MODULE DESCRIPTION FORM

نموذج وصف المادة الدراسية

Module Information معلومات المادة الدراسية					
Module Title	**Discrete Mathematics / الرياضياتالمتقطعة**			**Module Delivery**	
Module Type	**Core**			☒ **Theory** ☒ **Lecture** ☐ **Lab** ☒ **Tutorial** ☐ **Practical** ☐ **Seminar**	
Module Code	CYBT205				
ECTS Credits	**4**				
SWL (hr/sem)	**100**				
Module Level			**Semester of Delivery**	3	
Administering Department			**College**		
Module Leader			**e-mail**		
Module Leader's Acad. Title			**Module Leader's Qualification**	Ph.D. in Mathematics / Computer Science	
Module Tutor			**e-mail**		
Peer Reviewer Name			**e-mail**		
Scientific Committee Approval Date	2/11/2026		**Version Number**	1.0	

Relation with other Modules العلاقة مع المواد الدراسية الأخرى			
Prerequisite module	General Mathematics /Engeeniring of Mathematics	**Semester**	1 & 2
Co-requisites module	Data Structures	**Semester**	4

Module Aims, Learning Outcomes and Indicative Contents أهداف المادة الدراسية ونتائج التعلم والمحتويات الإرشادية	
Module Aims أهداف المادة الدراسية	This module aims to introduce students to the fundamental mathematical structures and principles that form the foundation of computer science and software engineering. It focuses on propositional and predicate logic, proof techniques, mathematical induction, set theory, relations, functions, combinatorics, recurrence relations, graph theory, trees, and Boolean algebra. The course aims to develop rigorous mathematical reasoning, algorithmic thinking, and problem-solving skills required for algorithm analysis, data structure design, and theoretical computer science.
Module Learning Outcomes مخرجات التعلم للمادة الدراسية	Understand propositional logic, predicate logic, truth tables, logical equivalences, and formal proof methods. Master mathematical induction and strong induction techniques to prove algorithms and mathematical theorems. Comprehend set operations, relations (equivalence relations, partial orders), and properties of functions. Apply combinatorial counting principles, permutations, combinations, and the Pigeonhole Principle to computer science problems. Solve linear recurrence relations and analyze recursive algorithm complexities. Understand graph theory fundamentals (Eulerian/Hamiltonian graphs, shortest path algorithms, graph coloring) and tree structures. Analyze Boolean functions, Boolean algebra, and simplify logic expressions using Karnaugh maps.
Indicative Contents المحتويات الإرشادية	The module will cover the following key areas: Propositional Logic, Logical Connectives, Truth Tables, and Logical Equivalences Predicates, Quantifiers, and Rules of Inference Proof Methods: Direct Proofs, Proof by Contraposition, Proof by Contradiction, and Proof by Cases Mathematical Induction, Strong Induction, and Well-Ordering Principle Set Theory: Operations, Cartesian Products, Venn Diagrams, and Computer Representation of Sets Relations: Reflexive, Symmetric, Transitive Properties, Equivalence Relations, and Partial Orderings (Posets) Functions: Injective, Surjective, Bijective, Composition, and Inverse Functions Elementary Combinatorics: Counting Principles, Permutations, Combinations, and Pigeonhole Principle Recurrence Relations, Solving Homogeneous and Non-homogeneous Recurrences, and Master Theorem Graph Theory: Terminology, Graph Representations, Paths, Cycles, Euler/Hamilton Paths, and Dijkstra's Shortest Path Trees: Tree Traversal, Spanning Trees, Minimum Spanning Trees (Kruskal & Prim Algorithms) Boolean Algebra: Boolean Functions, Gate Logic, DNF/CNF, and Karnaugh Maps

Learning and Teaching Strategies استراتيجيات التعلم والتعليم	
Strategies	The module will employ a combination of strategies including: Lectures: To provide theoretical foundations and mathematical proofs on discrete structures. Group Discussions: To facilitate problem-solving and critical evaluation of logical arguments. Problem-Solving Sessions: To work through mathematical proofs, set operations, and combinatorial calculations. Practical Tutorials / Lab Sessions: Programming exercises in Python/C++ to implement logic gates, set algorithms, recursion, and graph traversals (BFS/DFS). Tutorials: To practice complex proof methods, recurrence relations, and graph algorithms. Mid-term and final exams: To assess comprehensive knowledge and analytical problem-solving skills. Preparatory Week: To give students time to revise and clarify any difficulties before the final exam.

Student Workload (SWL) الحمل الدراسي للطالب محسوب لـ ١٥ اسبوعا			
Structured SWL (h/sem) الحمل الدراسي المنتظم للطالب خلال الفصل	48	**Structured SWL (h/w)** الحمل الدراسي المنتظم للطالب أسبوعيا	4
Unstructured SWL (h/sem) الحمل الدراسي غير المنتظم للطالب خلال الفصل	52	**Unstructured SWL (h/w)** الحمل الدراسي غير المنتظم للطالب أسبوعيا	4
Total SWL (h/sem) الحمل الدراسي الكلي للطالب خلال الفصل	100		

Module Evaluation تقييم المادة الدراسية					
		Time/Number	**Weight (Marks)**	**Week Due**	**Relevant Learning Outcome**
Formative assessment	**Quizzes**	2	5% (5)	5, 10	LO #1, 3 and 5
	Assignments	2	5% (5)	2, 12	LO #2, 4 and 6
	Projects / Lab.	1	10% (10)	Continuous	LO #7
	Report	1	10% (10)	13	LO #3, 5 and 7
	Midterm Exam	2 hr	20% (20)	8	LO #1 - 4

Summative assessment	**Final Exam**	2hr	50% (50)	16	All
Total assessment			100% (100 Marks)		

Delivery Plan (Weekly Syllabus) المنهاج الاسبوعي النظري	
	Material Covered
Week 1	Introduction to Discrete Mathematics: Propositional Logic, Truth Tables, and Logical Connectives.
Week 2	Logical Equivalences, De Morgan's Laws, Tautologies, and Contradictions.
Week 3	Predicates, Quantifiers (Universal & Existential), and Nested Quantifiers.
Week 4	Rules of Inference and Introduction to Formal Proof Methods.
Week 5	Proof Techniques: Direct Proof, Proof by Contraposition, Proof by Contradiction, and Proof by Cases.
Week 6	Mathematical Induction: Principles, Base Cases, Inductive Steps, and Applications.
Week 7	Strong Induction, Structural Induction, and Well-Ordering Property.
Week 8	Mid-term Exam (الفصل منتصف اختبار) & Set Theory: Operations, Power Sets, Cartesian Products, Venn Diagrams.
Week 9	Relations: Matrix & Graph Representations, Equivalence Relations, and Equivalence Classes.
Week 10	Partial Orderings (Posets), Hasse Diagrams, and Lattices.
Week 11	Functions: One-to-One, Onto, Bijections, Inverse Functions, and Function Composition.
Week 12	Elementary Combinatorics: Sum & Product Rules, Permutations, Combinations, and Pigeonhole Principle.
Week 13	Recurrence Relations: Formulating Recurrences, Solving First-Order and Linear Second-Order Homogeneous Recurrences.
Week 14	Introduction to Graph Theory: Terminology, Isomorphism, Connectedness, Euler and Hamiltonian Paths.
Week 15	Trees and Spanning Trees: Tree Traversal (Inorder, Preorder, Postorder), Prim's and Kruskal's MST Algorithms.
Week 16	**Boolean Algebra: Boolean Functions, DNF/CNF, Logic Gates, and Minimization using Karnaugh Maps / Final Exam.**

Learning and Teaching Resources مصادر التعلم والتدريس		
	Text	**Available in the Library?**
Required Texts	Rosen, K. H. (2019). "Discrete Mathematics and Its Applications" (8th ed.). McGraw-Hill Education.	Yes
Recommended Texts	Epp, S. S. (2020). "Discrete Mathematics with Applications" (5th ed.). Cengage Learning.	Yes
Websites	Yes	

Grading Scheme مخطط الدرجات				
Group	**Grade**	التقدير	**Marks (%)**	**Definition**
Success Group (50 - 100)	**A** - Excellent	امتياز	90 - 100	Outstanding Performance
	B - Very Good	جيد جدا	80 - 89	Above average with some errors
	C - Good	جيد	70 - 79	Sound work with notable errors
	D - Satisfactory	متوسط	60 - 69	Fair but with major shortcomings
	E - Sufficient	مقبول	50 - 59	Work meets minimum criteria
Fail Group (0 – 49)	**FX** – Fail	راسب (قيد المعالجة)	(45-49)	More work required but credit awarded
	F – Fail	راسب	(0-44)	Considerable amount of work required

Note: Marks Decimal places above or below 0.5 will be rounded to the higher or lower full mark (for example a mark of 54.5 will be rounded to 55, whereas a mark of 54.4 will be rounded to 54. The University has a policy NOT to condone "near-pass fails" so the only adjustment to marks awarded by the original marker(s) will be the automatic rounding outlined above.