

MODULE DESCRIPTION FORM

نموذج وصف المادة الدراسية

Module Information				
معلومات المادة الدراسية				
Module Title	Python Programming		Module Delivery	
Module Type	Basic learning activities		☒ Theory ☒ Lecture ☒ Lab ☐ Tutorial ☐ Practical ☐ Seminar	
Module Code	CET2202			
ECTS Credits	5			
SWL (hr/sem)	150			
Module Level	2	Semester of Delivery		4
Administering Department	CET	College	EETC	
Module Leader	Loay Talib Ahmed		e-mail	Loay.alsaffar@mtu.edu.iq
Module Leader's Acad. Title	Lecturer	Module Leader's Qualification	Ph.D.	
Module Tutor	Dr. Hisham Raad jafer MERZEH		e-mail	hisham.merzeh@mtu.edu.iq
Peer Reviewer Name	Dr. Osama abbas hussein		e-mail	Osama.abbas@mtu.edu.iq
Scientific Committee Approval Date	13/06/2023	Version Number	1.0	

Relation with other Modules			
العلاقة مع المواد الدراسية الأخرى			
Prerequisite module	Programming Essentials / CET1203	Semester	2
Co-requisites module	None	Semester	

Module Aims, Learning Outcomes and Indicative Contents

أهداف المادة الدراسية ونتائج التعلم والمحتويات الإرشادية

Module Aims أهداف المادة الدراسية	1. Introduce students to the fundamental concepts and principles of Python programming language. 2. Develop students' proficiency in writing Python code and solving programming problems. 3. Familiarize students with essential programming constructs, such as variables, data types, control flow structures, and functions. 4. Provide students with a solid foundation in object-oriented programming (OOP) and its application in Python. 5. Enable students to work with various data structures and perform operations on them. 6. Prepare students for practical application of Python in real-world scenarios, such as data manipulation, web scraping, and GUI development.
Module Learning Outcomes مخرجات التعلم للمادة الدراسية	1. Understand the fundamentals of Python programming language, including variables, data types, and basic operators. 2. Demonstrate proficiency in control flow structures, such as conditional statements and loops, to control program execution. 3. Develop functions and utilize function arguments to enhance code modularity and reusability. 4. Utilize exception handling techniques to effectively manage errors and ensure program robustness. 5. Gain familiarity with modules and packages to leverage existing code and extend Python's functionality. 6. Understand object-oriented programming (OOP) concepts and apply them to create classes, objects, and inheritance hierarchies. 7. Manipulate strings, lists, dictionaries, and sets to efficiently store and retrieve data. 8. Perform file handling operations, including reading from and writing to files. 9. Apply Python to practical tasks, such as web scraping, data manipulation, and analysis. 10. Demonstrate proficiency in working with files and directories, including navigating file systems and managing file permissions. 11. Develop graphical user interfaces (GUIs) using Python libraries to create interactive applications. 12. Prepare for exams by reviewing course materials, practicing exercises, and answering sample questions.
Indicative Contents المحتويات الإرشادية	Indicative content includes the following. <u>Part A: Introduction to Python and Basic Concepts (Estimated time: 10 hours)</u> Overview of Python programming language Installation and setup Variables and data types Basic operators Input and output operations

	<u>Part B: Control Flow and Functions (Estimated time: 16 hours)</u> Conditional statements (if, else, elif) Loops and iterations (for loop, while loop) Functions and function arguments Recursion <u>Part C: Data Structures and File Handling (Estimated time: 16 hours)</u> Strings and string manipulation Lists and list manipulation Dictionaries and sets File handling and input/output operations <u>Part D: Advanced Topics (Estimated time: 16 hours)</u> Exception handling and error management Modules and packages Object-oriented programming (OOP) concepts Classes, objects, inheritance, and polymorphism <u>Part E: Applications and Practical Projects (Estimated time: 16 hours)</u> Working with files and directories GUI programming Web scraping Data manipulation and analysis
--	--

Learning and Teaching Strategies

استراتيجيات التعلم والتعليم

Strategies	Effective learning and teaching strategies involve creating an engaging and interactive learning environment. This can be achieved through a combination of various approaches, such as incorporating active learning techniques like group discussions, problem-solving activities, and hands-on experiments. Additionally, employing visual aids, multimedia resources, and real-world examples can enhance comprehension and retention. Encouraging student participation and providing timely feedback also play vital roles in fostering student engagement and understanding. It is important to promote a growth mindset, encourage critical thinking, and create opportunities for collaboration and peer learning. By employing these strategies, educators can facilitate meaningful learning experiences and empower students to become active participants in their own learning journey.
--------------------------	--

Student Workload (SWL)

الحمل الدراسي للطالب

Structured SWL (h/sem) الحمل الدراسي المنتظم للطالب خلال الفصل	74	Structured SWL (h/w) الحمل الدراسي المنتظم للطالب أسبوعياً	5.2
Unstructured SWL (h/sem) الحمل الدراسي غير المنتظم للطالب خلال الفصل	76	Unstructured SWL (h/w) الحمل الدراسي غير المنتظم للطالب أسبوعياً	5.4
Total SWL (h/sem) الحمل الدراسي الكلي للطالب خلال الفصل	150		

Module Evaluation

تقييم المادة الدراسية

		Time/Number	Weight (Marks)	Week Due	Relevant Learning Outcome
Formative assessment	Quizzes	2	10% (10)	5, 10	LO #1-5, LO #5-8
	Assignments				
	Projects / Lab.	1	10% (10)	Continuous	
	Report	1	5% (5)	13	LO # 1-12
Summative assessment	Midterm Exam	2 hrs.	25% (25)	7	LO # 1-7
	Final Exam	4hrs.	50% (50)	16	All
Total assessment			100% (100 Marks)		

Delivery Plan (Weekly Syllabus)

المنهاج الاسبوعي النظري

	Material Covered
Week 1	Introduction to Python, Variables, Data Types, and Basic Operators
Week 2	Control Flow and Conditional Statements
Week 3	Loops and Iterations
Week 4	Strings and String Manipulation
Week 5	Lists and List Manipulation
Week 6	Dictionaries and Sets
Week 7	Midterm Exam
Week 8	Functions and Function Arguments
Week 9	File Handling and Input/Output Operations
Week 10	Exception Handling and Error Management
Week 11	Modules and Packages
Week 12	Object-Oriented Programming (OOP) Concepts
Week 13	Classes and Objects
Week 14	Inheritance and Polymorphism
Week 15	Working with Files and Directories
Week 16	Preparatory Week for Final Exam

Delivery Plan (Weekly Lab. Syllabus)

المنهاج الاسبوعي للمختبر

	Material Covered
Week 1	Introduction to Python, Variables, and Basic Operators
Week 2	Control Flow and Conditional Statements
Week 3	Loops and Iterations
Week 4	Strings and String Manipulation
Week 5	Lists and List Manipulation
Week 6	Dictionaries and Sets
Week 7	Midterm Exam (No lab session).
Week 8	Functions and Function Arguments
Week 9	File Handling and Input/Output Operations
Week 10	Exception Handling and Error Management
Week 11	Modules and Packages
Week 12	Object-Oriented Programming (OOP) Concepts
Week 13	Classes and Objects
Week 14	Inheritance and Polymorphism
Week 15	Working with Files and Directories
Week 16	Final Exam (No lab session).

Learning and Teaching Resources

مصادر التعلم والتدريس

	Text	Available in the Library?
Required Texts	Title: "Python Crash Course: A Hands-On, Project-Based Introduction to Programming" Author: Eric Matthes	
Recommended Texts	Title: "Learning Python" Author: Mark Lutz	No
Websites	URL: https://realpython.com	

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.				