**PRINCIPLES OF PROGRAMING LANGUAGES**

**SYLLABUS**

**UNIT-I**

##### Preliminary Concepts: Reasons for studying, Concepts of programming languages, programming domains, Language Evaluation criteria, influences on language design, Language categories, Programming Paradigms—Imperative, Object Oriented, Functional Programming, Logic programming. Programming Language Implementation – Compilation and Virtual machines, programming environments. Syntax and Semantics: general Problem of describing Syntax and Semantics, formal methods of describing syntax - BNF, EBNF for common programming languages features, parse trees, ambiguous grammars, attribute grammars, denotational semantics and axiomatic semantics for common programming language features.

**UNIT-II**

**Data types:** Introduction, primitive, character, user defined, array, associative, record, union, Pointer and reference types, design and implementation uses related to these types. Names, Variable, concept of binding, type checking, strong typing, type compatibility, named constants, Variable initialization. Expressions and Statements: Arithmetic relational and Boolean expressions, Short circuit evaluation mixed mode assignment, Assignment Statements, Control Structures – Statement Level, Compound Statements, Selection, Iteration, Unconditional Statements and guarded commands.

**UNIT-III**

**Subprograms and Blocks:** Fundamentals of sub-programs, Scope and lifetime of variable, static and dynamic scope, Design issues of subprograms and operations, local referencing environments, parameter passing methods, overloaded sub-programs, generic sub-programs, parameters that are sub-program names, design issues for functions user defined overloaded operators, co routines.

**UNIT-IV**

**Abstract Data types:** Abstractions and encapsulation, introductions to data abstraction, design issues, language examples, C++ parameterized ADT, object oriented programming in small talk, C++, Java, C#, Ada 95.

**Concurrency:** Subprogram level concurrency, semaphores, monitors, massage passing, Java threads, C# threads.

**Exception handling:** Exceptions, exception Propagation, Exception handler in Ada, C++ and Java.

**Logic Programming Language:** Introduction and overview of logic programming, basic elements of prolog, application of logic programming.

**UNIT - V**

**Functional Programming Languages:** Introduction, fundamentals of FPL, LISP, ML, Haskell, application of Functional Programming Languages and comparison of functional and imperative Languages.

Scripting Language: Pragmatics, Key Concepts, and Case Study: Python – Values and Types, Variables, Storage and Control, Bindings, and Scope, Procedural Abstraction, Data Abstraction, Separate Compilation, Module Library.

**SUGGESTED BOOKS**

**TEXT BOOKS**

T1. Concepts of Programming Languages by Robert. W. Sebesta 8/e, Pearson Education, 2008.

T2. Programming Language Design Concepts by D.A. Watt, Wiley Dreamtech, rp-2007.

**REFERENCE BOOKS**

R1. Programming Languages, 2nd Edition, A.B. Tucker, R.E. Noonan, TMH.

R2. Programming Languages by K.C. Louden, 2nd Edition, Thomson, 2003.

R3. LISP, Patric Henry Winston and Paul Horn, Pearson Education.

R4. Programming in Prolog, W.F. Clocksin & C.S. Mellish, 5th Edition, Springer.

R5. Programming Python, M. Lutz, 3rd Edition, O’Reilly, SPD, rp-2007.

R6. Core Python Programming, Chun, II Edition, Pearson Education, 2007.

R7. Guide to Programming with Python, Michael Dawson, Thomson, 2008

**SESSION PLAN:**

| **S. No** | **Topics** | **Sub-Topics** | **Lecture No.** | **Suggested Books (page no’s)** | **Remarks** |
| --- | --- | --- | --- | --- | --- |
| 1 | Course Overview | Unit Review, Tutorial Plan, Importance of Assignments, Textbooks Referred | L1 |  |  |
| 2 | Necessary Background | Concepts in programming Languages |  |  |
| UNIT-I (PRELIMINARY CONCEPTS) | | | | | |
| 3 | Reasons for studying concepts of programming languages | Reasons for studying concepts of programming languages | L2 | T1 Ch1(2-5)  R1 Ch1(1-10)  R2 Ch1 (1-4) |  |
| 4 | Programming domains | Scientific Applications  Business Applications  Artificial Intelligence  Systems Programming  Web software | L3 | T1 Ch1(5-8)  R1 Ch1(10)  R2 Ch1 (14-15) |  |
| 5 | Language Evaluation criteria | Readability  Writability  Reliability  Cost | L4 | T1 Ch1(8-20)  R1 Ch1 (11-23)  R2 Ch1 (4-17) (19-25)  R3 Ch1 (3-29)  R3 Ch2 (35-40) |  |
|
| 6 | Influences on Language design  Language Categories | Computer architecture  Programming Methodologies  Language Categories | L5 | T1 Ch1 (20-23)  R2 Ch1 (34-39)  T1 Ch (23-24) |  |
| 7 | Programming Paradigms | Imperative Programming  Object Oriented Programming  Functional programming  Logic Programming | L6 | T2 Ch1(13-19)  R2 Ch1 (25-28) |  |
| 8 | Programming Language Implementation | Compilation, pure Interprétation, hybrid implémentation, preprocessor | L7 | T1-Ch1 (25-32)  R2 Ch2 (53-65) |  |
| Virtual Machines |
| 9 | Programming environments | Programming environments |
| 10 | General problems of describing syntax and semantics | Language Recognizers  Language Generators | L8 | T1 Ch3 (115-17)  R1 Ch1 (25)  R1 Ch2 (33-40)  R2 Ch3 (69-88) |  |
| 11 | Formal Methods of Describing Syntax | BNF  EBNF for common programming languages features  Parse trees  Ambiguous grammars | L9  L10 | T1 Ch3 (117-130)  T2 Ch4 (83-92, 97-101)  R1 Ch2 (35-36)  R2 Ch3 (88-97)  R3 Ch3 (51-57) |  |
|
| 12 | Attribute grammars | Static Semantics  Basic Concepts  Attribute Grammars Defined | L11 | T1 Ch3(130-136)  R1 Ch2 (41, 44)  R2 Ch4 (128-130) |  |
| Intrinsic attributes  Examples of Attribute Grammars  Computing attribute values Evaluation |
| 13 | Denotational semantics and Axiomatic semantics for common programming language features | Operational semantics  Axiomatic semantics  Denotational semantics | L12 | T1 Ch3(136-154)  T2 Ch13 (585-611)  R1 Ch1 (39-45)  R2 Ch4(125-128) (130-137) |  |
|
| **UNIT- II (DATA TYPES)** | | | | | |
| 14 | Introduction, Primitive | Numeric types  Character types  Boolean types | L13 | T1 Ch6(234-243)  T2 Ch6(192-200)  R1 Ch3(111-120)  R2 Ch5(155-163,171-173,180-186)  R3 Ch5(117-119,121-132) |  |
| 15 | Character | Design Issues  Strings and their operations  String length options  Evaluation  Implementation of Character String types |
| 16 | User-Defined | Enumeration Types  Sub range Types  Implementation of User-defined Ordinal types | L14 | T1 Ch6(243-260)  T2 Ch6(206-211)  R1 Ch3(128-136)  R2 Ch5(173-180)  R2 ch6(209-219)  R3 Ch5(119-121) |  |
| 17 | Arrays | Design Issues  Arrays and Indexes  Subscript Bindings and Array categories  Array Initialization  Array operations  Rectangular and Jagged Arrays  Slices  Evaluation  Implementation of Array types | L15 | T1 Ch6(243-260)  T2 Ch6(206-211)  R1 Ch3(128-136)  R2 Ch5(173-180)  R2 ch6(209-219)  R3 Ch5(119-121) |  |
| 18 | Associative Arrays | Structure and Operations  Implementing Associative Arrays | L16 | T1 Ch6(260-268)  R2 ch6(219-226)  R3 Ch5(144-149) |  |
| 19 | Record | Definitions of records  References to Record fields  Operations on records  Evaluation  Implementation of Record Types |
| 20 | Union | Design Issues  Discriminated versus Free Unions  Ada union types  Evaluation  Implementation of Union types | L17 | T1 Ch6(268-284)  T2 Ch6(202-206)  T2 Ch6(212-215)  R1 Ch3(120-127)  R2 Ch5(186-189)  R3 Ch5(132-137) |  |
| 21 | Pointer and Reference Types | Design Issues  Pointer Operations  Pointer problems  Pointers in Ada  Pointers in C and C++  Pointers in Fortran95  Reference Types  Evaluation  Implementation of Pointer and Reference Types | L18 | T1 Ch6(268-284)  T2 Ch6(202-206)  T2 Ch6(212-215)  R1 Ch3(120-127)  R2 Ch5(186-189)  R3 Ch5(132-137) |  |
| 22 | Design and implementation uses related to these types | Design and implementation uses related to these types | L19 | T1 Ch6(268-284)  T2 Ch6(202-206)  T2 Ch6(212-215)  R1 Ch3(120-127)  R2 Ch5(186-189)  R3 Ch5(132-137) |  |
| 23 | Names | Design Issues  Name forms  Special words | L20 | T1 Ch5(190-195)  T2 ch5(167-171)  R1 Ch2 (25-27, 50-51)  R3 Ch4 (77-82)  R5Ch1(6-8) |  |
| 24 | Variables | Name  Address  Type  Value |
| 25 | Concept of binding | Binding of Attributes to Variables  Type Bindings  Storage Bindings and Lifetime | L21 | T1 Ch5(195-211),  T2 Ch6 (225-231),  R1 Ch2 (49-50)  R1 Ch3 (137-145)  R2 Ch5 (163-166)  R3 Ch5 (149-154) |  |
| 26 | Type Checking | Type Checking |
| 27 | Strong typing | Strong typing |
| 28 | Type compatibility | Type compatibility |
| 29 | Named constants | Named constants | L22 | T1 Ch5(222-224),  T2 Ch5 (159-165)  R3 Ch4 ( 86-89) |  |
| 30 | Variable Initialization | Variable Initialization |
| 31 | Arithmetic relational and Boolean Expressions | Operator Evaluation Order  Operand Evaluation Order  Relational Expressions  Boolean Expressions | L23 | T1 Ch7 (292-301, 306-308)  T2 Ch7 (262-269)  R1 Ch4 (174-179)  R2 Ch8 (294-304),  R3 Ch4(89-93) |  |
| 32 | Short circuit evaluation | Short circuit evaluation | L24 | T1 Ch7 (308-310) ,  R2 Ch8 (304-308) |  |
| 33 | Mixed mode assignment | Mixed mode assignment | T1 Ch7(313-314) |  |
| 34 | Assignment statements | Simple Assignments  Conditional Targets  Compound Assignment Operators  Unary Assignment Operators  Assignment as an Expression  List Assignments | T1 Ch7 (310-313) |  |
| 35 | Control structures-  Statement level, compound statements | Statement level control structures- compound statements | L25 | T1 Ch8 (321-346)  T2 Ch7 (270-282)  T1 Ch8 (321-346)  T2 Ch7 (270-282)  R1 Ch4 (179-186)  R2 Ch8 (310-323)  R3 Ch4(93-105) | GATE,  PGCET |
| 36 | Selection statements | Two-Way Selection statements  Multiple Selection Constructs | GATE,  PGCET |
| 37 | Iterative Statements | Counter controlled Loops  Logically controlled Loops  User located loop control Mechanisms  Iteration Based on Data Structures | L26 | T1 Ch8 (321-346)  T2 Ch7 (270-282)  R1 Ch4 (179-186)  R2 Ch8 (310-323)  R3 Ch4(93-105) | GATE,  PGCET |
| 38 | Unconditional Statements | Unconditional Branching | L27 | T1 Ch8 (321-346)  T2 Ch7 (270-282)  R1 Ch4 (179-186)  R2 Ch8 (310-323)  R3 Ch4(93-105 | GATE,  PGCET |
| 39 | Guarded commands | Guarded commands |
| **UNIT-III ( SUBPROGRAMS AND BLOCKS)** | | | | | |
| 40 | Fundamentals of subprograms | General Subprogram Characteristics  Basic Definitions  Parameters  Procedures and Functions | L28 | T1 Ch9 (354-360)  R1 Ch7 (354-356)  R2 Ch9 (345-357)  R3 Ch7 (199-202) | GATE,  PGCET |
| 41 | Static and dynamic Scope | Evaluation of Static Scoping  Dynamic Scope  Evaluation of Dynamic Scoping | L29 | T1 Ch5 (211-220),  R1 Ch2 (93-96),  R2 ch9 (363-366,396-407),  R3 Ch6(165-185) | GATE,  PGCET |
| 42 | Scope and Lifetime of variable | Scope and Lifetime |
| 43 | Local referencing environments | Local referencing environments |  | T1 Ch9(360-361)  T1 Ch9(361-362) | GATE,  PGCET |
| 44 | Parameters passing method | Semantics models of Parameter passing  Implementation models of Parameter passing  Parameter passing methods of major languages  Type-checking Parameters  Implementing Parameter-Passing Methods  Multidimensional array as parameters  Design considerations  Examples of Parameter Passing | L30  L31 | T1 Ch9 (363-381),  R1 Ch2 (97-105)  R2 Ch9 (374-380)  R3 Ch7 (202-207) | GATE,  PGCET |
| 45 | Overloaded subprograms | Overloaded subprograms | L32  L33 | T1 Ch9 (381-389)  R2 Ch9 (389-393)  R3 Ch7 (214-225)  R1 Ch5 (276-281)  R2 Ch6 (244-246) | GATE,  PGCET |
| 46 | Generic sub-programs | Generic subprograms in Ada  Generic functions in C++  Generic subprograms in other Languages |
| 47 | Parameters that are subprogram names | Parameters that are subprogram names |
| 48 | User defined overloaded operators co-routines | User defined overloaded operators co-routines | L34 | T1 Ch9(389-393),  R1 Ch4 (215)  R2 Ch11 (441-445) | GATE,  PGCET |
|  | |
| **UNIT-IV(ABSTRACT DATA TYPES )** | | | | | |
| 49 | Abstraction and encapsulation | The Concept of Abstraction | L35 | T1 Ch11(428-432, 448-453)  R2 Ch6 (234-244  R3 Ch8 (233-243, 247-256) |  |
| 50 | Introduction to data abstraction | Floating-Point as an Abstract Data type  User-Defined Abstract Data types  An example |
| 51 | Design issues | Design issues for Abstract Data types | L36 | T1 Ch11(432-445)  R2 Ch7 (264-272) |  |
| 52 | Language examples | Abstract Data types in Ada  Abstract Data types in C++  Abstract Data types in Java  Abstract Data types in C# |
| 53 | C++ parameterized ADT | C++ parameterized abstract data types | L37 | T1 Ch11 (445-447)  R3 Ch8 (256-262) |  |
| 54 | Object oriented programming in Smalltalk | General Characteristics  Type Checking and polymorphism  Inheritance  Evaluation of Smalltalk | L38 | T1 ch12(465-467)  T2 Ch10(446-452)  R1 Ch6(313-317) |  |
| 55 | Object oriented programming in C++ | General Characteristics  Inheritance  Dynamic binding  Evaluation | T1 ch12(467-477)  T2 Ch10(434-446)  R1 Ch6(302-308) |  |
| 56 | Object oriented programming in Java | General Characteristics  Inheritance  Dynamic binding  Evaluation | L39 | T1 ch12(477-479)  T2 Ch10(413-419)  R1 Ch6(317--327) |  |
| 57 | Object oriented programming in C# | General Characteristics  Inheritance  Dynamic binding  Evaluation | L40 | T1 ch12(479-481) |  |
| 58 | Object oriented programming in Ada95 | General Characteristics  Inheritance  Dynamic binding  Evaluation | T1 ch12(481-485)  R1 Ch6(308-311) |  |
| 59 | Subprogram level concurrency | Introduction,  Fundamental Concepts  Language Design for concurrency  Design Issues | L41 | T1 ch13 (496-503)  R3 Ch10 (287-291) |  |
| 60 | Semaphores | Introduction to semaphores  Cooperation synchronization  Competition synchronization  Evaluation | L42 | T1 Ch13 (503-508)  T2 Ch14 (643-648)  R1 Ch4 (220-222, 232-233)  R2 Ch11 (455-458)  R3 Ch10 (291-295) |  |
| 61 | Monitors | Introduction  Competition synchronization  Cooperation synchronization  Evaluation | T1 Ch13 (508-513)  T2 Ch14 (648-654)  R1 Ch4 (233-234)  R2 Ch11 (464-465)  R3 Ch10 (295-302) |  |
| 62 | Message passing | The concept of Synchronous Message passing  The Ada Synchronous Message passing Model  Cooperation synchronization  Competition synchronization  Task Termination  Priorities  Binary Semaphores  Protected Objects  Asynchronous Message passing | L43 | T1 Ch13 (513-525)  T2 Ch14 (654-660)  R3 Ch10 (302-306) |  |
| 63 | Java Thread’s | The Thread class  Priorities  Cooperation synchronization  Competition synchronization  Evaluation | L44 | T1 Ch13 (525-532)  T2 Ch14 (634-643)  R1 Ch6 (323-327)  R3 Ch10 (306-320) |  |
| 64 | C# Threads | Basic Thread Operations  Synchronizing Threads  Evaluation | T1 Ch13 (532-534) |  |
| 65 | Exceptions, exception Propagation | Introduction to Exception handling, Basic concepts  Design Issues  Continuation | L45 | T1 Ch14 (542-548)  R1 Ch4 (194-197, 202)  R3 Ch9 (269-273, 278-284) |  |
| 66 | Exception handler in Ada | Exception Handlers  Binding Exceptions to Handlers  Continuation  Other Design Choices  An Example  Evaluation | T1 Ch14 (548-554) |  |
| 67 | Exception handling in C++ | Exception Handlers  Binding Exception to Handler  Continuation  Other Design Choices | L46 | T1 Ch14 (554-558)  R1 Ch9 (197-200)  R2 Ch11 (437-441) |  |
| An Example  Evaluation |
| 68 | Exception handling in Java | Classes of Exception  Exception Handlers  Binding Exception to Handler  Continuation  Other Design Choices  An Example  The finally Clause  Evaluation |
| T1 Ch14 (558-566)  R1 Ch9 (200-201) |
| 69 | Introduction and overview of logic programming | Introduction to Logic Programming  An overview of logic programming | L47 | T1 Ch16 (618,624-625),  T2 Ch12(539-544)  R1 Ch8(377) |  |
| 70 | The basic elements of prolog | Terms  Fact Statements  Rule Statements  Goal Statements  The inference Process of Prolog  Simple arithmetic  List structures |  | T1 Ch16 (625-640)  T2 Ch12 (552-563)  R2 Ch8 (327-339)  R1 Ch8 (377-393) |  |
| 71 | Applications of logic programming | RDBMS  Expert Systems  Natural Language Processing | T1 Ch16 (646-647) |  |
| **UNIT-V** (**FUNCTIONAL PROGRAMMING LANGUAGES)** | | | | | |
| 72 | Introduction | Introduction functional programming language | L48 | T1 Ch15 (580-583)  T2 Ch11 (471-476, 520-523)  R3 Ch12(343-346) |  |
| 73 | Fundamentals of functional programming languages | Fundamentals of functional programming languages | T1 Ch15 (583-584)  T2 Ch11 (476-480)  R1 Ch7 (333-342) |  |
| 74 | LISP | Data Types and Structures  The First LISP Interpreter | L49 | T1 Ch15 (584-603)  T2 Ch11 (481-494)  R1 Ch7 (344 – 347)  R2 Ch6 (227-234)  R3 Ch12 (346-370)  R2 Ch10 (420-422) |  |
| 75 | ML | ML, | L50 | T1 Ch15 (604-607),  T2 Ch11 (494-507),  R1 Ch7 (352-363),  R2 Ch4 (138-139),  R2 Ch5 (166),  R2 Ch11 (437-440) |  |
| 76 | Haskell | Haskell | L51 | T1 Ch15 (607-611)  T2 Ch11 (512-520) |  |
| 77 | Applications of functional programming languages & comparison of functional & Imperative Languages | Applications of functional languages | L52 | T1 Ch15 (611-612)  R1 Ch8 (393-394) |  |
| A comparison of functional& Imperative Languages |
| 78 | Scripting language | Pragmatics  key concepts | L53 | T2 Ch10 (413-417) |  |
| 79 | Case study- Python | Values and types  Variables  Storage and control  Bindings and scope | L54 | T2 Ch10 (417-425) |  |
| Procedural abstraction  Data Abstraction  Separate compilation  Module Library | L55 |  |