**RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALAYA, BHOPAL**

**New Scheme Based On AICTE Flexible Curricula**

**Computer Science & Information Technology, V-Semester**

**Departmental Elective CSIT- 503 (C) Principles of Programming Languages**

## Course Objectives

- To introduce the major programming paradigms, and the principles and techniques involved in design and implementation of modern programming languages.
- To introduce notations to describe syntax and semantics of programming languages.
- To analyze and explain behavior of simple programs using concepts such as binding, scope, control structures, subprograms and parameter passing mechanisms.
- To introduce the concepts of concurrency control and exception handling

**UNIT-I**

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

**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, conceptof binding, type checking, strong typing, type compatibility, named constants, variable initialization, Sequence control with Expressions, Conditional Statements, Loops, Exception handling.

**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, design issues for functions overloaded operators, co routines.

**UNIT-IV**

Abstract Data types: Abstractions and encapsulation, introductions to data abstraction, Static and Stack-Based Storage management. heap based storage management. Garbage Collection. Object oriented programming in small talk, C++, Java, C#, PHP, Perl . Concurrency: Subprogram level concurrency, semaphores, monitors, massage passing, Java threads, C# threads

**UNIT–V**

Exception handling, Exceptions, exception Propagation, Exception handler in C++ and Java. Logic Programming Language : Introduction and overview of logic programming, basic elements of prolog, applicationof logic programming. Functional Programming Languages: Introduction, fundamentals. Introduction to 4GL.

**Reference Books:**

1. Sebesta,"Concept of Vprogramming Language", Pearson Edu.
2. Louden, "Programming Languages: Principles & Practices", Cengage Learning
3. Tucker, " Programming Languages: Principles and paradigms", Tata McGraw –Hill
4. Terrance W Pratt, "Programming Languages: Design and Implementation", Pearson Edu.
5. Cavlo Ghezzi& Mehdi Jazayeri" Programming Languages Concepts", Willey India
6 E Horowitz, "Programming Languages", 2nd Edition, Addison Wesley

**Suggested List of Experiments:**

1. Define a LISP function to compute sum of squares.
2. Define a LISP function to compute difference of squares.(if x > y return x2 -y2, otherwise y2 - x2 ).
3. Define aRecursive LISP function to solve Ackermann's Function.
4. Define a Recursive LISP function to compute factorial of a given number.
5. Define a Recursive LISP function which takes one argument as a list and returns last element of the ist. (Do not use last predicate).
6. Define a Recursive LISP function which takes one argument as a list and returns a list except last element of the list. (Do not use but last predicate).
7. Define a Recursive LISP function which takes one argument as a list and returnsreverse of the list. (Do not use reverse predicate).
8. Define a Recursive LISP function which takes two arguments first, an atom, second, a list, returns a list after.

**Course Outcomes:**

At the completion of the course, students will...

- Have the background for choosing appropriate programminglanguages for certain classes of programming problems
- Be able to program in an imperative (or procedural),an object-oriented, a functional, and a logical programming language
- Understand the significance of an implementation of a programming language in a compiler or interpreter
- Have the ability to learn new programming languages
- Have the capacity to express programming concepts and choose among alternative ways to express things
- Be able to design a new programming language
- Make good use of debuggers and related tools