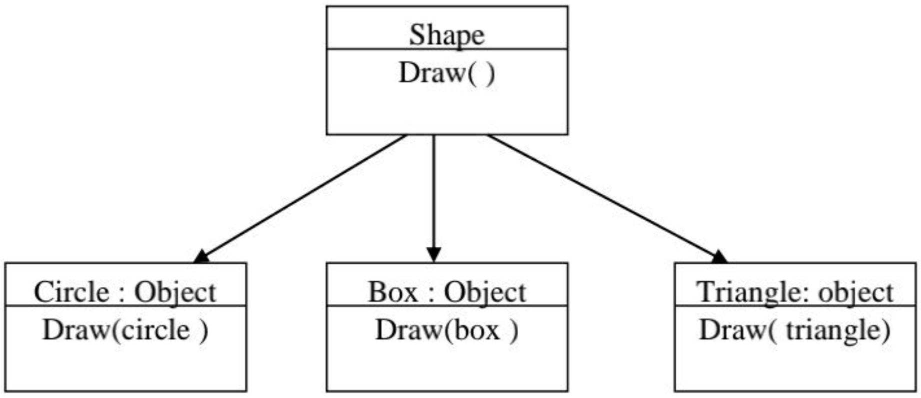


- Inheritance provides the idea of reusability.
- Example: Car Maruthi 800 posses the general property of Maruthi and its own characteristics which has the common properties of a car.

Polymorphism

- Polymorphism means one name, multiple forms.
- It allows us to have more than one function with the same name in a program.
- It also allows overloading of operators so that an operation can exhibit (show) different behaviors in different instances.

Example



Dynamic Binding

- Dynamic binding means that the code associated with a given procedure call is not known until the time of the call at run-time.
- Ex : The method draw( ) in the previous example will execute according to the parameters that are given at the run time. If the parameter are of circle Draw(circle) is executed.

Message passing

- Object oriented programming consists of a set of objects that communicate with each other.
- Object oriented programming involves the following steps:
    - Create the class
    - Create objects for the class
    - Establish communication among classes.
- Objects communicate with one another by sending and receiving information.

<u>BENEFITS OF OOPS</u>

- Through inheritance, we can eliminate redundant code and extend the use of existing classes.

- We can build programs form the standard working modules that communicate with one another, rather than having to start writing the code form scratch. This leads to saving of development time and 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 have multiple instances of an object to co-exist without any interference.
- 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 impermeable form.
- Object-oriented systems can be easily upgraded form 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.

OBJECT ORIENTED LANGUAGES

The languages should support several of the OOP concepts to claim that they are object-oriented. Depending upon the features they support, they can be classified into the following two categories:

1. Object based programming languages
2. Object-oriented programming languages

Object-based programming is the style of programming that primarily supports encapsulation and object identity. Major features that are required for Object-based programming are:

- ✶ Data Encapsulation
- ✶ Data Hiding and Access Mechanism
- ✶ Automatic Initialization and Clear-up of Objects
- ✶ Operator Overloading

Ada is a typical Object based programming language

Object-oriented programming incorporates all of Object based programming features along with two additional features , namely, **Inheritance** and **Dynamic Binding.** Object-oriented programming can therefore be characterized by the following statement:

Object-oriented programming = Object-based + Inheritance + Dynamic Binding

Languages that support these features include C++, Smalltalk and Java.

APPLICATION OF OOP

- Applications of OOP are beginning of gain importance in many areas. The most popular application of object oriented programming, up to now has been in the area of user interface design such as windows.

- Real – business systems are often much more complex and contain many more objects with complicated attributes and methods.
- The promising area for application of OOP includes:
    - Real-time systems
    - 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

WHAT IS C++

- C++ is an object oriented programming language.
- it was developed by Bjarne stroustrup at AT & T Bell Laboratories in Murray Hill, New Jersay, USA, in the early ,1980's.
- Stroustrup, an admirer of simula67 and a strong supports of C, wanted to combine the best of both the languages and create a more powerful language that could support OOP features and still retain the power and elegance of C.
- C++ is an extension of c with a major addition of the class construct features of simula67.
- However, later in 1983, the name was changed to C++.
- During the early 1990's the language underwent a number of improvements and changes.
- In November 1997, the ANSI/ISO standards committee standardized these changes and added several new features to the languages specification.
- The most important facilities that c++ adds onto C are classes, inheritance, function overloading, and operator overloading.
- These features enable creating of abstract data types, inherit properties from existing data types and support polymorphism.

APPLICATION OF C++

Following are the applications of C++:

- C++ is a versatile language for handling very large programs.
- C++ is suitable for any programming task including development of editors, compilers, databases, communication system and any complex real-life application system.

- C++ allows us to create hierarchy-related objects; we can build special object-oriented libraries which can be used later by many programmers.
- C++ programs are easily maintainable and expandable. When a new feature needs to be implemented, it is very easy to add to the existing structure of an object.
- It is expected that c+ will replace C as a general-purpose language in the near future.

A SIMPLE C++ PROGRAM

Let us begin with a simple example of a C++ program that prints a string on the screen

```
#include<iostream.h>
using namespace std;
int main()
{
cout<<"c++ is better than c";
return 0;
}
```

AN EXAMPLE WITH CLASS

One of the major features of c++ is classes. Following program illustrates the use of class.

```
#include<iostream.h>
using namespace std;
class Person
{
        char name[30];
        int age;
        public:
                void getdata(   void);
                void display(void);
};
void Person::getdata(void)
{
        cout<<"Enter the name:";
        cin>>name;
        cout<<"Enter the age:";
        cin>>age;
    }
    void Person::display(void)
    {
        cout<<"\n Name:"<<name;
        cout<<"\n age:"<<age;
    }
    void main()
    {
        Person ob;
        ob.getdata();
        ob.display();
    }
```

STRUCTURE OF C++ PROGRAM

- A typical C++ program would contain four sections.
- These sections may be placed in separate code files and then complied independently or jointly.

INCLUDE FILES
CLASS DECLARATION
MEMBER FUNCTION DEFINTION
MAIN FUNCTION (PROGRAM)

- It is a common practice to organize a program into three separate files.
- The class declarations are placed in a header file and the definitions of member functions go into another file.
- This approach enables the programmer to separate the abstract specification of the interface from the implementation details.
- The main program that uses the class is placed in a third file which "includes "the previous two files as well as any other files required.

PROGRAM FEATURES

- The C++ program is a collection of functions.
- Every C++ program must have a main().
- C++ is a free-form language.

- With a few exceptions, the compiler ignores carriage returns and white spaces.
- Like C, the C++ statements terminate with semicolons.

COMMENTS

- Comments start with a double slash symbol (//)and terminate at the end of the line. A comment any start anywhere in the line, and whatever follows till the end of the line is ignored.
- The double slash comment is basically a single line comment.
- The C comments symbol /*,*/ are still valid and are more suitable for multiline comments.
- **Ex:**
```
/*      This is an example C++ program to illustrate
        some of its features              */
```

THE IOSTREAM.H FILE

#include<iostream.h>

- This directive causes the preprocessor to add the contents of the iostream file to the program.
- It contains declarations for the identifier cout and the operator <<. some old versions of C++ use a header file called iostream.h
- The header file iostream should be included at the beginning of all programs that use input/output statements.

INPUT OPERATOR

- The statement cin is an input statement and causes the program to wait for the user to type in a number.
- The operator >> is known as extraction or get form operator.
- It extracts the value form the keyboard and assigns it to the variable on its right.
- This corresponds to the familiar scanf() operation. Like <<, the operator>> can also be overloaded.

OUTPUT OPERATOR

- The identifier cout is a predefined object that represents the standard output stream in C++.
- The operator << is called the insertion or put to operator.
- It inserts the contents of the variable on its right to the screen.
- The operator << is the bit-wise left-shift operator and it can still be used for this purpose.