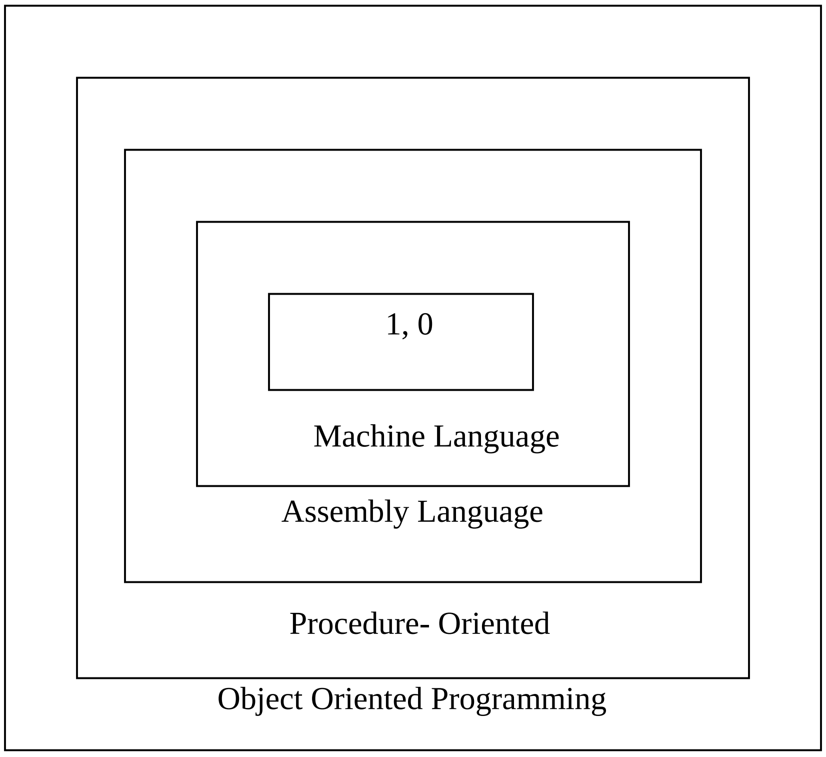
Software Crisis

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, and they are becoming more and more critical in view of the increasing complexity of software systems as well as the highly competitive nature of the industry. 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 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?
- How to improve the quality of software?
- How to manage time schedules?

Software Evaluation

Ernest Tello, A well known writer in the field of artificial intelligence, compared the evolution of software technology to the growth of the tree. Like a tree, the software evolution has had distinct phases "layers" of growth. These layers were building up one by one over the last five decades as shown in fig. 1.1, with each layer representing and improvement over the previous one. However, the analogy fails if we consider the life of these layers. In software system each of the layers continues to be functional, whereas in the case of trees, only the uppermost layer is functional



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 material*s". 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.

With the advent of languages such as c, structured programming became very popular and was the main technique of the 1980's. 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 result 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 implement the OOP concepts easily.

Procedure-Oriented Programming

In the procedure oriented approach, the problem is viewed as the sequence of things to be done such as reading, calculating and printing such as cobol, fortran and c. The primary focus is on functions. A typical structure for procedural programming is shown in fig.1.2. The technique of hierarchical decomposition has been used to specify the tasks to be completed for solving a problem.

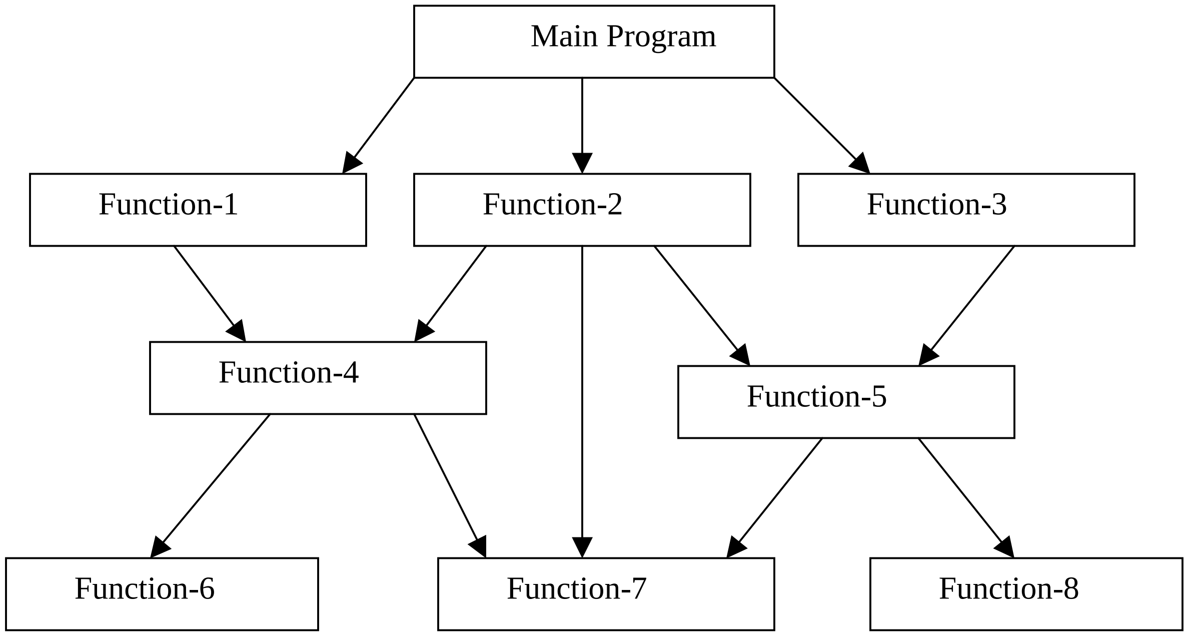


Fig. 1.2 Typical structure of procedural oriented programs

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*. We normally use flowcharts to organize these actions and represent the flow of control from one action to another.

In a multi-function program, many important data items are placed as global so that they may be accessed by all the functions. Each function may have its own local data. Global data are more vulnerable to an inadvertent change by a function. In a large program it is very difficult to identify what data is used by which function. In case we need to revise an external data structure, we also need to revise all functions that access the data. This provides an opportunity for bugs to creep in.

Another serious drawback with the procedural approach is that we do not model real world problems very well. This is because functions are action-oriented and do not really corresponding to the element of the problem.

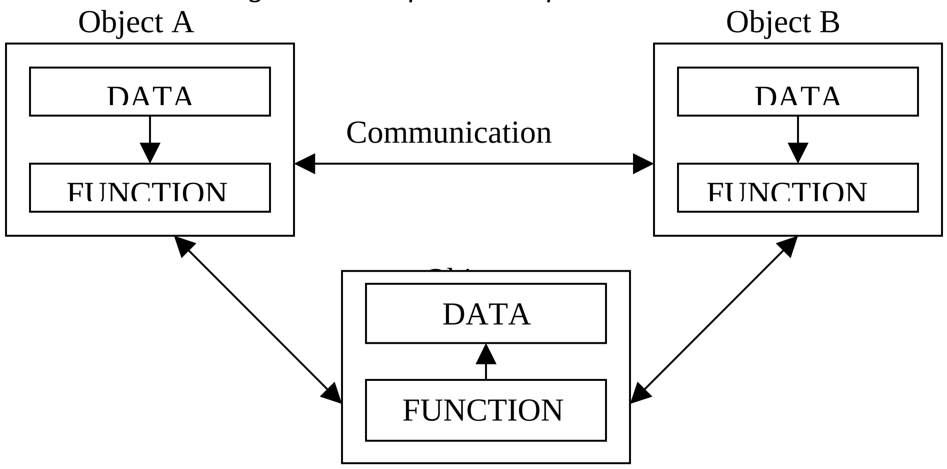
Some Characteristics exhibited by procedure-oriented programming are:

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

Object Oriented Paradigm

The major motivating factor in the invention of object-oriented approach is to remove some of the flaws encountered in the procedural approach. OOP treats 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 function that operate on it, and protects it from accidental modification from outside function. OOP allows decomposition of a problem into a number of entities called objects and then builds data and function around these objects. The organization of data and function in object-oriented programs is shown in fig.1.3. The data of an object can be accessed only by the function associated with that object. However, function of one object can access the function of other objects.

Organization of data and function in OOP



Some of the features of object oriented programming are:

- 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 ties together in the data structure.
- Data is hidden and cannot be accessed by external function.
- Objects may communicate with each other through function.
- New data and functions can be easily added whenever necessary.
- Follows bottom up approach in program design.

Object-oriented programming is the most recent concept among programming paradigms and still means different things to different people.