CSCI 4265: Formal Methods in Software Engineering

2022 Winter Session	
Total Class Sessions: 25	**Instructor: Staff**
Class Sessions Per Week: 6	**Classroom: TBA**
Total Weeks: 4	**Office Hours: TBA**
Class Session Length (Minutes): 145	**Language: English**
Credit Hours: 4	

Course Description:

This course will enable students to study formal methods used for the analysis of software systems to ensure software's qualities, including formal methods for specifying, validating and verifying the software systems. Topics cover formal methods used in software architecture, cleanroom software engineering, formal methods in Agent- Oriented software, robotics and integrated formal methods, applying formal methods for software engineering, reconstructing software architecture, formal method techniques, and software quality Assurance

Prerequisite: Foundations of Computer Science, Data Structures and Algorithms, Computer System Organization.

Learning objectives:

1. Understand the basic formal methods used in software engineering
2. Use formal techniques for software architecture
3. Learn in depth of formal methods in Intelligent System
4. Use formal techniques in reconstructing the Software Architecture
5. Develop basic understanding of algebraic formal methods, and advanced formal method.
6. Learn Alloy Analyzer.

Course Materials:

1. Logic in Computer Science by Michael Huth and Mark Ryan, 2nd ed., ISBN 0-521-54310- X
2. Software Reliability Methods by Doron Peled, ISBN: 0-387-95106-7
3. Discrete Structures, Logic, and Computability by James Hein, ISBN: 0-7637-1843-2
4.Model Checking by Edmund M. Clarke, Jr., Orna Grumberg, and Doron A. Peled, ISBN: 0-262-03270-8
5. Principles of the Spin Model Checker by Mordechai Ben-Ari, ISBN: 978-1-84628-769-5
6. Software Abstractions by Daniel Jackson, ISBN: 0-262-10114-9
7. Mathematical Theory of Computation by Zohar Manna, ISBN: 0-486-43238-6
8. Systems and Software Verification by Beatrice Berard, Michel Bidoit et al., ISBN: 3-540-

41523-8

Course Format and Requirements:

The course will take place in a computer lab and the course format including lecture, programming project, and in-class discussion.
The specific topics that will be covered in the classes are listed in the course syllabus. The class period will consist of an active learning environment. During a majority of the class time, students will be actively working on problems under the instructor's guides.
Attendance:
Attendance will not be taken but is strongly recommended. Each student will have three allowed absences and no grade deduction will be made for the first three absences. More than three unexcused absences will result in an automatic reduction in your participation grade, for instance from A- to B+. Your active participation in the class is expected and encouraged.

Course Assignments:

Quizzes: There will be 5 quizzes this semester, given during the discussion sections. Each quiz will be on the material covered that week. There will be NO make-ups for quizzes for any reason. All of the quizzes will be closed book.
Midterm Exam: The in-class, close-book and non-cumulative midterm exam will be given through this course. The midterm exam will be based on the knowledge covered in class. No excuse will be accepted if students do not have legitimate excuses for absence. Physician Statement is required for missing the exam due.
Weekly Projects : There will be four hands-on projects based on course need. It will count for 40% of your grade for the course. The projects will enrich students' knowledge on writing large programs. The score will be given based on the correctness of the program.
Final Exam: The final will be in-class, cumulative and close-book. The final exams will be based on concepts covered in class. Note that the final will not be taken during the normal class times. Exact time and location for final will be announced later.

Course Assessment:

Quizzes	15%
Weekly Projects	35%
Midterm Exams	20%

Final Exam	30%
Total	**100%**

Grading Scale (percentage):

A+	A	A-	B+	B	B-	C+	C	C-	D+	D	D-	F
98-100	**93-97**	**90-92**	**88-89**	**83-87**	**80-82**	**78-79**	**73-77**	**70-72**	**68-69**	**63-67**	**60-62**	**<60**

Academic Integrity:

Students are encouraged to study together, and to discuss lecture topics with one another, but all other work should be completed independently.

Students are expected to adhere to the standards of academic honesty and integrity that are described in the Chengdu University of Technology's *Academic Conduct Code.* Any work suspected of violating the standards of the *Academic Conduct Code* will be reported to the Dean's Office. Penalties for violating the *Academic Conduct Code* may include dismissal from the program. All students have an individual responsibility to know and understand the provisions of the *Academic Conduct Code*.

Special Needs or Assistance:

Please contact the Administrative Office immediately if you have a learning disability, a medical issue, or any other type of problem that prevents professors from seeing you have learned the course material. Our goal is to help you learn, not to penalize you for issues which mask your learning.

Course Schedule:

Module	Topic	Activity
1	**Go through syllabus and introduction to the course** **Introduction to Formal Methods** ➢ Testing and Quality Assurance ➢ Comparative Formal Methods ➢ Specification techniques and formal specification ➢ Software Integration and Documenting ➢ Formal Semantic	*Quiz 1*
2	**Cleanroom Software Engineering**	*Quiz 2*

	➢ Verification and Validation ➢ Formal Methods in Agent-Oriented Software ➢ Formal Methods Model Checking and Testing ➢ Describing Syntax and Semantics ➢ Advanced Formal Methods	*Weekly project*
3	**Robotics and Integrated formal Methods** ➢ Formal Methods of Development ➢ Formal Methods for Requirement Engineering ➢ Formal Methods in Software Engineering ➢ Algebric Formal Methods ➢ Formal Method Diffusion ➢ Creating a Skeletal Structure	*Quiz 3* *Weekly project* *Midterm Exam*
4	**Reconstructing the Software Architecture** ➢ Re-Engineering and Reuses of Software ➢ Formal Methods: Requirements Techniques ➢ Applying Formal Method for Software ➢ Vulnerabilities detection ➢ Applying formal Methods for Software ➢ Formal Methods: A Road map	*Quiz 4* *Weekly project*
5	**Software Quality Assurance** ➢ Program verification ➢ Symbolic Evaluation ➢ Alloy Analyzer ➢ Behavior Trees Syntax and Semantics ➢ Final Overview Review for Final Exam	*Quiz 5* *Weekly Project* *Final Exam (Cumulative): TBA*