	National Institute of Technology Meghalaya An Institute of National Importance	**CURRICULUM**

Programme	**Bachelor of Technologyin Computer Science and Engineering**	Year of Regulation	**2019-20**
Department	**Computer Science and Engineering**	Semester	**IV**

Course Code	Course Name	Credit Structure				Marks Distribution			
		L	T	P	C	INT	MID	END	Total
CS 220	**Principles of Programming Languages**	**3**	**0**	**0**	**3**	**50**	**50**	**100**	**200**

Course Objectives	**To enable the students to learn about various constructs and their respective comparisons in different high-level languages so that he can choose a suitable programming language for solving a particular problem.**	Course Outcomes	CO1	**Able to understand the history of programming languages and introduce abstraction, the concept of different language paradigms, and an overview of language design criteria.**
	To develop the student's ability to understand the salient features in the landscape of programming languages.		CO2	**Avail to understand how the syntactic structure of a language can be precisely specified using context-free grammar rules in Backus-Naur form (BNF).**
	To provide the students to gain experience with these paradigms by using example programming languages.		CO3	**Able to understand the abstractions of the operations that occur during the translation and execution of programs.**
	To develop the student's ability to gain experience with these paradigms by using example programming languages.		CO4	**Able to understand the usage of data types in various languages.**
			CO5	**Able to understand the procedure activation and parameter passing; and exceptions and exception handling.**
			CO6	**Able to understand the concepts like abstract data types, subprograms, and will be able to apply them in a realistic manner.**

No.	COs	Mapping with Program Outcomes (POs)												Mapping with PSOs		
		PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
1	CO1	2	0	0	0	0	1	2	0	1	1	0	0	2	1	1
2	CO2	2	3	1	1	0	2	1	0	3	2	1	2	1	2	2
3	CO3	3	2	1	0	2	3	0	1	0	1	3	1	3	2	2
4	CO4	1	0	3	2	0	2	1	0	3	2	1	0	1	2	2
5	CO5	2	0	1	0	2	3	1	0	1	2	1	0	3	2	3
6	CO6	1	2	0	3	1	2	0	2	0	1	0	0	2	3	2

SYLLABUS

No.	Content	Hours	COs
I	**Introduction:** **The Origins of Programming Languages, Abstractions in Programming Languages, Computational Paradigms, Language Definition, Language Translation, The Future of Programming Languages;**	**2**	**CO1**
II	**Language Design Criteria:** **Historical Overview, Efficiency, Regularity, Security, Extensibility, C++: An Object-Oriented Extension of C, Python: A General-Purpose Scripting Language;**	**2**	**CO1**
II	**Syntax and Analysis Parsing:** **Lexical Structure of Programming Languages, Context-Free Grammars and BNFs, Parse Trees and Abstract Syntax Trees, Ambiguity, Associativity, and Precedence, EBNFs and Syntax Diagrams, Parsing Techniques and Tools, Lexics vs. Syntax vs. Semantics, Case Study: Building a Syntax Analyzer for TinyAda;**	**6**	**CO2**
IV	**Basic Semantics:** **Attributes, Binding, and Semantic Functions, Declarations, Blocks, and Scope, The Symbol Table, Name Resolution and Overloading, Allocation, Lifetimes, and the Environment, Variables and Constants, Aliases, Dangling References, and Garbage, Case Study: Initial Static Semantic Analysis of TinyAda;**	**6**	**CO3**
V	**Data Types:** **Data Types and Type Information, Simple Types, Type Constructors, Type Nomenclature in Sample Languages, Type Equivalence, Type Checking, Type Conversion, Polymorphic Type Checking, Explicit Polymorphism, Case Study: Type Checking in TinyAda;**	**5**	**CO4**
VI	**Expressions and Statements:** **Expressions, Conditional Statements and Guards, Loops and Variations on WHILE, The GOTO Controversy and Loop Exits, Exception Handling, Case Study: Computing the Values of Static Expressions in TinyAda;**	**4**	**CO5**
VII	**Procedures and Environments:** **Procedure Definition and Activation, Procedure Semantics, Parameter-Passing Mechanisms, Procedure Environments, Activations, and Allocation, Dynamic Memory Management, Exception Handling and Environments, Case Study: Processing Parameter Modes in TinyAda;**	**5**	**CO5**
VIII	**Abstract Data Types and Modules:** **The Algebraic Specification of Abstract Data Types, Abstract Data Type Mechanisms and Modules, Separate Compilation in C, C++ Namespaces, and Java Packages, Ada Packages, Modules in ML, Modules in Earlier Languages, Problems with Abstract Data Type Mechanisms, The Mathematics of Abstract Data Types;**	**6**	**CO6**
	Total Hours	**36**	

Essential Readings

1. Louden KC. Programming languages: principles and practices. Cengage Learning; 2011.
2. Sebesta RW. Concepts of programming languages. Pearson Education India; 2016.
3. Sethi R, Sethi R. Programming languages: concepts and constructs. Reading: Addison-Wesley; 1996 Feb 2.

Supplementary Readings

1. Gabbrielli M, Martini S. Programming languages: principles and paradigms. Springer Science & Business Media; 2010.
2. Dowek G. Principles of programming languages. Springer Science & Business Media; 2009.
3. Kedar S, Thakare S. Principles of Programming Languages. Technical Publications; 2009.