Name:- Raman Sharma

Course name:- CCA

Course title :- Introduction to Programming Language

Date : 09-08-2021

Introduction to Programming Languages

Date
Page No. 1

Special Features of the text

The following are significant features of the text as compared to the standard texts

- Syntax! an introduction to regular expression, Context-free grammars, parsing, attribute grammars and abstract grammars
- Semantics! introductory treatment of algebraic, axiomatic denotational and operational semantics.
- Programming paradigms! the major programming paradigms are prominently featured.

- Languages design principal. Intently some prog. morning language ML design principle were given prominence.

Readership:-

This book is intended as an undergraduate text in the theory of programming languages. To gain maximum benefits from the text, the reader have should experience in a high-level programming languages such as pascal, Modula-2, C++, ML or Common Lisp, machine organization and programming and discrete mathematics.

Programming is a spectator sport. To gain maximum benefit from the text, the reader should construct programs in each of the paradigms, and write semantic specification, and implement a small programming languages.

Organization

Since the subject area PL: Programming languages as described in the "ACM/IEEE Computing Curricula 149" consists of a minimum of 47 hrs of lecture, the text contains more material can be covered in a single course.

Pedagogy

The text pedagogical support through various exercise and laboratory projects. Some of the projects are suitable for small group assignment. The exercise include programming languages Concept. For the students to gain maximum benefit from the text, the student should have access to a logic programming languages (su as Prolog), a moder functional language (suc as Scheme, ML or Haskell), a concurrent programming languages (Ada, SR, or Ocam), an objec oriented programming Languages (C++, Small-

Talk, Eiffel & Modula-3), and a modern.

1.1 Models of Computation

Computation model begin with a set of values. The values can be separated into two group primitive and composited. The primitive value (or type) are usually number, boolean values, and characters. The composite values (or type) are usually arrays, records and recursively defined values.

The Functional Model

The functional model of computation consists of a set of values, functions, and the operation

of Function application Function may be named and may be Composed with other Functions.

The Logic Model

The logic Model of Comput on is based on relat and logic inference. Pro grams consist of definitions of relation and Computations are inferences.

The imperative Model

The imperative model of computation consists of a State and the operation of 'assignment' which is used to modify the State.

Programs Consist of sequences of Commands and computations are Change of the state.

other Models

Programs in the concurrent programming model Consist of multiple processes or tasks which may exchange information. The Computation may occur Concurrently or in any order. Concurr -ency programming is primarily Concerned

Computbility

The method of Computation provided in a progra -mming languages. Most programming languages

utilize more than on
mode of Computatio
implement by the
programing language
One mode of
predomination but
provide imperative
constructs as well
while, Miranda.

1.2 Syntax and semantics

The notice used in
the functional and logi
models tends to reflect
common mathematical
practice and thus,
it tends toward simplic
and regularity. on
the other hands
the notation used for
the imperative mode

1.3 Pragmatics

Pragmatics is concerned about the usability of the languages, application area, ease of implementation and use, and the language's success in fulfilling its design goals. For a languages to have wide applicability it must make provision for abstraction, generalization and modularity. Abstraction permits the suppression of detail and provides constructs which permit the extension of programming languages.

1.4 Languages Design Crite

Programming languages are largely determined by the orientation the language designers to the area of readability, Writeability and efficient execution. Some languages are largely determined by the necessity for efficient implementation and execution. Other are designed to be faithful to a Computational

1.5 Further Reading

For a programming languages which presents programming languages from the virtual machine point of view see part [24]. For a programming languages from the virtual machine point of view of denotational semantics see Tennent [30]. For a programming languages text which present programming languages from a programming methodology point of view see Hehner [11].