



MODULE DESCRIPTION FORM



نموذج وصف المادة الدراسية

م.م. رشيد صبيح نوري
٢٠١٩/٠٤/٠٤

Module Information			
معلومات المادة الدراسية			
Module Title	Programming Fundamentals I		Module Delivery
Module Type	Core		<input checked="" type="checkbox"/> Theory
Module Code	IT104		<input type="checkbox"/> Lecture
ECTS Credits	7		<input checked="" type="checkbox"/> Lab
SWL (hr/sem)	175		<input checked="" type="checkbox"/> Tutorial
			<input type="checkbox"/> Practical
			<input type="checkbox"/> Seminar
Module Level	1	Semester of Delivery	1
Administering Department	UGI	College	College of Science
Module Leader	Mohsen Hassan Hosein	e-mail	mohsin.ha@uowa.edu.iq
Module Leader's Acad. Title	Assistant Professor	Module Leader's Qualification	Ph.D.
Module Tutor	-	e-mail	-
Peer Reviewer Name	-	e-mail	-
Scientific Committee Approval Date	-	Version Number	V1

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

<p>Module Objectives أهداف المادة الدراسية</p>	<p>The following are some key aims and benefits of studying Programming Fundamentals I:</p> <ol style="list-style-type: none"> 1. Introduction to Programming: Introduce students to the fundamental concepts of programming, including the role of programming languages, the software development process, and basic programming principles. 2. Problem Solving: Teach students how to analyze problems and develop algorithms to solve them. Emphasize problem-solving techniques, algorithm design, and decomposition of complex problems into smaller, manageable parts. 3. Input and Output: Teach students how to interact with the user and handle standard input/output operations, including reading from keyboard and display to screen. 4. Programming Language Basics: Familiarize students with the syntax, semantics, and basic constructs of a programming language, such as variables, data types, control structures (loops, conditionals), and functions. 5. Debugging and Testing: Teach students how to debug and test their programs to identify and fix errors. Explore techniques for error detection, debugging tools, and strategies for writing effective test cases
<p>Module Learning Outcomes مخرجات التعلم للمادة الدراسية</p>	<p>The following are some common learning outcomes for a Programming Fundamentals I:</p> <ol style="list-style-type: none"> 1. Knowledge of Programming Concepts: Demonstrate a solid understanding of fundamental programming concepts, including variables, data types, control structures, and basic algorithms. 2. Problem Solving Skills: Apply problem-solving techniques to analyze and solve programming problems by decomposing them into smaller, manageable parts and designing appropriate algorithms. 3. Proficiency in Programming Language: Develop proficiency in using a specific programming language covered in the course, including understanding the language's syntax, semantics, and basic constructs. 4. Effective Code Writing: Write clear, well-structured, and readable code that follows coding standards and best practices, including proper indentation, meaningful variable names, and appropriate comments. 5. Debugging and Testing Skills: Use debugging techniques and tools to identify and fix errors in programs. Develop effective test cases and perform testing to ensure program correctness and reliability.
<p>Indicative Contents المحتويات الإرشادية</p>	<p>The indicative contents of a Programming Fundamentals I module have a list of common topics that shown below:</p> <ol style="list-style-type: none"> 1-Introduction to Programming: Role of programming languages, Software development process, Basic programming principles and concepts. [15 hrs.] 2-Problem Solving and Algorithm Design: Problem analysis and requirements specification, Algorithm design techniques (e.g., topdown design, stepwise refinement), Flowcharts and pseudocode. [20hrs] 3-Input and Output: standard input/output operations, including reading from keyboard and display to screen. [10 hrs.]

	<p>4- Programming Language Basics: Variables and data types, Operators and expressions, Control structures (loops, conditionals). [30 hrs.]</p> <p>5- Modular Programming: Scope and lifetime of variables. [10 hrs.]</p> <p>6-Debugging and Testing: Common types of programming errors, Debugging techniques and tools. [10 hrs.]</p>
--	--

Learning and Teaching Strategies
استراتيجيات التعلم والتعليم

Strategies	<p>To teach a Programming Fundamentals I module, various strategies can be employed to facilitate effective learning and engagement. Here are some learning and teaching strategies commonly used in Programming Fundamentals I module:</p> <ol style="list-style-type: none"> 1- Lectures: Delivering lectures to present theoretical concepts, principles, and foundational knowledge of Programming Fundamentals I. Lectures can include visual aids, examples, and demonstrations to enhance understanding. 2- Interactive Discussions: Encourage students to actively participate in discussions by asking questions, sharing their thoughts, and engaging in peer-to-peer learning. Discussions can focus on challenging concepts, real-world applications, or case studies related to Programming Fundamentals I. 3- Hands-on Lab Sessions: Conduct practical lab sessions where students can gain hands-on experience with Programming Fundamentals I, 4 commands, and programming exercises. These sessions provide an opportunity to reinforce theoretical concepts and develop practical skills. 4- Group Projects: Assign group projects that involve designing, implementing, and evaluating components of Programming Fundamentals I. Group projects promote teamwork, problem-solving, and practical application of operating system concepts. 5- Online Resources and Tutorials: Provide access to online resources, tutorials, and interactive learning materials related to Programming Fundamentals I. This allows students to explore additional content, reinforce their understanding, and self-assess their progress. 6- Assessments and Feedback: Use a variety of assessment methods such as quizzes, assignments, projects, and exams to evaluate students' understanding of Programming Fundamentals I concepts. Provide timely and constructive feedback to help students improve their knowledge and skills.
-------------------	---




 أ.م.د. منى مصطفى نوزل
 ٢٠٢٤/٠٤/٠٤

Student Workload (SWL)

الحمل الدراسي للطالب محسوب لـ ١٥ اسبوعا

Structured SWL (h/sem) الحمل الدراسي المنتظم للطالب خلال الفصل		Structured SWL (h/w) الحمل الدراسي المنتظم للطالب أسبوعيا	
Unstructured SWL (h/sem) الحمل الدراسي غير المنتظم للطالب خلال الفصل		Unstructured SWL (h/w) الحمل الدراسي غير المنتظم للطالب أسبوعيا	
Total SWL (h/sem) الحمل الدراسي الكلي للطالب خلال الفصل			

Module Evaluation

تقييم المادة الدراسية

		Time/Number	Weight (Marks)	Week Due	Relevant Learning Outcome
Formative assessment	Quizzes	5	5% (5)	3,5,7,9,11	LO #1, #3 and #4
	Assignments	5	10% (10)	2,4,6,8,10	LO #1, #3 and #4
	Projects / Lab.	10	20% (20)	Continuous	All
	Onsite Assignments	5	5% (5)		LO #5, #8 and #10
	Seminar				
Summative assessment	Midterm Exam	2hr	10% (10)	9	LO #1, #2 and #3
	Final Exam	3hr	50% (50)	17	All
Total assessment			100% (100 Marks)		

Delivery Plan (Weekly Syllabus)

المنهاج الاسبوعي النظري

	Material Covered
Week 1	Problem solving
Week 2	Algorithms and flow charts
Week 3	Introduction to programming Languages
Week 4	Variables, Constants, keywords, types, operators, expression, assignment
Week 5	Simple I/O Functions
Week 6	Conditional Statements
Week 7	If Statement
Week 8	Nested If
Week 9	Mid Exam
Week 10	Switch Statement

Week 11	Iterative control statements + for Statements
Week 12	While Statement
Week 13	Do while
Week 14	Nested Loops
Week 15	Nested while
Week 16	Preparatory week before the final Exam

Delivery Plan (Weekly Lab. Syllabus)

المنهاج الاسبوعي للمختبر

	Material Covered
Week 1	IDE of Programming Language
Week 2	Examples for Algorithms and flow charts
Week 3	Using the IDE for writing sample of program
Week 4	Programs by using Variables, Constants, keywords, types, operators, expression, assignment
Week 5	Writing codes for 3 Programs Applying Simple I/O Functions
Week 6	Simple Conditional Statements programs
Week 7	Writing codes of If Statement programs
Week 8	Writing codes of Nested If programs
Week 9	Mid Exam
Week 10	Writing codes of Switch Statement programs
Week 11	Writing codes of Iterative control statements + for Statements programs
Week 12	Writing codes of While Statement programs
Week 13	Writing codes of Do while programs
Week 14	Writing codes of Nested Loops programs
Week 15	Writing codes of Nested while programs

Learning and Teaching Resources

مصادر التعلم والتدريس

	Text	Available in the Library?
Required Texts	C++: The Complete Reference, Fourth Edition, Herbert Schildt.	Yes
Recommended Texts	The C++ Programming Language, Third Edition, Bjarne Stroustrup.	No
Websites	https://stackoverflow.com/	



أ.م.د. شفيق مصطفى عزت
2022/2023

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.