NIRMA UNIVERSITY

Institute:	Institute of Technology
Name of Programme:	B.Tech. Computer Science and Engineering
Course Code:	2CSDE90
Course Title:	Formal Methods in Software Engineering
Course Type:	Departmental Elective
Year of Introduction:	2021-22

Credit Scheme

L	T	Practical Component				C
		LPW	PW	W	S	
2	0	2	-	-	-	3

Course Learning Outcomes (CLO):

At the end of the course, students will be able to –

1. explain the significance of formal methods in Software Engineering
2. infer formal specification languages based on propositional logic, predicate logic, relational calculus, and finite state machines
3. apply analysis techniques for formal specification languages with help of supporting tools
4. design formal specifications for software systems

Syllabus: **Total Teaching hours: 30**

Unit	Syllabus	Teaching hours
Unit-I	**Introduction:** Software development life cycle, Role of formal methods, Lightweight formal methods, Applications of formal methods, Classification of formal methods, Explicit vs. implicit models, executable vs. non-executable, Formal verification techniques, Levels in formal methods – Formal specification, Formal development and formal verification, Theorem provers **Stages in formal methods:** Specification, Development, Verification – Sign-off verification, Human-directed proof, Automated proof.	04
Unit-II	**Logic and Set Theory:** Propositional logic, Predicate logic, Sets and relations, Lambda calculus, Assertions, Declarations, Specifications and Code, Series and Sequence.	06
Unit-III	**Formal Events:** Proof of program correction, Application of Hoare logic to proof of algorithm correction, Programming approach by construction, Program refinement techniques, Verification of Sequential Programs	06

Unit-IV	**Formal Specification and Analysis**: Declarative modelling, Difference related to model checking, Alloy commands, Functions; predicates; facts; assertions and verifications (checks), Static vs. dynamic modelling, Simulation of an operation, Check safety properties. **Formal Development:** B method, Z notation, Event-B method	06
Unit-V	**Verification:** Verification of Sequential Programs, Static Analysis for Verifying Contracts, Symbolic Execution for Test Generation, Model-Based Test Generation, Explicit State Model Checking, Symbolic Model Checking, Model Checking Software: State bounds & Abstraction, Static Analysis of Concurrent Systems.	08

Self-Study: The self-study contents will be declared at the commencement of semester. Around 10% of the questions will be asked from self-study contents.

Suggested Readings/ References:

1. I. Van Horebeek & J. Lewi, Algebraic Specifications in Software Engineering
2. J. V. Guttag, E. Horowitz & D. R. Musser, The Design of Data Type Specifications, Current Trends in Programming Methodology (R. T. Yeh, ed.)
3. A. Diller, Z: An Introduction to Formal Methods ,Wiley
4. M. Huth and M. Ryan. Logic in Computer Science: Modeling and Reasoning about Systems. Cambridge University Press
5. Heitmeyer, C. and Mandrioli, Formal Methods for Real-Time Computing, Wiley
6. Hinchey, M.G., and Bowen, J.P., Application of Formal Methods, PH
7. Wordsworth, J.B., Software Engineering with B, Addison-Wesley
8. Daniel Jackson, Software Abstractions, MIT Press

Suggested List of Experiments:

Sr. No.	Practical Title	Hours
1	To explore tools for applying and testing predicate logic.	02
2	To explore the tools for writing Z notation and implement formal specifications.	02
3	To study and perform conceptual modeling of requirements using logic.	04
4	To study and perform algorithmic verification (model-checking) of design/models.	04
5	To study and verify functional correctness for abstract data types and refinement.	04

	6	To study and verify Hoare logic assertions, refinement of a program with respect to abstract data type specification.	04
	7	To explore tools for the software development process and implement a simulation.	02
	8	To perform verification of sequential programs.	04
	9	To study and perform white-box testing of a given application.	02
	10	To study and perform grey-box testing of a given application.	02
Suggested Case List:	-NA-		