University of Fallujah
College of Information Technology
Department of Cybersecurity

MODULE DESCRIPTION FORM

نموذج وصف المادة الدراسية

Module Information معلومات المادة الدراسية					
Module Title	**Analysis & Design of Algorithms**			**Module Delivery**	
Module Type	**Core**			☒ **Theory** ☒ **Lecture** ☒ **Lab** ☐ **Tutorial** ☐ **Practical** ☐ **Seminar**	
Module Code	**CYS225**				
ECTS Credits	**5**				
SWL (hr/sem)	**125**				
Module Level	2		**Semester of Delivery**	2	
Administering Department	Dept. of Cybersecurity		**College**	College of Information Technology	
Module Leader	Dr. Ahmad AlSabhany		**e-mail**	Ahmad.alsabhany@uofallujah.edu.iq	
Module Leader's Acad. Title	Asst. Prof.		**Module Leader's Qualification**	Ph.D.	
Module Tutor	Dr. Mohammed Abduljawad M. Al-Shibly		**e-mail**	mohammed.abduljawad@uofallujah.edu.iq	
Peer Reviewer Name	Dr. Omar Salah Shareef		**e-mail**	omar.alshareef@uofallujah.edu.iq	
Scientific Committee Approval Date	25-5-2026		**Version Number**	1.0	

Relation with other Modules العلاقة مع المواد الدراسية الأخرى			
Prerequisite module	None	**Semester**	
Co-requisites module	None	**Semester**	

Module Aims, Learning Outcomes and Indicative Contents أهداف المادة الدراسية ونتائج التعلم والمحتويات الإرشادية	
Module Objectives **أهداف المادة الدراسية**	The module builds upon the programming knowledge and skills acquired in Structured Programming I and II. 1. Develop a solid understanding of algorithm design and analysis principles. 2. Analyze the efficiency of algorithms using time and space complexity measures. 3. Apply algorithmic techniques to solve computational problems systematically. 4. Design and implement searching, sorting, and graph-based algorithms. 5. Compare alternative algorithmic solutions and evaluate their performance. 6. Apply recursive, greedy, and dynamic programming approaches to problem solving. 7. Develop practical skills in implementing, testing, and optimizing algorithms. 8. Prepare students for advanced study in software development, artificial intelligence, cybersecurity, and related computing fields.
Module Learning Outcomes **مخرجات التعلم للمادة الدراسية**	1. Explain the fundamental principles of algorithm design and analysis. 2. Analyze algorithm performance using time and space complexity measures. 3. Design and implement algorithms to solve computational problems. 4. Apply searching, sorting, and graph algorithms to practical applications. 5. Utilize recursive, greedy, and dynamic programming techniques in problem solving. 6. Compare alternative algorithms and evaluate their efficiency and suitability for specific problems. 7. Implement, test, and optimize algorithms using appropriate programming techniques. 8. Apply algorithmic thinking to software development, cybersecurity, and data processing problems.
Indicative Contents **المحتويات الإرشادية**	• Introduction to Algorithm Analysis and Design • Mathematical Foundations for Algorithm Analysis • Time and Space Complexity • Asymptotic Notation (Big O, Big Ω, Big Θ)

	• Best, Average, and Worst-Case Analysis • Recursive Algorithms and Recurrence Relations • Divide-and-Conquer Algorithms • Searching Algorithms • Sorting Algorithms • Algorithm Correctness and Performance Evaluation • Greedy Algorithm Design Techniques • Dynamic Programming • Backtracking Algorithms • Graph Representations and Traversal Algorithms • Shortest Path Algorithms • Minimum Spanning Tree Algorithms • Algorithm Optimization Techniques • Comparative Analysis of Algorithms • Applications of Algorithms in Software Development and Cybersecurity

Learning and Teaching Strategies استراتيجيات التعلم والتعليم	
Strategies	**The module is delivered through lectures, laboratory sessions, problem-solving activities, and programming exercises. Students develop algorithmic thinking through the design, implementation, analysis, and comparison of algorithms for a variety of computational problems. Real-world examples and case studies are used to demonstrate the practical application of algorithms in software development, data processing, cryptography, and cybersecurity. Laboratory activities emphasize algorithm implementation, performance evaluation, optimization techniques, and the selection of appropriate algorithms for specific problem domains.**

Student Workload (SWL) الحمل الدراسي للطالب محسوب لـ ١٥ اسبوعا			
Structured SWL (h/sem) الحمل الدراسي المنتظم للطالب خلال الفصل	78	**Structured SWL (h/w)** الحمل الدراسي المنتظم للطالب أسبوعيا	4
Unstructured SWL (h/sem) الحمل الدراسي غير المنتظم للطالب خلال الفصل	47	**Unstructured SWL (h/w)** الحمل الدراسي غير المنتظم للطالب أسبوعيا	4
Total SWL (h/sem) الحمل الدراسي الكلي للطالب خلال الفصل	**125**		

Module Evaluation
تقييم المادة الدراسية

		Time/Number	Weight (Marks)	Week Due	Relevant Learning Outcome
Formative assessment	Assignments	2	10% (10)	5 and 10	All
	Quiz	1	10% (10)	4	LO #1 - #3
	Practical (Lab)	1	10% (10)	9	LO #1 - #4
	Quiz (Lab)	1	10% (10)	13	All
Summative assessment	Midterm Exam	2hr	10% (10)	7	LO #5 - #8
	Final Exam	3hr	50% (50)	16	All
Total assessment			100% (100 Marks)		

Delivery Plan (Weekly Syllabus)
المنهاج الاسبوعي النظري

	Material Covered
Week 1	Introduction to Algorithms and Complexity Analysis
Week 2	Asymptotic Notation and Performance Evaluation
Week 3	Recursive Algorithms and Recurrence Relations
Week 4	Divide-and-Conquer Algorithms
Week 5	Searching Algorithms
Week 6	Sorting Algorithms
Week 7	Comparative Analysis of Searching and Sorting Algorithms
Week 8	Midterm Examination
Week 9	Greedy Algorithm Design Techniques
Week 10	Dynamic Programming
Week 11	Backtracking Algorithms
Week 12	Graph Representations and Traversal Algorithms
Week 13	Shortest Path Algorithms
Week 14	Minimum Spanning Tree Algorithms and Graph Optimization
Week 15	Algorithm Optimization, Applications, and Course Review
Week 16	Final Exam Preparation

Delivery Plan (Weekly Lab. Syllabus)
المنهاج الاسبوعي للمختبر

	Material Covered
Week 1	**Algorithm Analysis and Execution Time Measurement**
Week 2	**Complexity Analysis and Performance Comparison**
Week 3	**Implementing Recursive Algorithms**
Week 4	**Divide-and-Conquer Algorithms**
Week 5	**Linear Search and Binary Search Implementation**
Week 6	**Sorting Algorithms: Bubble, Selection, and Insertion Sort**
Week 7	**Sorting Algorithms: Merge Sort and Quick Sort**
Week 8	**Midterm Practical Assessment**
Week 9	**Greedy Algorithms: Activity Selection and Related Problems**
Week 10	**Dynamic Programming: Fibonacci and Knapsack Problems**
Week 11	**Backtracking Algorithms: N-Queens and Related Problems**
Week 12	**Graph Representation, BFS, and DFS Implementation**
Week 13	**Shortest Path Algorithms: Dijkstra's Algorithm**
Week 14	**Minimum Spanning Tree Algorithms: Prim's and Kruskal's Algorithms**
Week 15	**Algorithm Comparison, Optimization, and Project Presentation**

Learning and Teaching Resources **مصادر التعلم والتدريس**		
	Text	**Available in the Library?**
Required Texts	1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). Introduction to Algorithms (4th ed.). MIT Press.	Yes
Recommended Texts	1. Skiena, S. S. (2020). The Algorithm Design Manual (3rd ed.). Springer.	Yes
Websites	1. https://visualgo.net/en 2. https://www.geeksforgeeks.org/fundamentals-of-algorithms/	

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	**FX** – Fail	**راسب (قيد المعالجة)**	(45-49)	More work required but credit awarded

(0 – 49)	**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.