CS4022D Principles of Programming Languages

Course Overview

Saleena N
NIT Calicut

July 30, 2021

Outline

Programming Language - Introduction

- Concepts
- Constructs

Programming Language Definition

- Syntax
- Semantics
- Type System

Syntax

- Concerned with the *structure* of the program
- Formally specified
- Context Free Grammar / BNF

$S \rightarrow id = E$
$E \rightarrow E + E \mid id$

Semantics

- In linguistics - *study of meaning*
- Concerned with the meaning of syntactically valid programs
- Semantics of C language assignment statement: $id = E$?
  - describes how programs are evaluated / run-time behavior of programs ?

Semantics

- Required by programmers and implementers (writing compiler/interpreter)
- Mostly described informallly
- Formal Semantics: Approaches
  - Operational
  - Denotational
  - Axiomatic

Type System

- Types in the language and the rules for assigning types to language constructs
- Typing rule for C language assignment statement: $id = E$?
- Static checking to avoid certain run time errors
- Static Semantics

Programming Paradigm

- Pattern of problem solving thought
- Functional, Imperative, Logic, Object-oriented
- Functional - computation viewed as mathematical function mapping input to output
- Imperative - based on the Von Neumann model - stored programs, variables - program as a series of commands

Functional Programming

- Lambda Calculus (Alonzo Church) - foundation of functional programming
- Original functional programming language - LISP developed by John Mc Carthy - theorem proving, rule based systems, earlier AI applications - Scheme is a variant of LISP
- Pure Functional Programming - Computation viewed as mathematical function mapping input to output

Structural Operational Semantics (small-step)

- Language of Arithmetic Expression
- Formal Semantics - introduction
  - Evaluation rules
  - Evaluation Derivation
  - Properties

Lambda Calculus

- Syntax
- Semantics
- Programming in Lambda Calculus

Type Systems

- Language of Typed Arithmetic Expressions
  - Typing rules
  - Typing derivations
- Type Safety
- Typed Lambda Calculus

Extensions to Lambda Calculus

- Constructs: Let binding, Sequencing
- Data Structures: Pairs, Tuples, Records, Sums, Variants
- Impure features: References
- Exceptions

Subtyping

- Basics
- Subtyping - Records, Variants, Functions

Case Studies

- Object Oriented Language