Course Specification

(Bachelor)

Course Title:	Computational Structures (2)
Course Code:	SE3402
Program:	BSc in Software Engineering
Department:	Software Engineering
College:	College of Computing
Institution:	Umm Al Qura University
Version:	1.0
Last Revision Date:	22/04/2025

Table of Contents

A. General information about the course:

1. Course Identification

1. Credit hours: (3)					
2. Course type					
A.	☐University	☐College	☒ Department	☐Track	☐Others
B.	☒ Required		☐Elective		
3. Level/year at which this course is offered: (2nd year/ 4th level)					
4. Course General Description:					
Builds upon the foundational concepts introduced in Computational Structures 1, focusing on advanced mathematical and computational techniques essential for solving complex problems in software engineering. This course introduces students to advanced graph theory, formal languages, automata theory, and computational complexity, emphasizing their practical applications in algorithm design, system modeling, and software development. Students will explore topics such as tree structures, graph algorithms, finite automata, regular expressions, and matrix operations, applying these concepts to real-world problems like network optimization, compiler design, and computational graphics. The course also covers computational complexity and algorithm analysis, equipping students with tools to evaluate and optimize software solutions.					
5. Pre-requirements for this course (if any)**:**					
SE1401 - Computational Structures (1)					
6. Co-requisites for this course (if any)**:**					
N/A					
7. Course Main Objective(s):					
By the end of Computational Structures 2, students will: 1. Analyze and solve problems using advanced graph theory and tree algorithms. 2. Understand formal languages and automata theory as a basis for computational systems. 3. Use computational structures to optimize software engineering solutions.					

2. Teaching mode (mark all that apply)

No	Mode of Instruction	Contact Hours	Percentage
1	Traditional classroom	60	100%
2	E-learning	0	0
3	Hybrid • Traditional classroom • E-learning	0	0
4	Distance learning	0	0

3. Contact Hours (based on the academic semester)

No	Activity	Contact Hours
1.	**Lectures**	30
2.	**Laboratory/Studio**	30
3.	**Field**	0
4.	**Tutorial**	0
5.	**Others (specify)**	0
Total		60

B. Course Learning Outcomes (CLOs), Teaching Strategies and Assessment Methods

Code	Course Learning Outcomes	Code of PLOs aligned with the program	Teaching Strategies	Assessment Methods
1.0	**Knowledge and understanding**			
1.1	- Understand advanced graph theory concepts, including traversal algorithms and shortest path algorithms.	K1	Lecture, Tutorials	Quizz, Homework , Exam
1.2	- Explain the basics of automata theory, formal languages, and their applications in computational systems.	K2	Lecture, Tutorials	Quizz, Homework , Exam

Code	Course Learning Outcomes	Code of PLOs aligned with the program	Teaching Strategies	Assessment Methods
1.3	- Understand the principles of computational complexity and algorithm efficiency.	K1	Lecture, Tutorials	Quizz, Homework , Exam
1.4	- Describe the structure and operations of trees, including binary search trees and balanced trees.	K2	Lecture, Tutorials	Quizz, Homework , Exam
2.0	**Skills**			
2.1	- Implement graph and tree algorithms to solve real-world computational problems.	S1	Lecture, Tutorials	Quizz, Homework , Exam
2.2	- Design deterministic and non-deterministic finite automata for simple language recognition tasks.	S2	Lecture, Tutorials	Quizz, Homework , Exam
2.3	- Use regular expressions to validate strings and perform pattern matching.	S2	Lecture, Tutorials	Quizz, Homework , Exam
3.0	**Values, autonomy, and responsibility**			
3.1	- Foster an appreciation for the theoretical underpinnings of modern software engineering techniques.	V1	Lecture, Tutorials	Quizz, Homework , Exam
3.2	- Develop a habit of evaluating and optimizing computational solutions.	V2	Lecture, Tutorials	Quizz, Homework , Exam

C. Course Content

No	List of Topics	Contact Hours
1.	Advanced Graph Theory Connectivity and graph traversal algorithms: Depth-First Search (DFS), Breadth-First Search (BFS) Graph representations: Adjacency matrices and adjacency lists Applications: Shortest path algorithms (Dijkstra, Floyd-Warshall) Minimum spanning trees (Prim, Kruskal)	**8**

2.	Trees and Tree Algorithms Basics: Properties of trees Binary trees and traversals (in-order, pre-order, post-order) Advanced Trees: Binary Search Trees (BST) Balanced trees (AVL trees, Red-Black trees) Applications: Data storage and retrieval Syntax trees in compilers	**8**
3.	Formal Languages and Automata Formal Languages: Alphabets, strings, and languages Operations on languages (union, concatenation, closure) Finite Automata: Deterministic Finite Automata (DFA) Non-Deterministic Finite Automata (NFA) Equivalence of DFA and NFA Applications: Lexical analysis in compilers String matching algorithms	**8**
4.	Regular Expressions and Grammars Regular expressions: Syntax and operations Conversion to finite automata Context-free grammars (CFG): Parse trees Ambiguity in grammars Applications: Validation of inputs Compiler design	**8**
5.	Matrices and Linear Algebra in Computation Basics: Matrix operations (addition, multiplication, transposition) Determinants and inverses Applications: Graph algorithms Transformations in computer graphics	**8**
6.	Computational Complexity Big-O, Big-Theta, and Big-Omega notation Growth of functions and asymptotic analysis Applications:	**4**

	Analyzing algorithm efficiency Problem categorization (P vs. NP)	
7.	Advanced Topics in Modular Arithmetic Applications in algorithms: Modular exponentiation Modular multiplicative inverse Cryptographic applications: Basic understanding of RSA and digital signatures	8
8.	Combinatorics Basic counting principles (addition and multiplication rules) Permutations and combinations Applications: Algorithm design Optimization problems	8
Total		**60**

D. Students Assessment Activities

No	Assessment Activities *	Assessment timing (in week no)	Percentage of Total Assessment Score
1.	**Assignments**	**2-14**	**10**
2.	**Projects**	**2-14**	**20**
3.	**Practicals**	**2-14**	**10**
4.	**Mid Term**	**7**	**20**
5.	**Final Exam**	**16-17**	**40**

*Assessment Activities (i.e., Written test, oral test, oral presentation, group project, essay, etc.).

E. Learning Resources and Facilities

1. References and Learning Resources

Essential References	• Rosen, K. H. (2019). *Discrete mathematics and its applications* (8th ed.). McGraw-Hill Education. ISBN 978-1260091991. • Gersting, J. L. (2014). *Mathematical structures for computer science: Discrete mathematics and its applications* (7th ed.). W.H. Freeman and Company, a Macmillan Higher Education Company. ISBN 978-1429215107.

Supportive References	• Houston, K. (2009). *How to think like a mathematician: A companion to undergraduate mathematics*. Cambridge University Press. ISBN 978-0521719780.
Electronic Materials	
Other Learning Materials	

2. Required Facilities and equipment

Items	Resources
facilities (Classrooms, laboratories, exhibition rooms, simulation rooms, etc.)	Traditional Classroom
Technology equipment (projector, smart board, software)	Multimedia Projector
Other equipment (depending on the nature of the specialty)	N/A

F. Assessment of Course Quality

Assessment Areas/Issues	Assessor	Assessment Methods
Effectiveness of teaching	Students	Direct, Indirect
Effectiveness of Students' assessment	Faculty, Peer reviewer	Direct, Indirect
Quality of learning resources	Faculty, Course coordinator	Direct, Indirect
The extent to which CLOs have been achieved	Course coordinator, Program management committee	Direct
Other		

Assessors (Students, Faculty, Program Leaders, Peer Reviewers, Others (specify)

Assessment Methods (Direct, Indirect)

G. Specification Approval

COUNCIL /COMMITTEE	**SOFTWARE ENGINEERING DEPARTMENT COUNCIL**
REFERENCE NO.	**THE 17TH MEETING FOR THE ACADEMIC YEAR 1446H**
DATE	**22/04/2025**