MODULE DESCRIPTION FORM

نموذج وصف المادة الدراسية

Module Information معلومات المادة الدراسية					
Module Title	**Operating Systems**			**Module Delivery**	
Module Type	Core			☒ **Theory** ☒ **Lab**	
Module Code	**IT3107**				
ECTS Credits	**6**				
SWL (hr/sem)	**150**				
Module Level	UGIII		**Semester of Delivery**	1	
Administering Department	Information Technology		**College**	College of Computer Science and Information Technology	
Module Leader	Ali Abdul Hussein Ibrahim		**e-mail**	aliabdulhussein@uowa.edu.iq	
Module Leader's Acad. Title	Assistant Lecturer		**Module Leader's Qualification**	MSc	
Module Tutor	Ali Abdul Hussein Ibrahim		**e-mail**	aliabdulhussein@uowa.edu.iq	
Peer Reviewer Name	Dr. Ali Karim Abdulrahim		**e-mail**	alialmujab@uowa.edu.iq	
Scientific Committee Approval Date	01/09/2026		**Version Number**	V-1	

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

Department Head Approval **Dean of the College Approval**

Module Aims, Learning Outcomes and Indicative Contents	
Module Aims	1. Understand the rationale behind the current design and implementation of modern OS's by considering the historic evolution of various OS 2. Familiarize students with the all services provided by Operating System. 3. Understand different approaches to memory management. 4. Students should be able to use system calls for managing processes, memory and the file system. 5. Understand the importance of using threads and processes in an operating system. 6. Recognize and differentiate between uni-process and multi processes and how processes are synchronized and scheduled. 7. Learn how to maintain a consistent view of data across the OS. 8. Expose the role of synchronization in OS and the problems that could arise if synchronization is not handled properly. 9. Identify the importance of Semaphores as a way of preventing Race Conditions and other more advanced alternatives techniques
Module Learning Outcomes	1. Identify the fundamental concepts of operating system and the main services provided. 2. Categorize the various components of a computer system and how they interact with an operating system; 3. Explain the different types of operating systems and the major systems in use today. 4. Describe the System calls and identify main system call categories. 5. Discuss the importance processes and threads and in an operating system; 6. Describe the process concurrency issues. 7. Discuss context switching and how it is used in an operating system. 8. Identify classic synchronization problems such as race condition and inter-process communication. 9. Describe how semaphores can be used in an operating system to prevent synchronization issues.
Indicative Contents	Indicative content includes the following. 1. **Introduction to Operating Systems** It deals with a high level introduction to Operating Systems (OS). The Operating System acts as a platform of information exchange between your computer's hardware and the applications running on it. First, it starts with a discussion on some of the earliest Operating Systems. Review the general OS structure and give a basic functional overview. Discuss the services of the modern Operating Systems and devices that we are familiar with. 2. **System Calls and Interrupts**. It explains the system call(a method for a computer program to request a service from the kernel of OS) and its role in operating systems on which it is running; it is a method of interacting with the operating system via programs. It acts as a link between the operating system and a process, allowing user-level programs to request operating system services. The kernel system can only be accessed using system calls. Identify the main categories of system calls in various operating systems 3. **Processes and Threads** It discusses two central building blocks of modern operating systems: Processes and Threads. Processes (instances

	of a running computer program) and threads (a specific task running within a program) are integral to the understanding of how an OS executes a program and the communication of information between each of the computer's architectural layers. We will start with an overview of each concept, including definitions, uses, and types. it discusses the commonalities and differences between processes and threads. It ends with a discussion on Context Switches and the important role they play in CPU scheduling. 4. **Synchronization** Generally, there is a number of different entities will need to access data, it is important to learn how to maintain a consistent view of data across the OS. That's why we need a good synchronization management system. Providing an overview of why synchronization is so important in an Operating System and the problems that could arise if synchronization is not handled properly. Identify the main synchronization issues such as Race Conditions ,or system flaws in which the output of a given process is problematically dependent on the sequence of other events **5. Semaphores** Semaphore is a way of preventing Race Conditions and other more advanced alternatives to Semaphores, such as Monitors and Messages. Semaphores are used to solve synchronization problems, but there are some advantages to using them: • Semaphores impose deliberate constraints that help programmers avoid errors. • Solutions using semaphores are often clean and organized, making it easy to demonstrate their correctness. 1. • Semaphores can be implemented efficiently on many systems, so solutions that use semaphores are portable and usually efficient

Learning and Teaching Strategies	
Strategies	The learning and teaching strategies for studying the Operating systems in an IT department comprises a balanced strategy of theoretical understanding and practical application. Lectures, interactive discussions, practical exercises that come from theoretical foundation and seminars. Group work, ring discussion and projects enable hands-on experience with operating systems. Online resources, assessments, and feedback aid in reinforcing learning. Virtual labs and consistent learning that support the practical skills development and staying updated with cutoff trends. These strategies ensure a comprehensive understanding of operating systems and their impact in the IT field.

Student Workload (SWL)			
Structured SWL (h/sem)	60	**Structured SWL (h/w)**	5
Unstructured SWL (h/sem)	87	**Unstructured SWL (h/w)**	6
Total SWL (h/sem)	147 + 3 final = 150		

Module Evaluation

		Time/Number	Weight (Marks)	Week Due	Relevant Learning Outcome
Formative assessment	**Quizzes**	4	2 (8%)	2,4,8,10	1,2,3,4
	Project	1	5 (5%)	12	all
	Lab	5	3 (15%)	all	all
	H. W	4	2 (8%)	3,5,9,11	all
	Onsite Assignments	2	2 (4%)	3,11	all
Summative assessment	**Midterm Exam**	1/1hr	10%(10)	7	
	Final Exam	1/3hr	50%(50)	16	
Total assessment			100%(100 Mark)		

Delivery Plan (Weekly Syllabus)	
	Material Covered
Week 1	Introduction to Operating Systems: concepts and functionality, Abstract Computer Component Overview
Week 2	History of Operating Systems, Different Operating Systems Overview(UNIX-based, Linux (Ubuntu) Windows, Mobile, Real Time)
Week 3	Types of Operating System Structures :: Batch Operating System, Layered Operating System , Exo- kernel Operating System, and Mico- kernel l Operating System
Week 4	Operating System Services
Week 5	System Calls in Operating System, Interrupts
Week 6	Process Management in OS , Attributes of a Process, Process States, Process Schedulers, Process Queues
Week 7	Process Context Switching
Week 8	Threads :Threads Creation, Threads Attributes, Threads versus Processes,
Week 9	Concurrency: Processes, Threads, and Address Spaces
Week 10	Process Synchronization Problems (Critical Section Problem): mutual exclusion, progress, bounded waiting.
Week 11	Solution for Critical section problem: Lock Variable, Paterson Solution, sleep and wake
Week 12	Processes & Threads Creation, intro to race conditions
Week 13	Semaphores: Definition, Syntax, Semaphore type
Week 14	Classical synchronization problems (Mutex, Readers-Writers Problem, Dining-Philosophers Problem)
Week 15	Producer-consumer problem and solution
Week 16	Preparatory week before the final Exam

Delivery Plan (Weekly Lab Syllabus)	
	Material Covered
Week 1	Lab 1: Setting up the Linux(Ubuntu) environment
Week 2	Lab 2: Learn interactive system communication terminal, directions, and editors
Week 3	Lab 3: practicing basic system calls, and how to use it.
Week 4	Lab 4: Learn about Operating system code using C and SYSTEM CALL library
Week 5	Lab 5: Learn Files in C using terminal and editor, read and write(word line, and block)
Week 6	Lab 6: Read and identify variables though terminal only using C ode
Week 7	Lab 7: Learn Process library, Process creation, and Process termination using C Code,
Week 8	Lab 8: Implementation of multi processes program with different actions Computing the summation, find maximum, or find minimum
Week 9	Lab 9: Learn basic thread creation , attributes and termination using C code
Week 10	Lab 10: Implementation of single tread program like Computing the Average using One Thread
Week 11	Lab 11: Synchronization in pthreads: concurrency in the thread runtime
Week 12	Lab 12: Learn and implement multi-threading program
Week 13	Lab 13: Learn Race condition problem in of multi- threading runtime
Week 14	Lab 14:Implement Race condition problem in mutual exclusion of multi- threading
Week 15	Lab 15: Apply mutex (mutual locks) to solve Race condition problem in mutual exclusion, build and Implement integrated project for each student

Learning and Teaching Resources		
	Text	**Available in the Library?**
Required Texts	Operating System Concepts, ABRAHAM SILBERSCHATZ, PETER BAER GALVIN, GREG GAGNE, 9 EDITION, Copyright!© 2013, 2012, 2008 John Wiley& Sons	Yes
Recommended Texts	Operating Systems: Internals and Design Principles, William Stallings , 28 Feb 2011.	no
Websites	https://www.quora.comhttps://www.sanfoundry.com/operating-system-mcqs-application-io-interface-1 https://www.geeksforgeeks.org/operating-systems/ https://www.virtualbox.org https://www.ubuntu.com/download/desktop	

APPENDIX:

GRADING SCHEME				
Group	**Grade**	**Mark**	**Marks (%)**	**Definition**
Success Group (50 - 100)	**A -** Excellent	Excellent	90 - 100	Outstanding Performance
	B - Very Good	Very Good	80 - 89	Above average with some errors
	C - Good	Good	70 - 79	Sound work with notable errors
	D - Satisfactory	Fair / Average	60 - 69	Fair but with major shortcomings
	E - Sufficient	Pass / Acceptable	50 - 59	Work meets minimum criteria
Fail Group (0 – 49)	**FX –** Fail	Fail (Pending)	(45-49)	More work required but credit awarded
	F – Fail	Fail	(0-44)	Considerable amount of work required
Note:				

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.

ملاحظة: هذا النموذج تم وضعه وتقديمه من قبل مديرية ضمان الجودة في وزارة التعليم العالي والبحث العلمي