# Paradigms of Programming

N.Kranti Kumar,201301017,UG 4

## I. Introduction

A Programming paradigm is a style or way of programming. Some languages make it easy to write in some paradigms but not others.Programming is a rich discipline and practical programming languages are usually quite complicated. Fortunately, the important ideas of programming languages are simple.This chapter is intended to give readers with some programming experience a running start for these ideas. Although we do not give formal definitions, we give precise intuitions and good references so that interested readers can quickly get started using the concepts and languages that implement them.
There are 4 Major Paradigm's in Programming:

- Imperative.
- Logic.
- Functional.
- Object Oriented.

### A. *Imperative Programming*

The 'first do this, next do that' is a short phrase which really in a nutshell describes the spirit of the imperative paradigm. The basic idea is the command, which has a measurable effect on the program state. The phrase also reflects that the order to the commands is important. 'First do that, then do this' would be different from 'first do this, then do that'.
This can only be understood mathematically by associating a sequence of values with x let us say x1, x2,..., where xt denotes the value the variable x has at some time t. Thus the above statement can be translated into mathematics as

$$x(t + 1) = x(t) + 1 \qquad (1)$$

### B. *Functional Programming*

Functional programming is in many respects a simpler and more clean programming paradigm than the imperative one. The reason is that the paradigm originates from a purely mathematical discipline: the theory of functions. The imperative paradigm is rooted in the key technological ideas of the digital computer, which are more complicated, and less 'clean' than mathematical function theory.The high level of abstraction, especially when functions are used, supresses many of the details of programming and thus removes the possibility of commiting many classes of errors. The lack of dependence on assignment operations, allowing programs to be evaluated in many different orders. This evaluation order independence makes function-oriented languages good candidates for programming massively parallel computers. The absence of assignment operations makes the function-oriented programs much more amenable to mathematical proof and analysis than are imperative programs, because functional programs possess referential transparency.

### C. *Logic Programming*

While the functional paradigm emphasises the idea of a mathematical function, the logic paradigm focusses on predicate logic, in which the basic concept is a relation. Logic languages are useful for expressing problems where it is not obvious what the functions should be. Thus, for example where people are concerned, it is natural to use relations.The system solves the problem, so the programming steps themselves are kept to a minimum.Proving the validity of a given program is simple.

### D. *Object Oriented Programming*

The object-oriented paradigm has gained great popularity in the recent decade. The primary and most direct reason is undoubtedly the strong support of encapsulation and the logical grouping of program aspects. These properties are very important when programs become larger and larger. The underlying, and somewhat deeper reason to the success of the object-oriented paradigm is probably the conceptual anchoring of the paradigm. An object-oriented program is constructed with the outset in concepts, which are important in the problem domain of interest. In that way, all the necessary technicalities of programming come in second row.A new class (called a derived class or subclass) may be derived from another class (called a base class or superclass) by a mechanism called inheritance. The derived class inherits all the features of the base class: its structure and behavior(response to messages). In addition, the derived class may contain additional state (instance variables), and may exhibit additional behavior (new methods to resond to new messages). Significantly, the derived class can also override behavior corresponding to some of the methods of the base class: there would be a different method to respond to the same message. Also, the inheritance mechanism is allowed even without access to the source code of the base class. The ability to use inheritance is the single most distinguishing feature of the OOP paradigm. Inheritance gives OOP its chief benefit over other programming paradigms - relatively easy code reuse and extension without the need to change existing source code.The mechanism of modeling a program as a collection of objects of various classes, and furthermore describing many classes as extensions or modifications of other classes, provides a high degree of modularity.

## II. Conclusion

There are several paradigms of programming which are employed for clean,efficient methods of Programming.Every

Programming Paradigm satisfies the needs of the required programming aspect according to the requirement and need of the Project.

## III. Biblography

- www.cs.bham.ac.uk