Subject: Principles of Programming Languages (POPL) Course Code: COMP 301

Type: Core

Course Description:

This course will introduce many of the principles of programming language design and implementation via comparative studies of programming languages thus providing an insight to the different programming language concepts and constructs. Different programming paradigms – Procedural, functional, logic etc. would also be covered.

Course Objectives:

- To provide an introduction to formalisms for specifying syntax and semantics of programming languages, including an introduction to the theory of formal languages;
- To provide an exposure to core concepts and principles in contemporary programming languages, and
- To explore various important programming methodologies, such as functional programming, logic programming, programming with abstract data types

Text book: *Concepts of Programming Languages*, Eighth Edition. Robert W. Sebesta. Pearson Education, Inc.

References for further readings:

Programming Languages Design and Implementation, Fourth Edition. Terrence W. Pratt, Marvin V. Zelkowitz. Pearson Education, Inc.

Programming Languages: Concept and Constructs, Second Edition. Ravi Sethi. Addision-Wesley.

Essentials of Programming Languages, Second Edition. Daniel P. Friedman, Mitchell Wand and Christopher T. Haynes. MIT Press.

Lectures:

Lecture 1:

Introduction

- Reasons for Studying Concepts of Programming Languages
- Programming Domains
- Language Evaluation Criteria
- Influences on Language Design
- Language Categories
- Language Design Trade-offs
- Implementation Methods
- Programming Environments

Lecture 2:

Evolution of the Major Programming Languages

Lecture 3

Describing Syntax and Semantics

- Introduction
- The General Problem of Describing Syntax
- Formal Methods of Describing Syntax
- Attribute Grammars
- Describing the Meanings of Programs: Dynamic Semantics

Lecture 4:

Lexical and Syntax Analysis

- Introduction
- Lexical Analysis
- The Parsing Problem
- Recursive-Descent Parsing
- Bottom-up Parsing

Lecture 5:

Names, Bindings, Type Checking, and Scopes

- Introduction
- Names
- Variables
- The Concept of Binding
- Type Checking

Lecture 6

Type Checking and Scopes continued…

- Strong Typing
- Type Equivalence
- Scope
- Scope and Lifetime
- Referencing Environments
- Named Constants

Lecture 7

Data Types

- Introduction
- Primitive Data Types
- Character String Types
- Array Types
- Associative Arrays
- Record Types
- Union Types
- Pointer and Reference Types

Lecture 8

Expression and Assignment Statements

- Introduction
- Arithmetic Expressions
- Overloaded Operators
- Type Conversions
- Relational and Boolean Expressions
- Short-Circuit Evaluation
- Assignment Statements
- Mixed-mode Assignment

Lecture 9

Subprograms

- Introduction
- Fundamentals of Subprograms
- Design Issues of Subprograms

Local Referencing Environments

Parameter- Passing Methods

Parameters That Are Subprograms

Lecture 10

Subprograms continued…

Overloaded Subprograms

Generic Subprograms

Design Issues for Functions

User-Defined Overloaded Operators

Coroutines

Lecture 11

Implementing Subprograms

The General Semantics of Calls and Returns

Implementing “Simple” Subprograms

Implementing Subprograms with Stack-Dynamic Local Variables

Nested Subprograms

Blocks

Implementing Dynamic Scoping

Lecture 12

Abstract Data Types and Encapsulation Constructs

The Concept of Abstraction

Introduction to Data Abstraction

Design Issues for Abstract Data Types

Language Examples

Parameterized Abstract Data Types

Encapsulation Constructs

Naming Encapsulations

Lecture 13:

Functional Programming Languages

Introduction

Mathematical Functions

Fundamentals of Functional Programming Languages

The First Functional Programming Language: LISP

COMMON LISP

ML

Haskell

Applications of Functional Languages

A Comparison of Functional and Imperative Languages

Lecture 14

Logic Programming Languages

Introduction

A Brief Introduction to Predicate Calculus

Predicate Calculus and Proving Theorems

An Overview of Logic Programming

The Origins of Prolog

The Basic Elements of Prolog

The Deficiencies of Prolog

Applications of Logic Programming