INSTITUTE OF AERONAUTICAL ENGINEERING

(Autonomous)

Dundigal, Hyderabad - 500 043

INFORMATION TECHNOLOGY

COURSE DESCRIPTION FORM

Course Title	:	**PRINCIPLES OF PROGRAMMING LANGUAGES**			
Course Code	:	A40511			
Regulation/Academic Year	:	R15-JNTUH/2016-17			
Course Structure	:	Lectures	Tutorials	Practicals	Credits
		4	-	-	4
Course Coordinator	:	B.DHANALAXMI , Associate Professor, IT			

I. COURSE OVERVIEW:

The course addresses growing importance of Programming languages, their uses, and importance of using different programming tools. Course addresses various influences of language design and language implementation techniques like, compilers, interpreters. Describing the meanings of syntax and semantics, comparison of different semantics like, operational, denotational, axiomatic semantics and generating parse trees. This course also explains about different expressions and statements used in different programming languages. Comparison of functional programming and logic programming. Exceptions and exception handling procedures of different programming languages like, C#, C++, Java, Ada95. It also introduces the importance of scripting languages and features, data types of Python language.

II. PREREQUISITES:

Level	**Credits**	**Periods/Week**	**Prerequisites**
UG	4	5	Basic programming concepts, Object Oriented Programming concepts.

III. COURSE ASSESSMENT METHODS:

a) Marks Distribution

Session Marks	**University End Exam Marks**	**Total Marks**
There shall be 2 midterm examinations. Each midterm examination consists of subjective type and Objective type tests. The subjective test is for 10 marks, with duration of 1 hour. The objective type test is for 10 marks with duration of 20 minutes. It consists of 10 Multiple choice and 10 objective type questions, the student has to answer all the questions and	75	100

each carries half mark. First midterm examination shall be conducted for the first two and half units of syllabus and second midterm examination shall be conducted for the remaining portion. Five marks are earmarked for assignments. There shall be two and half units of syllabus and second midterm examination shall be conducted for the remaining portion. Five marks are earmarked for assignments. There shall be two assignments in every theory course. Marks shall be awarded considering the average of two assignments in each course.		

IV. EVALUATION SCHEME:

S.No	Component	Duration	Marks
1	I Mid Examination	80 Minutes	20
2	I Assignment	-	05
3	II Mid Examination	80 Minutes	20
4	II Assignment	-	05
5	External Examination	3 Hours	75

V. COURSE OBJECTIVES:

1. Define the structure and design principles of programming languages.
2. Describe the skills in analyzing and using the features of programming languages.
3. Summarize the difference between logic programming and functional programming languages.
4. Express the role of variable declarations in programming languages, in particular to binding, scope, and substitution of variables.
5. Demonstrate the use of scripting languages

VI. COURSE OUTCOMES:

Upon completion of this course, students will be able to:

1. List the procedures to express syntax and semantics in formal notation.
2. Apply suitable programming paradigm for the given application.
3. Understand the principles related in designing the programming languages.
4. Elaborates the features of attribute grammars and use expressions and statements in a program..
5. List out the concepts of object oriented programming and logic programming languages.
6. Recall the importance of semaphores, monitors, message passing and Java threads.
7. Apply scripting languages in web design.

VII. HOW PROGRAM OUTCOMES ARE ASSESSED:

Program Outcomes		Level	Proficiency assessed by
PO1	**Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.	**H**	Assignment, Exercises
PO2	**Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.	**S**	Exercises
PO3	**Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.	**S**	Exercises
PO4	**Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.	**N**	**……..**
PO5	**Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.	**S**	Design, Exercises
PO6	**The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.	**N**	**……..**
PO7	**Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.	**N**	**……..**
PO8	**Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.	**N**	**……..**

PO9	**Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.	**S**	Workshop
PO10	**Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.	**N**	……..
PO11	**Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.	**N**	……..
PO12	**Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.	**S**	Exams, Discussions

VIII. HOW PROGRAM SPECIFIC OUTCOMES ARE ASSESSED:

Program Specific Outcomes		**Level**	**Proficiency assessed by**
PSO1	**Professional Skills:** The ability to research, understand and implement computer programs in the areas related to algorithms, system software, multimedia, web design, big data analytics, and networking for efficient analysis and design of computer-based systems of varying complexity.	**S**	Lectures, Assignments
PSO2	**Software Engineering practices:** The ability to apply standard practices and strategies in software service management using open-ended programming environments with agility to deliver a quality service for business success.	H	Projects
PSO3	**Successful Career and Entrepreneurship:** The ability to employ modern computer languages, environments, and platforms in creating innovative career paths, to be an entrepreneur, and a zest for higher studies.	S	Guest Lectures

N - None **S - Supportive** **H - Highly Related**

IX. 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, 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, case study: Python- values and types, variables, storage and control, bindings and scope, procedural abstraction, data abstraction, separate compilation, module library.

Text books:

1. Concepts of Programming Languages Robert .W. Sebesta 9/e, Pearson Education.

2. Programming Language design concepts, D.A. Watt, Wiley dreamtech.

References:

1. Programming languages, 2nd edition, A.B.Tucker, R.E. Noonan, TMH.

2. Programming Languages, K.C.Louden, 2nd edition, Thomson, 2003.

3. LISP Patric Henry Winston and Paul Horn Pearson Education.

4. Programming in Prolog, W.F.Clocksin, &C.S.Melish, 5th Edition, Springer.

5. Programming in Python, M.Lutz, 3rd Edition,O'reilly,SPD,rp-2007.

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

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

X. COURSE PLAN:

The course plan is meant as a guideline. There may probably be changes.

Lecture No	Course Learning Outcomes	Topics to be covered	References
1-6	Reviews the concepts of programming languages.	Reasons for studying, concepts of programming languages, Programming domains, Language Evaluation Criteria, influences on Language design, Language categories.	T1 pg:1.2 T1 pg:1.5 T1 pg:1.7-1.19 T1 pg:1.22
7-8	List out various programming paradigms.	Programming Paradigms – Imperative, Object Oriented, functional Programming, and Logic Programming.	T1 pg:1.25
9-10	Recall the design issues of implementation.	Programming Language Implementation – Compilation and Virtual Machines,	T1 pg:2.2-2.22
11-19	Recall various programming environments.	Programming environments, General Problem of describing Syntax and Semantics, Formal methods of describing syntax - BNF, EBNF for common programming languages feature. Parse trees. Ambiguous grammars.	T1 pg:1.32 T1 pg:3.3 T1 pg:3.5 T1 pg:3.6 T1 pg:3.7
20-25	Elaborates the features of attribute grammars.	Attribute grammars, denotational semantics and axiomatic semantics for common programming language features.	T1 pg:3.20 T1 pg:3.27

26-39	List out various 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.	T1 pg:6.4 T1 pg:6.13 T1 pg:6.41 T1 pg:6.45 T1 pg:5.2-5.8 T1 pg:6.58
40-46	Use expressions and statements in a program.	Arithmetic relational and Boolean expressions, Short circuit evaluation mixed mode assignment, Assignment Statements, Control Structures – Statement Level, Compound Statements. Selection, Iteration, Unconditional Statements, guarded commands.	T1 pg:7.3-7.17 T1 pg:7.19 T1 pg:8.2 T1 pg:8.4 T1 pg:8.32
47-50	Tabulate different parameter passing techniques.	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.	T1 pg:9.2 T1 pg:9.13 T1pg:9.14 T1pg: 9.16
51-53	Differentiate sub-program and co-routines.	Overloaded sub-programs, generic sub-programs, parameters that are sub-program names, design issues for functions user defined overloaded operators, co routines.	T1pg: 9.39-9.41 T1pg: 9.37 T1pg: 9.50
54-57	List out the concepts of object oriented programming.	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.	T1pg: 10.3 T1pg: 10.7 T1pg: 10.24 T1pg: 11.13
58-60	Recall the importance of semaphores, monitors, message passing.	Subprogram level concurrency, semaphores, monitors, message passing, Java threads, C# threads, Exceptions, exception Propagation, Exception handler in Ada, C++ and Java.	T1pg: 12.6 T1pg: 12.17 T1pg: 13.2
61	Apply logic programming concepts.	Introduction and overview of logic programming, basic elements of prolog, application of logic	T1pg: 14.3

		programming.	
62	Use of functional programming languages.	Introduction, fundamentals of FPL, LISP, ML, Haskell.	T1pg:14.7
63	Tabulate functional and logic programming languages.	Application of Functional Programming Languages and comparison of functional and imperative Languages.	T1pg: 14.37
64-65	Apply scripting languages in web design.	Pragmatics, key concepts, case study: Python- values and types, variables. Storage and control, bindings and scope, procedural abstraction, data abstraction, separate compilation, module library.	T1pg: 16.5 T1pg: 16.29

XI MAPPING COURSE OBJECTIVES LEADING TO THE ACHIEVEMENT OF PROGRAM OUTCOMES:

Course Objectives	Program Outcomes												Program Specific Outcomes		
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
1	S								S						S
2		S	S										S		
3					S				S			S			
4	H													H	
5												S			
6		S													S
7			S		S								S		

S= Supportive H = Highly Related

XII MAPPING COURSE OUTCOMES LEADING TO THE ACHIEVEMENT OF PROGRAM OUTCOMES:

Course Outcomes	Program Outcomes												Program Specific Outcomes		
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
1	H														S
2		S	S						S				S		
3	S														
4	H													H	
5					S							S			
6		S													S
7					S							H			

S= Supportive H = Highly Related

Prepared by: B. Dhanalaxmi, Associate Professor, IT

HOD, IT