Ministry of Higher Education and Scientific Research
Scientific Supervision and Scientific Evaluation Apparatus
Directorate of Quality Assurance and Academic Accreditation
Accreditation Department

Academic Program and Course Description Guide

2025

Introduction:

The educational program is a well-planned set of courses that include procedures and experiences arranged in the form of an academic syllabus. Its main goal is to improve and build graduates' skills so they are ready for the job market. The program is reviewed and evaluated every year through internal or external audit procedures and programs like the External Examiner Program.

The academic program description is a short summary of the main features of the program and its courses. It shows what skills students are working to develop based on the program's goals. This description is very important because it is the main part of getting the program accredited, and it is written by the teaching staff together under the supervision of scientific committees in the scientific departments.

Academic Program Description Form

University Name: University of Basrah

Faculty/Institute: College of Computer Science & Information Technology

Scientific Department: Department of Computer Science

Academic or Professional Program Name: Bachelor in Computer Science

Final Certificate Name: Bachelor in Computer Science

Academic System: Semesters

Description Preparation Date: 14/9/2025

File Completion Date: 14/9/2025

Signature:

Head of Department Name:

Asst. Prof. Saba A. Saddam

Date: 14/9/2025

Signature:

Scientific Associate Name:

Prof. Abbas H. Hasan

Date: 14/09/2025

Department of Quality Assurance and University Performance

Director of the Quality Assurance and University Performance Department:

Dr. Arafat N. Jasim

Date:

Signature:

Approval of the Dean

1. Program Vision

Our vision is to be a leading computer science department recognized for its innovation, excellence, and societal impact. We strive to be at the forefront of computer science education, research, and technology development, equipping our students with the knowledge, skills, and mindset necessary to address complex challenges and make meaningful contributions in academia, industry, entrepreneurship, and public service. We aim to cultivate a culture of curiosity, collaboration, and creativity, where interdisciplinary approaches are embraced, ethical considerations are paramount, and technological advancements are leveraged to address global problems and improve the human condition.

2. Program Mission

The mission of our computer science department is to provide a comprehensive and rigorous education in computer science that prepares students for successful careers, advanced studies, and lifelong learning in the rapidly evolving field of computing. We are committed to fostering a diverse and inclusive community of learners and researchers, promoting excellence in teaching and research, and engaging in collaborations that contribute to the advancement of computer science and its applications.

3. Program Objectives

1. Prepare and qualify specialists to meet the demands of the public and private labor market in computer science and information technology by diversifying learning and teaching methods and training students to apply acquired knowledge and skills to solve real-life problems.
2. Create an appropriate environment for students, enabling them to apply their acquired knowledge and skills to identify the needs and problems of society and social issues related to computers and information technology.
3. Offer distinguished academic programs in computer science and information technology, both theoretical and applied, that comply with international standards for academic quality and meet the needs of the labor market.
4. Encourage and develop scientific research in the fields of computer science and information technology in general, and in the fields of artificial intelligence, linguistics, software, networks, and databases.
5. Creating a stimulating environment for faculty members to develop their knowledge, teaching, and research skills.
6. Building and developing partnerships with the government, private sectors, and the community, including all its various institutions.

7. Program Accreditation

8. Other external influences

None

9. Program Structure

Program Structure	Number of Courses	Credit hours	Percentage	Reviews*
Institution Requirements	7	14	9%	
College Requirements	7	20	14%	
Department Requirements	25	108	77%	
Summer Training	1	0		
Other				

*This can include notes whether the course is basic or optional.

10.Program Description

Year/Level	Course Code	Course Name	Credit Hours	
			theoretical	practical
First Year First Semester		Programming I	3	2
		Mathematics for computing	3	
		Computer Skills	2	2
		English Language I	2	
		Democracy Education & Human Rights	2	
		Fitness and Sport	2	
First Year Second Semester		Programming II	3	2
		Digital Logic Design	3	2
		Computer Applications	2	2
		Discrete Structures	3	
		Principles of Information technology	2	2
		English Language II	2	
Second Year First Semester		Object Oriented programming I	2	2
		Computer Graphics	2	2
		Microprocessor and assembly language	2	2
		System Analysis and Design	3	
		Probability and statistics	3	
		Arabic Language Skills	2	

Second Year Second Semester		Object Oriented programming II	2	2
		Visual Programming	2	2
		Computation theory	3	
		Database concepts and design	2	2
		Data Structures I	2	2
		Numerical methods	2	2
Third Year First Semester		Artificial Intelligence	2	2
		Software Engineering	3	
		Web Programming I	2	2
		Computer Networks I	3	
		Data Structures II	2	2
		Concepts of Programming languages	2	
Third Year Second Semester		Complier construction	3	2
		Computer Network II	2	2
		Web Programming II	2	2
		Operations Research	3	
		Computer Ethics	2	
		Computer Architecture	3	
Fourth Year First Semester		Operating Systems	2	2
		Mobile Applications Programming	2	2
		Computer vision	2	2
		Data Mining	2	
		Cloud computing	2	
		Computational Intelligence	3	
		Graduation Project		4
Fourth Year Second Semester		Computer simulation	3	
		Computer Security	3	
		Human-Computer Interaction	3	
		Knowledge Engineering	3	
		Communication Skills	3	
		Selected Topics	3	

8. Expected learning outcomes of the program

Knowledge

A1. The student will learn programming languages, the skills of designing various application programs using several programming languages, and finding scientific solutions to societal problems through programming.
A2. The student will be taught the basics of computer network management and the ability to use and develop wired and wireless communication and networking tools, in addition to teaching the student the skills of website design and supervision.
A3. The student will be provided with the basic rules for evaluating and building software systems, enabling them to analyze and evaluate systems before beginning to design the system. The student's knowledge of the basics of implementing software systems will increase through understanding the mechanisms of computer operation.
A4. The student's skills in building intelligent systems, which are based on analysis, inference, heuristics, and self-learning, will be developed.

Skills

B1. Design, write, and debug software using programming languages.
B2. Use appropriate computer-designed support tools.
B3. Master the skills of research, report writing, presentation, discussion, and internet research related to course topics.
B4. Master the skills of critical and analytical thinking and problem-solving.

Ethics

C1. The student develops a positive attitude toward learning computer science.
C2. The student takes pride in his practical skills when directly using the computer.
C3. The student participates and cooperates with his classmates to produce public service websites.
C4. The student senses the importance of the knowledge he receives in facilitating many of the tasks he performs.

9. Teaching and Learning Strategies

10. Evaluation methods

1. Central and monthly exams.
2. Instant exams.
3. Scientific reports.
4. Practical exams.
5. Research projects.

11. Faculty

Faculty Members

Academic Rank	Specialization		Special Requirements/Skills (if applicable)	Number of the teaching staff	
	General	Special		Staff	Lecturer
Professor	Computer Science			3	
Assistant Professor	Computer Science			7	
Lecturer	Computer Science			11	2
Assistant Lecturer	Computer Science			12	2

Professional Development

Mentoring new faculty members

- E-Learning
- Attending training courses and workshops
- Attending conferences
- Cooperating with professional faculty members

Professional development of faculty members

12. Acceptance Criterion

- Central Admission
- The student's average with the student's desire to be accepted in departments.

13. The most important sources of information about the program

College website:
https://en.cit.uobasrah.edu.iq/

14. Program Development Plan

Program Skills Outline															
				Required program Learning outcomes											
Year/Level	Course Code	Course Name	Basic or optional	Knowledge				Skills				Ethics			
				A1	A2	A3	A4	B1	B2	B3	B4	C1	C2	C3	C4
First		Programming1	Basic	☐		☐		☐	☐	☐	☐	☐	☐	☐	☐
		Programming2	Basic	☐		☐		☐	☐	☐	☐	☐	☐	☐	☐
		Logic Design	Basic	☐		☐		☐	☐	☐	☐	☐	☐	☐	☐
		Discrete Structures	Basic							☐	☐				☐
		Computer skills	Basic	☐	☐				☐	☐		☐	☐	☐	☐
Second		Object-Oriented Programming 1	Basic	☐		☐		☐	☐	☐	☐	☐	☐	☐	☐
		Object-Oriented Programming 2	Basic	☐		☐		☐	☐	☐	☐	☐	☐	☐	☐
		Data Structures 1	Basic	☐		☐		☐	☐	☐	☐	☐	☐	☐	☐
		Visual Programming	Basic	☐		☐		☐	☐	☐	☐	☐	☐	☐	☐
		Microprocessors	Basic	☐		☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
		Computation Theory	Basic			☐	☐				☐	☐	☐		☐
Third		Software Engineering	Basic	☐		☐		☐	☐			☐		☐	
		Artificial Intelligent	Basic	☐		☐	☐		☐			☐	☐	☐	☐
		Computer Networks 1	Basic	☐		☐	☐		☐			☐	☐	☐	☐
		Computer Networks 2	Basic	☐		☐	☐		☐			☐	☐	☐	☐

		Computer Architecture	Basic			□	□	□	□	□	□	□	□	□	□
		Compiler Construction	Basic	□		□		□	□	□	□	□	□	□	□
		Data Structures 2	Basic	□		□		□		□	□	□	□	□	□
		Web Programming 1	Basic	□		□		□	□	□	□	□	□	□	□
		Web Programming 2	Basic	□		□		□	□	□	□	□	□	□	□
Fourth		Operating Systems	Basic			□	□	□	□	□	□	□	□	□	□
		Computer Security	Basic	□	□	□	□	□	□	□					
		Mobile Applications	Basic	□		□	□	□	□	□	□	□	□	□	□
		Communication Skills	Basic				□			□	□				□
		Cloud Computing	Basic	□	□	□	□	□	□	□					

First Year - First Semester

Programming I

1. Course Name:	
Programming I	
2. Course Code:	
UoB12345	
3. Semester / Year:	
1st / 2025-2026	
4. Description Preparation Date:	
01/06/2023	
5. Available Attendance Forms:	
6. Number of Credit Hours (Total) / Number of Units (Total):	
125/5	
7. Course administrator's name (mention all, if more than one name):	
Name: Maalim A. Aljabery Email: maalim.aljabery@uobasrah.edu.iq	
8. Email: Course Objectives	
Course Objectives	By the end of this course, students should be able to: • **Understand Core Programming Concepts:** Grasp the fundamentals of programming, including variables, data types, operators, expressions, and control structures. • **Develop Problem-Solving Skills:** Analyze problems logically and design algorithms to solve them efficiently. • **Write and Debug Programs:** Implement basic programs using a high-level programming language and effectively debug and test them. • **Apply Structured Programming Techniques:** Use modular design, functions, and proper coding practices to create clear and maintainable programs. • **Build Computational Thinking:** Strengthen logical reasoning and algorithmic thinking applicable to real-world computing problems. • **Prepare for Advanced Courses:** Establish a foundation for subsequent programming and software development courses in the undergraduate curriculum.

9. Teaching and Learning Strategies	
Strategy	When teaching a programming I course to beginners, it's important to adopt strategies that cater to their foundational understanding and gradually build their knowledge and skills. Here are some effective learning and teaching strategies for beginners in a Programming I course:

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
16	3/W		Programming I		

11. Course Evaluation

12. Learning and Teaching Resources

Required textbooks (curricular books, if any)	• **Algorithm and Programming Fundamentals**: *"Introduction to Programming Using Java"* by David J. Eck *"Java Programming: From Problem Analysis to Program Design"* by D.S. Malik
Main references (sources)	
Recommended books and references (scientific journals, reports...)	**Textbooks and Reference Books** • **Core Java Programming Books**: ○ *"Java: How to Program"* by Deitel & Deitel ○ *"Head First Java"* by Kathy Sierra & Bert Bates ○ *"Effective Java"* by Joshua Bloch (for best practices, even at an introductory level)
Electronic References, Websites	**Online Resources** • **Official Java Documentation**: Oracle Java Documentation • **Educational Platforms**: Codecademy – Java Course Coursera, edX, Udemy – Java Fundamentals Courses • **Interactive Coding Websites**: HackerRank – Java Practice LeetCode – Java Problems

Mathematics For Computing

Module Information					
Module Title	Mathematics For Computing			Module Delivery	
Module Type	Core			☒ Theory ☒ l Lectures	
Module Code					
ECTS Credits					
SWL (hr/sem)	100				
Module Level	1		Semester of Delivery	1	
Administering Department	Type Dept. Code		College	Type College Code	
Module Leader	Naser Oda Jassim		e-mail	Nasir.jasim@uobasrah.edu.iq	
Module Leader's Acad. Title	Lecturer		Module Leader's Qualification	Ph.D.	
Module Tutor	Name (if available)		e-mail	E-mail	
Peer Reviewer Name	Name		e-mail	E-mail	
Scientific Committee Approval Date	13/09/2025		Version Number	1.0	

Relation with other Modules			
Prerequisite module	Mathematics for computing	Semester	
Co-requisites module	None	Semester	

Module Aims, Learning Outcomes and Indicative Contents	
Module Objectives	-Cognitive Goals 1. Upon Successful completion of this subject, students should : 2. Be able to use algebra accurately; 3. Be able to plot and interpret graphs 4. Be able to use exponential, logarithm, and trigonometric functions in applications; 5. Be able to calculate the sums of arithmetic and geometric series and use them in simple financial calculations; 6. Be able to use basic rules of differentiation and calculate derivatives of simple functions; 7. Be able to use matrices in solving linear systems of equations; -Skill goals 1. Enable the student to refer the mathematical problem to a program and find a solution through the computer. 2. Student realization of the close relationship between mathematical problems and computer programs

Module Learning Outcomes	Important: Write at least 6 Learning Outcomes, better to be equal to the number of study weeks. 1. This subject is designed for students who enter university without a strong background in mathematics 2. It is also for students who are planning to enrol in subjects requiring basic numeracy skills, such as sciences, computing and information technology. 3. The subject reinforces calculation skills and basic algebra. 4. This subject is designed to work with formulas. 5. It is also to use applications of exponential and logarithmic functions. 6. It is designed to apply matrix to solve linear systems of equations.
Indicative Contents	Indicative content includes the following. Part A – Sequences and series Sequence is a function whose domain is the set of natural numbers. The terms of the sequence are the function values. There will be studied two types of sequences: arithmetic and geometric sequences with their partial sums. While a series means the infinite sum of a geometric sequence. [12 hrs] Part B – Matrices Matrices are simply a rectangular array of numbers with m rows and n columns. There will be studied some: types of matrices, algebra of matrices. It is also studied how to find the inverse of a matrix, how to use a matrix and its inverse to solve a linear system of equations, and how to find the determinant of a matrix and use it to solve a linear system of equations. [12 hrs] Part C – Derivatives and integrals Derivatives mean that if $f: x \to y$ is a function, the derivative of a function f at a point x_0 written $f'(x_0)$; is given by $f'(x_0) = \lim_{x \to x_0} \frac{f(x)-f(x_0)}{x-x_0}$, If this limit exists and finite. There will be studied the derivatives of usual functions, implicit derivatives, derivatives of trigonometric functions, derivatives of exponential and logarithm functions. Graphical of exponential and logarithm functions. While integrals means that if $f(x)$ function defined at some interval, let $F(x)$ be another function such that $F'(x) = f(x)$, $F(x)$ called an infinite integral of $f(x)$ and is written as the following form $\int \mathrm{f(x)dx} = \mathrm{F(x))} + \mathrm{C}$. [12 hrs]. Part D – Interest Interest is the rental fee charged by a lender to a business or an individual for the use of money. There will be studied simple and compound interests. Simple interest means that the interest is calculated *only once* for the entire time period of the loan. At the end of the time period, the borrower repays the principal plus the Interest . while compound interest means that means that the interest is calculated more than once during the time period of the loan. [9 hrs].
Learning and Teaching Strategies	
Strategies	1.Explain the topic in detail by the teacher by writing the topic and explaining it on the board and other teaching aids

	2. Discussion during the lecture period 3. Doing homework 4. See the websites of the subject

Student Workload (SWL)			
Structured SWL (h/sem)	102	Structured SWL (h/w)	7
Unstructured SWL (h/sem)	98	Unstructured SWL (h/w)	6
Total SWL (h/sem)	200		

Module Evaluation					
		Time/Number	Weight (Marks)	Week Due	Relevant Learning Outcome
Formative assessment	Quizzes	2	10% (10)	5 and 10	LO #1, #2 and #10, #11
	Assignments	2	10% (10)	2 and 12	LO #3, #4 and #6, #7
	Projects / Lab.	1	10% (10)	Continuous	All
	Report	1	10% (10)	13	LO #5, #8 and #10
Summative assessment	Midterm Exam	2hr	10% (10)	7	LO #1- #7
	Final Exam	3hr	50% (50)	16	All
Total assessment			100% (100 Marks)		

Delivery Plan (Weekly Syllabus)	
	Material Covered
Week 1	Introduction- Sequences
Week 2	Arithmetic sequences and their partial sims
Week 3	Geometric sequences and their partial sums
Week 4	Series
Week 5	Matrices and algebra of matrices
Week 6	Inverse of matrices
Week 7	Solving linear system of equations by using inverse of matrices
Week 8	Determinant and using it to solve linear system of equations
Week 9	Derivatives
Week 10	Derivatives of trigonometric, exponential, logarithm functions
Week 11	Integrals
Week 12	Integral of trigonometric, exponential, logarithm functions
Week 13	Interest and simple interest

Week 14	Compound interest
Week 15	Present and future values of an annuity
Week 16	Preparatory week before the final Exam

Learning and Teaching Resources		
	Text	Available in the Library?
Required Texts	Cheryl Cleaves, Margie Hobbs and Jeffry Noble	Yes
Recommended Texts	James Stewart , Lothar Redlin and Saleem Watson Robert Brechner and George Bergeman	yes
Websites		

Grading Scheme			
Group	Grade	Marks %	Definition
Success Group (50- 100)	A- Excellent	90- 100	Outstanding Performance
	B- Very Good	80- 89	Above average with some errors
	C- Good	70- 79	Sound work with notable errors
	D- Satisfactory	60- 69	Fair but with major shortcomings
	E- Sufficient	50- 59	Work meets minimum criteria
Fail Group (0 – 49)	FX – Fail	(45-49)	More work required but credit awarded
	F – Fail	(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.

Computer Skills

1. Course Name:	
Computer Skills	
2. Course Code:	
3. Semester / Year:	
First semester	
4. Description Preparation Date:	
13 / 9 / 2025	
5. Available Attendance Forms:	
6. Number of Credit Hours (Total) / Number of Units (Total)	
75	
7. Course administrator's name (mention all, if more than one name)	
Name: Email:	
8. Email: Course Objectives	
Course Objectives	This course aims at teaching students how to use a variety of computer applications as tools to improve students' performance in school, increase their future productivity in the work place and enhance their level of critical thinking. Students will use computer networks and applications to locate, evaluate, and use information, create written documents and oral presentations. This course will assist students in understanding the underlying concepts of these technologies and provide project-oriented learning opportunities. The goal is for students to become independent users of information, computer technology and library resources.
9. Teaching and Learning Strategies	
Strategy	The primary approach for delivering this module will focus on fostering active student engagement in exercises, while simultaneously enhancing their critical thinking abilities. This will be accomplished through a combination of classroom and laboratory sessions, interactive tutorials, and the incorporation of captivating

	sampling activities to facilitate hands-on learning experiences for the students.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
		The student will acquire fundamental computer skills that can be effectively applied to data processing and presentation tasks. This includes gaining proficiency in essential computer operations, such as file management, utilizing productivity tools, and navigating digital interfaces. Through practical application, the student will develop the ability to handle and manipulate data, as well as create compelling presentations.			

11. Course Evaluation

12. Learning and Teaching Resources

Required textbooks (curricular books, if any)	Microsoft Office 2013 Visual Quickstart Guideby Steve Schwartz
Main references (sources)	
Recommended books and	Gary B. Shelly, Misty E. Vermaat (2010). Microsoft Office 2010: Brief. Cengage Learning. OR any ECDL, ICDL or IC3 books

references (scientific journals, reports...)	
Electronic References, Websites	https://www.microsoft.com

First Year - Second Semester

Programming II

1. Course Name:
Programming II
2. Course Code:
CS106
3. Semester / Year:
2nd Semester / 2024-2025
4. Description Preparation Date:
1/10/2024
5. Available Attendance Forms:
Lectures
6. Number of Credit Hours (Total) / Number of Units (Total)
4 / 141
7. Course administrator's name (mention all, if more than one name)
Name: Dr. Salah F. Saleh Email: aldarraji@uobasrah.edu
8. Email: Course Objectives

Course Objectives	Here are some module aims typically associated with a Programming II course. These aims describe the overarching goals and objectives of the course: 1. This course covers basic concepts and techniques for programming including: repetition statements (while and for). 2. In this course the students can learn how to deal with arrays. 3. The programming II aims to learn how to understand the strings.

9. Teaching and Learning Strategies

Strategy	The key is to move from concrete examples to abstract concepts and back again, constantly reinforcing the "why" behind each topic. **Overarching Teaching Philosophy** • **Code-Alongs & Live Coding:** Don't just show finished code. Build programs live from scratch. This models the thought process, including debugging and problem-solving. • **Predict, Run, Investigate:** For a given code snippet, ask students to: 1. *Predict* the output. 2. *Run* the code (or you run it). 3. *Investigate* why it produced that output. This builds critical analysis skills. • **Scaffolded Projects:** Design projects where Week 5's work becomes a module for Week 7's, which is integrated into Week 14's final project.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
15	45	At the end of this course, students should be able to design, write and test c++ program to implement a working solution to a given problem.	• Repetition • while Looping structure • do..while • Nested Control Structures • Arrays • 2D Arrays • Array as parameter • Strings • Array of string • Structures Compare the structure with the arrays • Access field of structure • Fields Assigning values	Lectures Lab.	• Quizzes • Assignment • Projects • Report

11. Course Evaluation

- Quizzes
- Assignment
- Projects
- Report
- Midterm Exam
- Final Exam

12. Learning and Teaching Resources	
	1. Problem solving with c++ by Walter Savitch, 7th edition,2009. 2. C++: The Complete Reference by Herbert Schildt, 4th edition, 2003
	A first book of c++ by Gary Bronson, 4th edition, 2012 by Gary Bronson

Discrete Structures

1. Course Name:	
Discrete Structures	
2. Course Code:	
3. Semester / Year:	
Second semester	
4. Description Preparation Date:	
13/9/2025	
5. Available Attendance Forms:	
6. Number of Credit Hours (Total) / Number of Units (Total)	
125	
7. Course administrator's name (mention all, if more than one name)	
Name: Dr. Shatha Falih Hendi Email:shatha.falih@uobasrah.edu.iq	
8. Email: Course Objectives	
Course Objectives	1. We can develop our mathematical ability 2. Discrete mathematic is the gateway to more advanced courses in all

	part of math. 3. Discrete mathematics provides the math foundations for many computer science courses 4. Discrete mathematics contains the necessary math back ground for solving problems in operation research, chemistry, and engineering.
9. Teaching and Learning Strategies	
Strategy	1. Convergent and divergent thinking. 2. Project-based learning. 3. Experiential learning. 4. Peer teaching. 5. Inquiry-based learning. 6. Problem-based learning. 7. Reciprocal teaching.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
		1. formulate solutions for selected mathematical problem 2. Apply objective mathematical reasoning to systems composed of discrete objects. 3. Assess mathematical proofs. 4. Interpret situations that have a predetermined sequence of actions that depend on a limited sequence of events. 5. categorize all possible outcomes for a series of events, or all possible collections of a set of objects; 6. diagram			

		hierarchical relationships between individual entities within a given situation using relations; and 7. Diagram hierarchical relationships between individual entities within a given situation using function. 8. apply Trees of mathematical or system entities as tools in computer science to solve various real-world problems; and Apply Graph of mathematical or system entities as tools in computer science to solve various real-world problems.			

11. Course Evaluation

12. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	Essential Discrete Mathematics for Computer Science, by Harry Lewis and Rachel Zax, Princeton University Press , ASIN: B07H5384J5, 2019.
Main references (sources)	
Recommended books and references (scientific journals, reports...)	Discrete Structures, Logic, and Computability by James L. Hein, Jones & Bartlett Learning; 4 edition, 2015.
Electronic References, Websites	https://www.cs.cornell.edu

Second Year - First Semester

Object Oriented Programming I

Module Information					
Module Title	Object oriented programming I			Module Delivery	
Module Type	Core			Theory ☒ Lecture ☒ Lab ☐ Tutorial ☐ Practical ☐ Seminar	
Module Code					
ECTS Credits	8				
SWL (hr/sem)					
Module Level	2		Semester of Delivery	1	
Administering Department	cs		College	IT	
Module Leader	Name		e-mail	E-mail	
Module Leader's Acad. Title			Module Leader's Qualification		
Module Tutor	Name (if available)		e-mail	E-mail	
Peer Reviewer Name	Name		e-mail	E-mail	
Scientific Committee Approval Date	01/06/2024		Version Number	1.0	
Relation with other Modules					
Prerequisite module	None		Semester		
Co-requisites module	None		Semester		
Module Aims, Learning Outcomes and Indicative Contents					

Module Aims	THIS COURSE WILL PROVIDE A BASIC UNDERSTANDING OF THE METHODS AND TECHNIQUES OF DEVELOPING A SIMPLE TO MODERATELY COMPLEX WEB SITE. USING THE CURRENT STANDARD WEB PAGE LANGUAGE, STUDENTS WILL BE INSTRUCTED ON CREATING AND MAINTAINING A SIMPLE WEB SITE. AFT ER THE FOUNDATION LANGUAGE HAS BEEN ESTABLISHED, THE AID OF AN WEB EDITOR WILL BE INTRODUCED. THIS COURSE WILL PROVIDE A RIGOROUS TREATMENT OF OBJECT-ORIENTED CONCEPTS (DESIGN AND IMPLEMENTATION OF OBJECTS, CLASS CONSTRUCTION AND DESTRUCTION, ENCAPSULATION, INHERITANCE, AND POLYMORPHISM) USING JAVA AS AN EXAMPLE LANGUAGE.
Module Learning Outcomes	DEVELOPMENT OF SOUND PROGRAMMING AND DESIGN SKILLS, PROBLEM SOLVING AND MODELING OF REAL-WORLD PROBLEMS FROM SCIENC E, ENGINEERING, AND ECONOMICS USING THE OBJECT-ORIENTED PARADIGM.
Indicative Contents	Indicative content includes the following. 1 Programming style 2 Basic statements with looping and repetitions 3 One dimensional Arrays 4 Two dimensional Arrays 5 Classes and methods 6 Constructors, Variable types, Overloading 7 UML diagrams 8 Programming by contract: preconditions, postconditions and invariants 9 Exception Handling 10 Polymorphism 11 Encapsulation 12 Inheritance 13 Designing interfaces
Learning and Teaching Strategies	
Strategies	Type something like: The main strategy that will be adopted in delivering this module is to encourage students' participation in the exercises, while at the same time refining and expanding their critical thinking skills. This will be achieved through classes, interactive tutorials and by considering type of simple experiments involving some sampling activities that are interesting to the students.

Student Workload (SWL)			
Structured SWL (h/sem)	102	Structured SWL (h/w)	7
Unstructured SWL (h/sem)	98	Unstructured SWL (h/w)	6.5
Total SWL (h/sem)	200		

Module Evaluation				
	Time/Number	Weight (Marks)	Week Due	Relevant Learning Outcome

Formative assessment	Quizzes	2	10% (10)	5, 10	LO #1, 2, 10 and 11
	Assignments	2	10% (10)	2, 12	LO # 3, 4, 6 and 7
	Projects / Lab.	1	10% (10)	Continuous	
	Report	1	10% (10)	13	LO # 5, 8 and 10
Summative assessment	Midterm Exam	2 hr	10% (10)	7	LO # 1-7
	Final Exam	2hr	50% (50)	16	All
Total assessment			100% (100 Marks)		

Delivery Plan (Weekly Syllabus)	
	Material Covered
Week 1	Programming style
Week 2	Basic statements with looping and repetitions
Week 3	One dimensional Arrays and Two dimensional Arrays
Week 4	Classes and methods
Week 5	Classes and methods
Week 6	Constructors, Variable types
Week 7	Types of constructors
Week 8	Overloading
Week 9	UML diagrams
Week 10	Programming by contract: preconditions
Week 11	Programming by contract: postconditions and invariants
Week 12	Exception Handling
Week 13	Introduction to Inheritance, Encapsulation
Week 14	Polymorphism
Week 15	Designing Interfaces
Week 16	Preparatory week before the final Exam

Delivery Plan (Weekly Lab. Syllabus)	
	Material Covered
Week 1	Lab 1: Programming style,Basic statements with looping and repetitions
Week 2	Lab 2: One dimensional Arrays
Week 3	Lab 3: two dimensional Arrays
Week 4	Lab 4: Classes and methods

Week 5	Lab 5: Constructors, Variable types,, Overloading
Week 6	Lab 6: Programming by contract: preconditions,postconditions and invariants
Week 7	Lab 7: Polymorphism,Encapsulation,Inheritance

Learning and Teaching Resources		
	Text	Available in the Library?
Required Texts	C. Thomas Wu (2010). An Introduction to Object-Oriented Programming with Java. Fifth Edition. McGraw-Hill.	Yes
Recommended Texts	2] Herbert Schildt (2007). Java: The Complete Reference. Seventh Edition. McGraw-Hill.	No
Websites		

Grading Scheme				
Group	Grade		Marks (%)	Definition
Success Group (50- 100)	A- Excellent		90- 100	Outstanding Performance
	B- Very Good		80- 89	Above average with some errors
	C – Good		70- 79	Sound work with notable errors
	D- Satisfactory		60- 69	Fair but with major shortcomings
	E- Sufficient		50- 59	Work meets minimum criteria
Fail Group (0 – 49)	FX – Fail		(45-49)	More work required but credit awarded
	F – Fail		(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.

Probability and Statistics

1. Course Name:	
Probability and Statistics	
2. Course Code:	
201	
3. Semester / Year:	
First semester	
4. Description Preparation Date:	
13\9\2025	
5. Available Attendance Forms:	
6. Number of Credit Hours (Total) / Number of Units (Total)	
125	
7. Course administrator's name (mention all, if more than one name)	
Name: Email:	
8. Email: Course Objectives	
Course Objectives	•
9. Teaching and Learning Strategies	
Strategy	
10. Course Structure	

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
		1. Understand the vocabulary of probability and statistics . 2. Understanding the nature of statistics as an integrated system of knowledge. 3. Developing student's statistical concepts. 4. An attempt to reach the concepts of probability and statistics . The ability to solve complex statistical problems.			

11. Course Evaluation

12. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	Book of probability method
Main references (sources)	The complete reference probability basic
Recommended books and references (scientific journals, reports...)	
Electronic References, Websites	

Second Year - Second Semester

Object Oriented Programming II

Module Information					
Module Title	Object oriented programming II			Module Delivery	
Module Type	Core			Theory ☒ Lecture ☒ Lab ☐ Tutorial ☐ Practical ☐ Seminar	
Module Code					
ECTS Credits	8				
SWL (hr/sem)					
Module Level	2		Semester of Delivery	1	
Administering Department	Cs		College	It	
Module Leader	Name		e-mail	E-mail	
Module Leader's Acad. Title			Module Leader's Qualification		
Module Tutor	Name (if available)		e-mail	E-mail	
Peer Reviewer Name	Name		e-mail	E-mail	
Scientific Committee Approval Date	01/06/2024		Version Number	1.0	
Relation with other Modules					
Prerequisite module	None		Semester		
Co-requisites module	None		Semester		
Module Aims, Learning Outcomes and Indicative Contents					

Module Aims	THIS COURSE WILL PROVIDE A BASIC UNDERSTANDING OF THE METHODS AND TECHNIQUES OF DEVELOPING A SIMPLE TO MODERATELY COMPLEX WEB SITE. USING THE CURRENT STANDARD WEB PAGE LANGUAGE, STUDENTS WILL BE INSTRUCTED ON CREATING AND MAINTAINING A SIMPLE WEB SITE. AFT ER THE FOUNDATION LANGUAGE HAS BEEN ESTABLISHED, THE AID OF AN WEB EDITOR WILL BE INTRODUCED. THIS COURSE WILL PROVIDE A RIGOROUS TREATMENT OF OBJECT-ORIENTED CONCEPTS (DESIGN AND IMPLEMENTATION OF OBJECTS, CLASS CONSTRUCTION AND DESTRUCTION, ENCAPSULATION, INHERITANCE, AND POLYMORPHISM) USING JAVA AS AN EXAMPLE LANGUAGE.
Module Learning Outcomes	Introducing advanced entity programming. ➢ How to use objects within programming as a modern concept and develop students' ability to programmatically ➢ Enhancing the student's ability to think in abstract terms when solving computer science problems and diversity in solution problems in different ways and how to relate them to reality ➢ Addressing advanced new concepts in programming such as multithreading, graphical user interface, and others.
Indicative Contents	Indicative content includes the following. 1-Wrapper classes 2-Inner classes 3-Multithreading 4-Generics 5-GUI design 6-Data base access 7-Distribution

Learning and Teaching Strategies	
Strategies	Type something like: The main strategy that will be adopted in delivering this module is to encourage students' participation in the exercises, while at the same time refining and expanding their critical thinking skills. This will be achieved through classes, interactive tutorials and by considering type of simple experiments involving some sampling activities that are interesting to the students.

Student Workload (SWL)			
Structured SWL (h/sem)	102	Structured SWL (h/w)	7
Unstructured SWL (h/sem)	98	Unstructured SWL (h/w)	6.5
Total SWL (h/sem)	200		

Module Evaluation					
		Time/Number	Weight (Marks)	Week Due	Relevant Learning Outcome
	Quizzes	2	10% (10)	5, 10	LO #1, 2, 10 and 11

Formative assessment	Assignments	2	10% (10)	2, 12	LO # 3, 4, 6 and 7
	Projects / Lab.	1	10% (10)	Continuous	
	Report	1	10% (10)	13	LO # 5, 8 and 10
Summative assessment	Midterm Exam	2 hr	10% (10)	7	LO # 1-7
	Final Exam	2hr	50% (50)	16	All
Total assessment			100% (100 Marks)		

Delivery Plan (Weekly Syllabus)	
	Material Covered
Week 1	Review of OOP Fundamentals
Week 2	Encapsulation and Inheritance
Week 3	Polymorphism and Abstraction
Week 4	Wrapper classes
Week 5	Wrapper classes
Week 6	Inner classes
Week 7	Inner classes
Week 8	Multithreading
Week 9	Multithreading
Week 10	Generics
Week 11	Generics
Week 12	GUI design
Week 13	GUI design
Week 14	Data base access
Week 15	Distribution
Week 16	Preparatory week before the final Exam

Delivery Plan (Weekly Lab. Syllabus)	
	Material Covered
Week 1	Lab 1:Wrapper classes
Week 2	Lab 2: Inner classes
Week 3	Lab 3:-Multithreading
Week 4	Lab 4: Generics
Week 5	Lab 5: GUI design

Week 6	Lab 6: Data base access
Week 7	Lab 7: Distribution

Learning and Teaching Resources		
	Text	Available in the Library?
Required Texts	C. Thomas Wu (2010). An Introduction to Object-Oriented Programming with Java. Fifth Edition. McGraw-Hill.	Yes
Recommended Texts	2] Herbert Schildt (2007). Java: The Complete Reference. Seventh Edition. McGraw-Hill.	No
Websites		

Grading Scheme				
Group	Grade		Marks (%)	Definition
Success Group (50- 100)	A- Excellent		90- 100	Outstanding Performance
	B- Very Good		80- 89	Above average with some errors
	C – Good		70- 79	Sound work with notable errors
	D- Satisfactory		60- 69	Fair but with major shortcomings
	E- Sufficient		50- 59	Work meets minimum criteria
Fail Group (0 – 49)	FX – Fail		(45-49)	More work required but credit awarded
	F – Fail		(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.

Visual Programming

1. Course Name:
Visual Programming
2. Course Code:
3. Semester / Year:
First semester
4. Description Preparation Date:
13 / 9 / 2025
5. Available Attendance Forms:
6. Number of Credit Hours (Total) / Number of Units (Total)
75
7. Course administrator's name (mention all, if more than one name)
Name: Email:
8. Email: Course Objectives

Course Objectives	1. Ability to demonstrate knowledge of interface design principles and be able to apply them in a visual programming environment. 2. The student should have knowledge of Object Oriented Concepts and how to implement them in a visual programming environment.

9. Teaching and Learning Strategies

Strategy	Readings, self-learning, panel discussions. - Classroom exercises and activities. - Guiding students to some websites to benefit from them to develop abilities. - Holding research seminars through which some problems are explained and analyzed and the mechanism for finding solutions. - The main strategy that will be adopted in delivering this module is to encourage students' participation in the exercises, while at the same time refining and expanding their critical thinking skills. - This will be achieved through classes, interactive tutorials and by considering type of simple experiments involving some sampling activities that are interesting to the students

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
		A- Cognitive goals 1.Transforming the vision and path of traditional programming concepts towards visual programming 2.Expanding the student's knowledge from the idea of scattered small programs to an integrated application 3.Expanding the student's			

		knowledge of Object Oriented 4.Expanding the student's knowledge towards programming the use of sound, images and video for presentation requirements B- The soft skills objectives of the course. 1. Developing the student's skills in searching for ideas to present as proposals for discussion to implement simplified projects 2. Developing the student's programming skills through implementing some of the ideas presented and discussed, such as: Programming some games or educational programs in a smooth and useful review manner.			
11. Course Evaluation					

12. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	
Main references (sources)	
Recommended books and references (scientific journals, reports...)	The Complete Reference Visual Basic .NET Programming Visual Basic .NET An Introduction to Programming Using Visual Basic 2012
Electronic References, Websites	

Computation theory

1. Course Name:
Computation theory
2. Course Code:
3. Semester / Year:
second semester
4. Description Preparation Date:
13 / 9 / 2025
5. Available Attendance Forms:
6. Number of Credit Hours (Total) / Number of Units (Total)
125
7. Course administrator's name (mention all, if more than one name)

Name: Email:	
8. Email: Course Objectives	
Course Objectives	The aim of this course is to introduce students to the fundamental area of computer science which enables students to focus on the study of abstract models of computation. These abstract models allow the students to assess via formal reasoning what could be achieved through computing when they are using it to solve problems in science and engineering. The goal is to allow them to answer fundamental questions about problems, such as whether they can or not be computed. The course introduces basic computation models and their properties. The students will be able to express computer science problems as mathematical statements and to formulate proofs.
9. Teaching and Learning Strategies	
Strategy	- Readings, self-learning, panel discussions. - Classroom exercises and activities. - Guiding students to some websites to benefit from them to develop abilities. - Holding research seminars through which some problems are explained and analyzed and the mechanism for finding solutions. Type something like: The main strategy that will be adopted in delivering this module is to encourage students' participation in the exercises, while at the same time refining and expanding their critical thinking skills. This will be achieved through classes, interactive tutorials and by considering type of simple experiments involving some sampling activities that are interesting to the students.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
		• Knowledge and understanding: ❖ Clarifying the basic concepts in computational theory through a			

		set of tools. ❖ Gaining skills in problem-solving. ❖ Acquisition of basic skills as an introduction to building languages. ❖ Acquisition of theoretical concepts to deal with RE's, DFA's, NFA's, Stack's, Turing machines, and Grammars. • Subject-specific skills: ❖ The ability to design (FAs, NFAs, Grammar, languages modelling, small compilers basics). ❖ The ability to think about solving the problem according to specific rules. ❖ Writing scientific reports Know the comparison between (Natural and Formal Languages).			
11.Course Evaluation					
12. Learning and Teaching Resources					
Required textbooks (curricular					

books, if any)	
Main references (sources)	(Michael Sipser), Introduction to the Theory of computation (Third Edition).
Recommended books and references (scientific journals, reports...)	Theory of Computation Simplified, (Varsha H. PatilVaishali S. PawarSwati A. Bhavsar), 2022.
Electronic References, Websites	https://elc.uobasrah.edu.iq/enrol/index.php?id=72

Database Concepts & Design

1. Course Name:	
Database Concepts & Design	
2. Course Code:	
CSITCS209	
3. Semester / Year:	
2024- 2025	
4. Description Preparation Date:	
1/9/2025	
5. Available Attendance Forms:	
Regular attendance	
6. Number of Credit Hours (Total) / Number of Units (Total)	
150/3	
7. Course administrator's name (mention all, if more than one name)	
Name: Iman Mohsen Hassan Email: iman.hassan@uobasrah.edu.iq	
8. Email: Course Objectives	
Course Objectives	• The objective of this course is to introduce students to database management systems. It helps the student to present an actual practical project on realistic interaction and acquisition of skills by collecting information and dealing with a real institution through open discussion with the professor and his fellow students. Topics include 1. Data, Information, and File system 2. Database and database users 3. Database system concepts and architecture 4. Data modeling using the Entity Relationship Diagram (ERD) 5. The relational data model and relational data constraints 6. Functional dependencies and normalization for relational databases 7. The Relational Algebra, 8. Relational database design for ER to relational mapping 9. Organization records in the file 9. Disk storage, basic file structure and hashing, 10. SQL schema definition, constraints, queries and views. 11. Acquisition of skills by using some functions of MSAccess.
9. Teaching and Learning Strategies	
Strategy	The main strategy that will be adopted in delivering this module is to encourage students' participation in the exercises, while at the same time refining and expanding their thinking skills. This will be achieved through classes, Labs. and interactive discussions.

10. Course Structure					
Week	**Hours**	**Required Learning Outcomes**	**Unit or subject name**	**Learning method**	**Evaluation method**
1	5	Introduction to Database	Data, Information, Data Base (DB), Relational Data Base (RDB), Data Base Management System (DBMS)	Theory Lecture Lab	
2	5	Characteristics of Database, Advantages and Disadvantages	Phase1, Phase2, ER Diagram, Main components of ER Diagram, Entities, Entity Attributes, Domain	Theory Lecture Lab	Assignments Lab. (homework)
3	5	Main phases of database design	Primary Key, Foreign Keys, Types of Relation Ships, Phase3, Phase4	Theory Lecture Lab	Assignments Lab. (homework)
4	5	Constructing an ER model	Attributes Types, Single, Multivalued, Compound, Derived, Stored, Key & Optional Attribute.	Theory Lecture Lab	Assignments Lab. (homework)
5	5	ER Diagram Symbols and Notations	Entity, Weak Entity, Attribute, Multivalued Attribute, Derived Attribute, Key Attribute, Relationship.	Theory Lecture Lab	Quizzes
6	5	Cardinality and Ordinality	How to Draw ER Diagrams, ER Diagram Best Practices, Exercises.	Theory Lecture Lab	Assignments Lab. (homework)
7	5	THE RELATIONAL ALGEBRA	Unary Relational Operations: SELECT and PROJECT, Sequences of Operations and the RENAME Operation	Theory Lecture Lab	Assignments Lab. (homework)
8	5	THE RELATIONAL ALGEBRA	Relational Algebra Operations from Set Theory:	Theory Lecture Lab	Assignments Lab. (homework)

			A. UNION, INTERSECTION, and MINUS B. The CARTESIAN PRODUCT (CROSS PRODUCT) Operation		
9	5	THE RELATIONAL ALGEBRA	Binary Relational Operations: JOIN and DIVISION 1. The Join Operation A. Inner join, Variations of JOIN (The EQUIJOIN and NATURAL JOIN)	Theory Lecture Lab	Assignments Lab. (homework)
10	5	THE RELATIONAL ALGEBRA	B. Outer join: Left Outer Join, Right Outer Join Precedence of relational Operations 2. The Division Operation	Theory Lecture Lab	Quizzes
11	5	Files and Records	Records and Record Types, Fixed Length Records, Formatting records of a file of Fixed length records, Variable Length Records, Formatting records of a file of variable-length records (Other options), Formatted a file of records with optional fields, Formatting A repeating field, Formatting file that includes records of different types	Theory Lecture Lab	Assignments Lab. (homework)
12	5	Organizing records in the file Organizing Files on Disk	Record Blocking and Spanned vs Un spanned Records Allocating File Blocks on Disk:		Assignments Midterm Exam

			Contiguous allocation, Linked allocation, Indexed allocation		
13	5	File Headers	, Files of Unordered Records (Heap Files), Files of Ordered Records (Sorted Files)	Theory Lecture Lab	Assignments Lab. (homework)
14	5	Hashing Techniques Hashing Function	Hash table, The idea behind hashing Direct, Subtraction, & Modulo Division Hashing	Theory Lecture Lab	Assignments Lab. (homework)
15	7	Preparatory week before the final Exam		Lecture Lab	

11. Course Evaluation

		Time/Number	Weight (Marks)	Week Due	Relevant Learning Outcome
Formative assessment	Quizzes	2	10% (10)	5, 10	LO #1,2,3,4,5,6,7,8
	Assignments	1	5% (5)	12	LO #1,2,3,4,5,6,7,8
	Assignments Lab.	1	10% (10)	Continuous	
	Midterm Exam	2hr	25% (10)	8,12	LO #12,3,4,5,6,7,8
Summative assessment	Final Exam	3hr	35% (50)	16	All
	Final Lab. Exam	1hr	15%(15)	16	All
Total assessment			100% (100 Marks)		

12. Learning and Teaching Resources

Required textbooks (curricular books, if any)	• Database Concepts 6th Edition, David M. Kroenke,David J. Auer
Main references (sources)	• Database System Concepts Fourth Edition" by Abraham Silberschatz Henry F. Korth S. Sudarshan , McGraw-Hill ISBN 0-07-255481-9

Recommended books and references (scientific journals, reports...)	• Access 2013 the missing manual, Matthew macdonald • FUNDAMENTALS OF Database Systems 6th EDITION, Ramez Elmasri
Electronic References, Websites	https://link.springer.com/book/10.1007/978-3-540-48399-1

Computational Thinking

1. Course Name:	
Computational Thinking	
2. Course Code:	
CT101	
3. Semester / Year:	
1st Year / 1st Semester	
4. Description Preparation Date:	
14/9/2025	
5. Available Attendance Forms:	
In-class / Online	
6. Number of Credit Hours (Total) / Number of Units (Total)	
3 Credit Hours / 2 Units (Theory + Practical)	
7. Course administrator's name (mention all, if more than one name)	
Name: Dr.Murtaja Ali Saare Email:murtaja.sari@uobasrah.edu.iq	
8. Email: Course Objectives	
Course Objectives	• Develop students' ability to apply computational thinking to solve real-world problems. • Introduce problem decomposition, pattern recognition, abstraction, and algorithm design. • Enhance logical reasoning and creativity in problem-solving. • Provide students with practical skills for algorithmic thinking and digital solution design.
9. Teaching and Learning Strategies	
Strategy	• Interactive lectures with discussions.

	• Problem-based learning (PBL). • Hands-on programming labs and exercises. • Group projects and peer collaboration. • Case studies and real-world problem-solving examples.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	2	Understand course goals and CT concepts	Introduction to Computational Thinking	Lecture & Discussion	Quiz
2	2	Apply decomposition to complex problems	Problem Decomposition	Lecture + Exercise	Assignment
3	2	Identify patterns in data/problems	Pattern Recognition	Lecture + Case Study	Quiz
4	2	Apply abstraction to simplify problems	Abstraction	Lecture + Practical	Assignment
5	2	Design step-by-step solutions	Algorithm Design Basics	Lecture + Lab	Assignment
6	2	Understand pseudocode and flowcharts	Representing Algorithms	Lab + Practice	Quiz

7	2	Apply problem-solving strategies in coding	Basic Programming for CT	Lab + Hands-on	Practical Test
8	2	Develop teamwork skills	Group Project (Midterm)	Group Work	Project Evaluation
9	2	Apply CT in science and engineering	CT Applications I	Lecture + Case Study	Quiz
10	2	Apply CT in social sciences and daily life	CT Applications II	Lecture + Discussion	Assignment
11	2	Evaluate algorithms (efficiency & correctness)	Algorithm Evaluation	Lecture + Lab	Practical Test
12	2	Practice iterative problem-solving	Debugging and Refinement	Lab + Peer Work	Assignment
13	2	Apply CT in interdisciplinary projects	CT in Real-World Scenarios	Group Work	Project
14	2	Review and integrate CT skills	Revision & Integration	Seminar	Participation

15	2	Demonstrate full understanding	Final Project Presentation	Project Work	Project & Report

11. Course Evaluation
• Quizzes: 15% • Assignments: 20% • Practical Labs: 15% • Midterm Project: 20% • Final Project & Presentation: 30%

12. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	Wing, J. (2017). Computational Thinking. MIT Press.
Main references (sources)	Selby, C., & Woollard, J. (2013). Computational Thinking: The Developing Definition.
Recommended books and references (scientific journals, reports...)	Denning, P. 009). Beyond Computational Thinking. Communications of the ACM.
Electronic References, Websites	https://www.computationalthinking.org, https://csunplugged.org, https://scratch.mit.edu

Third Year - First Semester

Artificial Intelligence

1. Course Name:

Artificial Intelligence
2. Course Code:
3. Semester / Year:
First / 2024-2025
4. Description Preparation Date:
5. Available Attendance Forms:
Class learning/ E-learning
6. Number of Credit Hours (Total) / Number of Units (Total)
7. Course administrator's name (mention all, if more than one name)
Name: Asmaa Shareef Email: asmaa.shareef@uobasrah.edu.iq
8. Email: Course Objectives

Course Objectives	• Learn how to train a computer to think experimentally in an intelligent way. • Learn how intelligent programming works. • Learn how the Prolog program works.
9. Teaching and Learning Strategies	
Strategy	• Readings, self-study, and discussion groups • Exercises and activities in the classroom and science lab • Directing students to websites for their use • The main strategy used in teaching this unit is to encourage students to participate in exercises, while honing and expanding their critical thinking skills.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
First	4	Introduction to Artificial Intelligence and General Programming Concepts	Introduction to Artificial Intelligence + Introduction to Structured Programming	Theoretical and practical	General questions and discussion
Second	4	Introduction to Artificial Intelligence	AI characteristics, goals, applications, and problems +Definition of variables, data	Theoretical and practical	General questions and discussion

		and General Programming Concepts	types, and variables in the Prolog language		
Third	4	Knowledge Representation and Transaction Analysis in Prolog	The concept of a knowledge base and its representation methods + Logical and mathematical operations	Theoretical and practical	General questions and discussion
Fourth	4	Representing knowledge and applying logical relationships	Studying types of knowledge representation in AI + Examples of logical relational programming	Theoretical and practical	General questions and discussion or exam
Fifth	4	General questions and discussion	Studying theorems in AI + Examples of programming mathematical relationships	Theoretical and practical	General questions and discussion
Sixth	4	Study of theorems and application of mathematical relationships	Applying the theorem to a set of examples + Applying the Prolog language to solve mathematical problems and series	Theoretical and practical	General questions and discussion
Seventh	4	Clarity and some rules of induction	Studying mathematical deduction and induction methods + Applying the Prolog language to solve mathematical problems and sequences	Theoretical and practical	General questions and discussion
Eighth	4	Problem spaces, search methods, and the general structure of lists in Prolog	Blind Search and Mining Search + Introduction to Lists	Theoretical and practical	General questions and discussion
Ninth	4	Blind search and list programming	Depth-first, breadth-first search	Theoretical and practical	General questions and discussion

			+ List programming in Prolog		
Tenth	4	Excavation research and deletion and addition operations in lists	Hill Climbing Search, Best Search First + Programming Add/Delete Operations	Theoretical and practical	General questions and discussion
Eleventh	4	Exploratory research and various programs in the lists	A*+ Encourage branching, specification, and research using different programs using menus.	Theoretical and practical	General questions, discussion and exam
Twelfth	4	Artificial Intelligence Issues and the Concept of Belonging	Solving problems using AI methods + Programming the relationship of belonging	Theoretical and practical	General questions and discussion
Thirteenth	4	Artificial Intelligence Issues and Applications of Belonging Relationships	Solve problems using AI methods + Various programs using member	Theoretical and practical	General questions, discussion and exam
Fourteenth	4	Expert Systems And the Concept of List Merge Relationship Append	Basic concepts and components, building a knowledge hall, and inference techniques + Programming a list merge relationship (append)	Theoretical and practical	General questions and discussion
Fifteenth	4	Expert systems and append applications	Determinants in expert systems and some of their applications + Various programs using append	Theoretical and practical	General questions and monthly exam

11. Course Evaluation	
12. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	Stuart Russel, Peter Norvig, "Artificial Intelligence: A Modern Approach", 3th edition, Prentice-Hall, 2009.
Main references (sources)	1. E. Charniak, D. McDermott, "Introduction to Artificial Intelligence", 4th edition, Addison Wisely, 2000.

	2. Ivan Bratko, "Prolog Programming for Artificial Intelligence", 4th edition, Pearson Education, 2011. 3. George F. Luger, "Artificial Intelligence: Structures and Strategies for Complex Problem Solving", 6th edition, Addison Wesley 2008.
Recommended books and references (scientific journals, reports...)	https://www.journals.elsevier.com/artificial-intelligence
Electronic References, Websites	https://download-internet-pdf-ebooks.com/88-1-library-books

Web Programming I

1. Course Name:

Web Programming I
2. Course Code:
3. Semester / Year:
2ND year
4. Description Preparation Date:
10/9/2025
5. Available Attendance Forms:
Daily Attendance Sheet
6. Number of Credit Hours (Total) / Number of Units (Total):15
7. Course administrator's name (mention all, if more than one name)
Name:Prof.Dr.Raad A. Muhajjar Email:Raad.muhajjar@uobasrah.edu.iq
8. Email: Course Objectives

Course Objectives	The objectives of this program are to: • **Build Foundational Web Development Skills** o Provide students with a strong understanding of HTML, CSS, and JavaScript as the core building blocks of modern web development. • **Enable Practical Application** o Equip students to design, develop, and publish functional, interactive, and visually appealing websites. • **Promote Best Practices** o Develop awareness of web standards, accessibility guidelines, responsive design principles, and coding conventions. • **Foster Problem-Solving and Creativity** o Encourage students to apply logical thinking and creativity in solving design and programming challenges. • **Prepare for Advanced Learning and Careers** o Lay the groundwork for advanced courses in web technologies, software development, and related fields. o Provide transferable skills relevant to the professional world, such as teamwork, critical thinking, and digital literacy.
9. Teaching and Learning Strategies	
Strategy	The *Web Programming I* course adopts a variety of teaching and learning strategies to ensure students develop both theoretical understanding and practical skills: 1. **Lectures (Theory Delivery)** o Provide foundational knowledge of HTML, CSS, and JavaScript. o Use multimedia presentations and live coding demonstrations.

	2. Hands-On Laboratory Sessions o Conduct practical exercises in computer labs to apply lecture concepts. o Guide students through coding tasks, debugging, and small projects. 3. Project-Based Learning (PBL) o Assign individual and group projects (e.g., building a personal portfolio site). o Encourage creativity, problem-solving, and application of best practices. 4. Active and Collaborative Learning o Use pair programming, group discussions, and peer code reviews. o Encourage teamwork and knowledge sharing. 5. E-Learning and Online Resources o Integrate Learning Management Systems (LMS) for assignments, quizzes, and resources. o Provide supplementary tutorials, coding sandboxes (e.g., CodePen, JSFiddle), and video lessons. 6. Formative Assessments and Feedback o Use short quizzes, coding exercises, and in-class activities for continuous evaluation. o Provide timely feedback to help students improve progressively. 7. Self-Directed Learning o Encourage students to explore web development tools, online documentation, and communities. o Promote independent problem-solving and lifelong learning habits. 8. Demonstrations and Case Studies o Showcase real-world websites and applications to highlight best practices. o Analyze case studies of good vs. poor web design and coding practices.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	3	Understand course overview and web development basics	Introduction to Web Programming & Internet Concepts	Lecture + Discussion	Participation
2	3	Describe HTML structure and basic tags	HTML Basics: Elements, Headings, Paragraphs, Links	Lecture + Lab	Quiz + Lab exercises
3	3	Create structured web pages with lists, images, and tables	HTML Lists, Images, Tables	Lecture + Lab	Lab exercises
4	3	Implement forms and input controls	HTML Forms and Input Elements	Lecture + Lab	Lab exercises + Quiz
5	3	Apply CSS styling to HTML elements	CSS Basics: Selectors, Properties, Colors	Lecture + Lab	Lab exercises
6	3	Design page layout using CSS	CSS Box Model, Margins, Padding, Borders	Lecture + Lab	Lab exercises
7	3	Implement advanced CSS	CSS Positioning, Flexbox, Grid	Lecture + Lab	Lab exercises + Quiz

		styling and positioning			
8	3	Apply learned concepts in a small project	Midterm Project: Simple Web Page	Project-Based Learning	Midterm Project Evaluation
9	3	Add interactivity with JavaScript	JavaScript Basics: Variables, Data Types, Operators	Lecture + Lab	Lab exercises
10	3	Control program flow using conditions and loops	JavaScript: Conditionals & Loops	Lecture + Lab	Lab exercises
11	3	Manipulate web page elements dynamically	JavaScript DOM Manipulation	Lecture + Lab	Lab exercises
12	3	Handle events and validate forms	JavaScript Events & Form Validation	Lecture + Lab	Lab exercises + Quiz
13	3	Apply functions and arrays in JavaScript	JavaScript Functions & Arrays	Lecture + Lab	Lab exercises
14	3	Integrate HTML, CSS, and JavaScript in a full project	Capstone Project Development	Project-Based Learning	Project Progress Evaluation
15	3	Present final projects and review all topics	Capstone Project Presentation & Course Review	Presentation + Discussion	Final Project Evaluation + Participation

11. Course Evaluation

- Continuous Assessment: Quizzes and lab exercises are conducted weekly to provide timely feedback and track progress.
- Project-Based Assessment: Both midterm and final projects assess students' ability to integrate theory into practical web development tasks.
- Participation: Students are encouraged to actively engage in labs, discussions, and peer reviews.
- Flexibility: Evaluation methods may be adjusted to suit online or blended learning environments, ensuring fairness and accessibility.

12. Learning and Teaching Resources

Required textbooks (curricular books, if any)	
Main references (sources)	"JavaScript for Modern Web Development: Building a Web Application Using HTML, CSS, and JavaScript" *Publisher:* Skillsoft, 2020 *Overview:* Complete guide for learning web development from basics to building a web application using HTML, CSS, and JavaScript.
Recommended books and references (scientific journals, reports...)	"JavaScript: The Definitive Guide, 7th Edition" *Author:* David Flanagan *Publisher:* O'Reilly Media, 2020 *Overview:* Complete reference for JavaScript covering the latest features and best practic

Electronic References, Websites	W3Schools *Description:* Educational website with interactive tutorials and examples for HTML, CSS, and JavaScript. *Link:* W3Schoos

Third Year - Second Semester

Compiler Constructions

1. Course Name:	
Compiler Constructions	
2. Course Code:	
3. Semester / Year:	
Second semester	
4. Description Preparation Date:	
1 / 9 / 2025	
5. Available Attendance Forms:	
6. Number of Credit Hours (Total) / Number of Units (Total)	
7. Course administrator's name (mention all, if more than one name)	
Name: Dr Adala M. Chyaid Email:	
8. Email: Course Objectives	
Course Objectives	The course illustrates how the theory of language translation introduced in preliminary courses can be applied to the construction of compilers and interpreters. The unit covers building compilers from scratch as well as using compiler generators. In this process, the unit also identifies and explores the key issues involved in compiler design. Constructing a compiler/interpreter for a small language is an essential component of this unit, enabling students to acquire the necessary skills. 1. Understanding the fundamental techniques used in compiler construction, such as lexical analysis, top-down parsing, bottom-up parsing, context-sensitive analysis, and intermediate code generation. 2. Understanding the essential data structures employed in compiler construction, such as abstract syntax trees, symbol tables, three-address code, and stack machines.
9. Teaching and Learning Strategies	
Strategy	Readings, self-learning, panel discussions. - Classroom exercises and activities. - Guiding students to some websites to benefit from them to develop abilities. - Holding research seminars through which some problems are explained and analyzed and the mechanism for finding solutions. - The main strategy that will be adopted in delivering this module is to encourage students' participation in the exercises, while at the same time refining and expanding their critical thinking skills.

	- This will be achieved through classes, interactive tutorials and by considering type of simple experiments involving some sampling activities that are interesting to the students

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
		Knowledge and Understanding 1. Explain the fundamental concepts of compilers and their role in translating high-level programming languages into machine code. 2. Distinguish the main phases of a compiler: Lexical Analysis Syntax Analysis Semantic Analysis Intermediate Code Generation Code Optimization Code Generation 3. Demonstrate knowledge of data structures and algorithms used in each phase (e.g., DFA, Parse Trees, Symbol Tables). **Practical Skills** 4. Design a simple lexical analyzer using tools such as Lex or manual programming. 5. Build a parser using techniques such as LL or LR parsing.			

		6. Implement symbol tables and perform semantic checking. 7. Generate intermediate code from a high-level language program. 8. Apply basic code optimization techniques. Cognitive Skills 9. Analyze a compiled program and identify errors (lexical, syntax, semantic). 10. Compare compilers and interpreters. 11. Evaluate the impact of code optimization techniques on performance and efficiency. General Skills 12. Collaborate in teams to develop parts of a mini-compiler. 13. Document and explain the steps of compiler development. 14. Solve programming problems using compiler design methodologies.			
11. Course Evaluation					

12. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	
Main references (sources)	
Recommended books and references (scientific journals, reports...)	The Complete Reference Visual Basic .NET Programming Visual Basic .NET An Introduction to Programming Using Visual Basic 2012
Electronic References, Websites	

Web Programming II

1. Course Name:

Web Programming II	
2. Course Code:	
3. Semester / Year:	
2ND year	
4. Description Preparation Date:	
12/9/2025	
5. Available Attendance Forms:	
Daily Attendance Sheet	
6. Number of Credit Hours (Total) / Number of Units (Total):15	
7. Course administrator's name (mention all, if more than one name)	
Name:Prof. Dr. Raad A. Muhajjar Email:raad.muhajjar@uobasrah.edu.iq	
8. Email: Course Objectives	
Course Objectives	The objectives of this course are: 1. Understanding PHP Basics: Learn the fundamentals of PHP programming language, including syntax, variables, data types, operators, control structures, and functions. 2. Web Development Concepts: Gain an understanding of web development concepts such as client-server architecture, HTTP protocol, request/response cycle, and the role of PHP in web development. 3. Working with HTML and CSS: Learn how to integrate PHP code within HTML and CSS to create dynamic web pages. Understand how to generate HTML content using PHP and manipulate CSS styles based on dynamic conditions. 4. Handling Form Data: Explore techniques for handling form submissions using PHP. Learn how to retrieve form data, validate and sanitize input, and perform server-side form processing. 5. Working with Databases: Understand the basics of database management systems and how to interact with databases using PHP. Learn how to establish database connections, execute SQL queries, and handle result sets. 6. Session and Cookies Management: Explore techniques for managing user sessions and cookies using PHP. Learn how to create, store, and retrieve session data, as well as how to implement user authentication and authorization. 7. File Handling: Gain knowledge on file handling operations in PHP, such as reading from and writing to files, uploading files, and manipulating file metadata.

9. Teaching and Learning Strategies	
Strategy	The *Web ProgrammingII* course adopts a variety of teaching and learning strategies to ensure students develop both theoretical understanding and practical skills: 1. **Lectures (Theory Delivery)** o Provide foundational knowledge of PHP, and DataBase. o Use multimedia presentations and live coding demonstrations. 2. **Hands-On Laboratory Sessions** o Conduct practical exercises in computer labs to apply lecture concepts. o Guide students through coding tasks, debugging, and small projects. 3. **Project-Based Learning (PBL)** o Assign individual and group projects (e.g., building a personal portfolio site). o Encourage creativity, problem-solving, and application of best practices. 4. **Active and Collaborative Learning** o Use pair programming, group discussions, and peer code reviews. o Encourage teamwork and knowledge sharing. 5. **E-Learning and Online Resources** o Integrate Learning Management Systems (LMS) for assignments, quizzes, and resources. o Provide supplementary tutorials, coding sandboxes (e.g., CodePen, JSFiddle), and video lessons. 6. **Formative Assessments and Feedback** o Use short quizzes, coding exercises, and in-class activities for continuous evaluation. o Provide timely feedback to help students improve progressively. 7. **Self-Directed Learning** o Encourage students to explore web development tools, online documentation, and communities. o Promote independent problem-solving and lifelong learning habits. 8. **Demonstrations and Case Studies** o Showcase real-world websites and applications to highlight best practices. o Analyze case studies of good vs. poor web design and coding practices.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	3	Understand PHP syntax, variables, data types, and operators	Introduction to PHP	Lecture + Hands-on coding	Short quiz + coding exercises
2	3	Apply control structures, loops, and functions in PHP	Introduction to PHP	Lecture + Lab work	Lab assignment
3	3	Explain client-server architecture and HTTP protocol	Web Development Basics	Lecture + Discussion	Quiz
4	3	Demonstrate request/response cycle, HTML & CSS basics, integrate	Web Development Basics	Hands-on coding + Demo	Practical exercise

		PHP with HTML/CSS			
5	3	Create HTML forms and handle submissions with PHP	Form Handling and Validation	Lab work	Coding assignment
6	3	Validate and sanitize user input, display form errors	Form Handling and Validation	Lecture + Lab	Lab test
7	3	Explain relational databases and establish DB connection with PHP	Database Interaction	Lecture + Lab practice	Quiz + coding exercise
8	3	Execute SQL queries and retrieve results using PHP	Database Interaction	Hands-on lab	Coding project
9	3	Understand sessions, cookies, and manage user sessions	Session Management & Authentication	Lecture + Lab	Quiz + coding demo
10	3	Implement authentication, authorization, and secure session handling	Session Management & Authentication	Case study + Lab	Coding project
11	3	Perform file reading/writing, handle file uploads and validation	File Handling and Uploading	Lab work	Practical exercise
12	3	Manipulate file metadata, directory handling	File Handling and Uploading	Lecture + Lab	Coding assignment
13	3	Use APIs in PHP, make API requests	Working with APIs	Lecture + Demo	Quiz
14	3	Parse API responses (JSON/XML), integrate external APIs	Working with APIs	Lab work	Coding project
15	3	Present group project and reflect on learning outcomes	Project Presentations & Wrap-up	Group work + Discussion	Group presentation

11. Course Evaluation

- Continuous Assessment: Quizzes and lab exercises are conducted weekly to provide timely feedback and track progress.
- Project-Based Assessment: Both midterm and final projects assess students' ability to integrate theory into practical web development tasks.

• Participation: Students are encouraged to actively engage in labs, discussions, and peer reviews. Flexibility: Evaluation methods may be adjusted to suit online or blended learning environments, ensuring fairness and accessibility.
• 12. Learning and Teaching Resources
Required textbooks (curricular books, if any)

Main references (sources)	Welling, L., & Thomson, L. (2017). PHP and MySQL Web Development (5th ed.). Addison-Wesley.
Recommended books and references (scientific journals, reports...)	Freeman, E., & Robson, E. (2020). Head First HTML and CSS (2nd ed.). O'Reilly.
Electronic References, Websites	W3Schools *Description:* Educational website with interactive tutorials and examples for HTML, CSS, and JavaScript. *Link:* W3Schoos

Operations Research

Module Information

Module Title	Operations Research	Module Delivery	
Module Type	Core	☒ Theory ☒ Lecture ☐ Lab ☐ Tutorial ☐ Practical ☐ Seminar	
Module Code	UoB12345		
ECTS Credits	8		
SWL (hr/sem)	200		

Module Level		Semester of Delivery	
Administering Department	Type Dept. Code	College	Type College Code
Module Leader	Name	e-mail	Nasir.jasim@uobasrah.edu.iq
Module Leader's Acad. Title	Lecturer	Module Leader's Qualification	Ph.D.
Module Tutor	Name (if available)	e-mail	E-mail
Peer Reviewer Name	Name	e-mail	E-mail
Scientific Committee Approval Date	13/09/2025	Version Number	1.0

Relation with other Modules

Prerequisite module	None	Semester	
Co-requisites module	None	Semester	

Module Aims, Learning Outcomes and Indicative Contents

Module Objectives	1. Modelling realistic problems with different mathematical formulas. 2. Finding a solution to any problem available in the labor market after modelling it using different methods of solution. 3. Searching for the best solution to the problem and searching for the best method used to deliver the product to the labor market.
Module Learning Outcomes	Cognitive goals 1. Enable the student to identify problems in the labor market. 2. The student's ability to model realistic problems. 3. Enabling the student to solve any problem he encounters in the labor market by converting it into a mathematical model and solving it using one of the solutions. Skill objectives for the course 1. Work as a member of a team to solve any problem in the market. 2. Understanding mathematics through practice
Indicative Contents	Indicative content includes the following. Part A – Linear Programming Constructing Linear Programming Models, Forms of Linear programming model, The formulation of linear programming Model, Method of solution of Linear programming Model. [8hrs]

	Part B- Method of solution of Linear programming Model Graphical method, Simplex Method. [8 hrs] Part C- Artificial Variable Technique, Duality in Linear Programming Two Phase Method, Duality and Simplex Method [9 hrs] Part D – Transportation Problems Method for Initial Basic Feasible Solution to a transportation problem, North-West Corner Rule, Least Cost Method, Vogel's Approximation Method, Testing the initial basic feasible solution and obtaining the optimal solution, Stepping Stone Method, Modified Distribution method. [10 hrs] Part E – Assignment Problems [6 hrs]
Learning and Teaching Strategies	
Strategies	Providing distinguished educational and research services that keep pace with local and international quality standards in the fields of computer and informatics. These services allow for preparing a distinguished, competitive graduate. In addition to that, the completion of high-end scientific research and effective participation in community service are key to building a knowledge-based economy.

Student Workload (SWL)			
Structured SWL (h/sem)	102	**Structured SWL (h/w)**	7
Unstructured SWL (h/sem)	98	**Unstructured SWL (h/w)**	6.5
Total SWL (h/sem)	200		

Module Evaluation

		Time/Number	Weight (Marks)	Week Due	Relevant Learning Outcome
Formative assessment	Quizzes	2	10% (10)	5 and 10	LO #1, #2 and #10, #11
	Assignments	2	10% (10)	2 and 12	LO #3, #4 and #6, #7
	Projects / Lab.	1	10% (10)	Continuous	All
	Report	1	10% (10)	13	LO #5, #8 and #10
Summative assessment	Midterm Exam	2hr	10% (10)	7	LO #1- #7
	Final Exam	3hr	50% (50)	16	All
Total assessment			100% (100 Marks)		

Delivery Plan (Weekly Syllabus)

	Material Covered
Week 1	Introduction – Linear programming Models, Forms of Linear programming Models

Week 2	Application Examples, Graphical Methods for Solving Linear Programming Models
Week 3	Simplex Method
Week 4	Solving Linear Programming Problems by the Simplex Method
Week 5	Artificial Variable Technique
Week 6	Duality in Linear Programming Problem
Week 7	Duality and Simplex Method
Week 8	Assignment 1
Week 9	Transportation Problems
Week 10	Initial Basic Feasible Solution of Transportation Problems
Week 11	Optimal Solution of Linear Programming Problems
Week 12	Unbalanced Transportation Problem
Week 13	Assignment 2
Week 14	Assignment Problems
Week 15	**The Hungarian Method for Assignment Problem**
Week 16	**Preparatory week before the final Exam**

Delivery Plan (Weekly Lab. Syllabus)	
	Material Covered
Week 1	
Week 2	
Week 3	
Week 4	
Week 5	
Week 6	
Week 7	

Learning and Teaching Resources		
	Text	**Available in the Library?**
Required Texts	Makebest Decisions Through Operations Research, S.D.SHARMA	Yes
Recommended Texts	Prem Kumar Gupta, D.S. HIRA, S.CHAND بحوث العمليات ((مفهوما وتطبيقا) تأليف الدكتور حامد سعد نور الشمرتي	Yes
Websites		

Grading Scheme

Group	Grade		Marks %	Definition
Success Group (50- 100)	A- Excellent		90- 100	Outstanding Performance
	B- Very Good		80- 89	Above average with some errors
	C- Good		70- 79	Sound work with notable errors
	D- Satisfactory		60- 69	Fair but with major shortcomings
	E- Sufficient		50- 59	Work meets minimum criteria
Fail Group (0 – 49)	FX – Fail		(45-49)	More work required but credit awarded
	F – Fail		(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.

Computer Organization and Architecture

Module Information

Module Title	Computer Organization and Architecture			Module Delivery	
Module Type	Core			☒ Theory ☒ Lecture ☒ Lab ☐ Tutorial ☐ Practical ☐ Seminar	
Module Code	UoB12345				
ECTS Credits	5				
SWL (hr/sem)	125				
Module Level		2	Semester of Delivery		1
Administering Department		Type Dept. Code	College	Type College Code	
Module Leader	Name		e-mail	E-mail	
Module Leader's Acad. Title		Professor	Module Leader's Qualification		Ph.D.
Module Tutor	Name (if available)		e-mail	E-mail	
Peer Reviewer Name		Name	e-mail	E-mail	
Scientific Committee Approval Date		01/06/2023	Version Number		1.0

Relation with other Modules			
Prerequisite module	None	Semester	
Co-requisites module	None	Semester	

Module Aims, Learning Outcomes and Indicative Contents	
Module Aims	Here are some module aims typically associated with a Computer Organization & Architecture course. These aims describe the overarching goals and objectives of the course: 1. To provide students with a solid understanding of the fundamental concepts and principles of computer organization and architecture. 2. To introduce students to the components and operation of a computer system, including the CPU, memory, and I/O subsystems. 3. To familiarize students with the Von Neumann architecture and its role in modern computer systems. 4. To develop students' understanding of digital logic and Boolean algebra, enabling them to design and analyze combinational and sequential logic circuits. 5. To introduce students to different number systems and their representations in digital systems. 6. To explore the principles of data representation and arithmetic, including signed number representations and arithmetic operations. 7. To introduce students to the concepts and techniques of instruction-level parallelism and pipelining. 8. To enable students to analyze and resolve hazards and dependencies in pipelined architectures. 9. To provide students with a comprehensive understanding of memory systems, including cache memory organization and virtual memory concepts. 10. To introduce students to I/O systems, interfaces, and programming techniques. 11. To familiarize students with microprocessor architecture and programming, including instruction set architecture (ISA) and assembly language programming.

	12. To develop students' ability to evaluate and optimize the performance of computer systems. 13. To introduce students to parallel processing and multicore architectures, including the principles of cache coherence and synchronization. 14. To explore emerging trends and technologies in computer organization and architecture, such as quantum computing and cloud computing. This module aims to provide a broad overview of the goals and objectives of a Computer Organization & Architecture course.
Module Learning Outcomes	Here are some module learning outcomes that are typically associated with a Computer Organization & Architecture course. These outcomes represent the knowledge, skills, and competencies that students are expected to achieve upon completing the course: 1. Understand the fundamental components and principles of computer organization and architecture. 2. Demonstrate knowledge of the Von Neumann architecture and its components. 3. Explain the instruction execution cycle and the role of the CPU. 4. Analyze and design combinational and sequential logic circuits. 5. Demonstrate an understanding of number systems and their representations in digital systems. 6. Explain the principles of data representation and arithmetic operations. 7. Understand the concepts and techniques of instruction-level parallelism and pipelining. 8. Analyze and resolve hazards and dependencies in pipelined architectures. 9. Describe the organization and hierarchy of memory systems, including cache memory. 10. Understand virtual memory concepts and address translation mechanisms. 11. Explain I/O systems, interfaces, and programming techniques. 12. Understand the principles of microprocessor architecture and programming. 13. Analyze and evaluate the performance of computer systems. 14. Understand the principles and techniques of parallel processing and multicore architectures. 15. Identify and discuss emerging trends and technologies in computer organization and architecture. These module learning outcomes reflect the core knowledge and skills that students are expected to gain throughout the course. 9.
Indicative Contents	Here are some indicative contents for a Computer Organization & Architecture course targeted at beginners. These contents cover the fundamental concepts and topics typically included in such a course: 1. Introduction to Computer Systems • Overview of computer organization and architecture • Basic components of a computer system • Von Neumann architecture and its principles 2. Number Systems and Digital Logic • Binary, decimal, and hexadecimal number systems • Boolean algebra and logic gates • Combinational and sequential logic circuits 3. Data Representation • Binary representation of integers and characters • Signed number representation (sign-magnitude, one's complement, two's complement)

	• Floating-point representation 4. Central Processing Unit (CPU) • CPU components and organization • Instruction execution cycle • CPU performance and factors affecting it. 5. Memory Systems • Memory hierarchy and its importance • Primary memory (RAM, ROM) and secondary storage (hard drives, solid-state drives) • Caches and cache organization 6. Instruction Set Architecture (ISA) • Overview of instruction sets and their formats. • Addressing modes and instruction types • Instruction decoding and execution. 7. Input/Output Systems • I/O devices and interfaces • I/O communication methods (programmed I/O, interrupt driven. I/O, DMA) • I/O performance and bottlenecks 8. Processor Design and Organization • Basic CPU design principles (fetch-decode-execute cycle) • Instruction pipelining and hazards. • Control unit and microprogramming 9. Computer Arithmetic • Binary arithmetic operations (addition, subtraction, multiplication, division) • Fixed-point and floating-point arithmetic • Arithmetic logic unit (ALU) design 10. Introduction to Assembly Language Programming • Basics of assembly language programming • Instruction syntax and addressing modes. • Simple assembly programs and debugging 11. Introduction to Parallel Processing • Concepts of parallel processing and its importance • Flynn's taxonomy (SISD, SIMD, MISD, MIMD) • Multicore processors and their organization 12. Emerging Trends in Computer Architecture • Introduction to emerging technologies (quantum computing, neuromorphic computing) • Cloud computing and virtualization • Energy-efficient computing and green computing concepts These indicative contents provide beginners with a solid foundation in computer organization and architecture.
Learning and Teaching Strategies	
Strategies	When teaching a Computer Organization & Architecture course to beginners, it's important to adopt strategies that cater to their foundational understanding and gradually build their knowledge and skills. Here are some effective learning and teaching strategies for beginners in a Computer Organization & Architecture course: 1. Visual Aids and Analogies: Use visual aids such as diagrams, charts, and illustrations to simplify complex concepts. Analogies comparing computer

components to familiar real-world objects can make abstract ideas more relatable and easier to understand.

2. Step-by-Step Approach: Break down complex topics into smaller, manageable steps. Present the material in a sequential manner, building upon previously covered concepts. This helps beginners grasp the fundamentals before moving on to more advanced topics.
3. Direct Activities: Provide firsthand activities that allow beginners to interact with hardware components or simulation software. This can include assembling simple computer systems, performing basic circuit simulations, or writing simple programs. Direct activities reinforce learning and make abstract concepts more tangible.
4. Practical Examples: Use practical examples and real-life scenarios to demonstrate the relevance and application of the concepts being taught. Relate the material to everyday situations or commonly used technologies to help beginners connect theory to practice.
5. Scaffolding: Provide scaffolding support by gradually reducing assistance as students gain confidence and proficiency. Start with guided exercises and gradually increase the level of complexity and autonomy. This helps beginners develop their critical thinking skills and independent thinking.
6. Interactive Discussions: Encourage interactive discussions to promote active engagement and peer learning. Beginners can ask questions, share their perspectives, and learn from their classmates' experiences. This fosters a supportive learning environment where beginners can build their understanding collaboratively.
7. Concept Mapping and Summarizing: Encourage beginners to create concept maps or summaries of the material covered. Concept maps visually organize the relationships between different concepts, while summaries help reinforce understanding and retention.
8. Concrete Examples: Use concrete examples and familiar scenarios to explain abstract concepts. Relate computer organization and architecture to everyday experiences, such as explaining how a CPU functions like the brain of a computer or how cache memory is like a high-speed storage closet.
9. Incremental Assessments: Break assessments into smaller, incremental tasks to evaluate and reinforce learning along the way. This can include quizzes, short assignments, or mini projects that gradually increase in complexity as beginners progress through the course.
10. Encourage Questions: Create a supportive environment that encourages beginners to ask questions without hesitation. Answer questions patiently and provide explanations in a clear and accessible manner. This helps beginners clarify their doubts and deepen their understanding.
11. Provide Additional Resources: Offer supplementary resources, such as textbooks, online tutorials, and reference materials, to support beginners' learning outside the classroom. These resources can provide alternative explanations, additional examples, and further practice opportunities.
12. Regular Feedback and Guidance: Provide timely and constructive feedback on assignments and assessments to guide beginners' progress. Highlight their strengths and provide specific suggestions for improvement to help them grow and build confidence.

By employing these strategies, you can create an inclusive and supportive learning environment for beginners in a Computer Organization & Architecture course. Adjust

	the pace and depth of the course to accommodate their learning needs and gradually build their knowledge and skills in the subject.

Student Workload (SWL)			
Structured SWL (h/sem)	45	Structured SWL (h/w)	
Unstructured SWL (h/sem)	80	Unstructured SWL (h/w)	
Total SWL (h/sem)	125		

Module Evaluation					
		Time/Number	Weight (Marks)	Week Due	Relevant Learning Outcome
Formative assessment	Quizzes	2	10% (10)	5, 10	LO #1, 2, 10 and 11
	Assignments	2	10% (10)	2, 12	LO # 3, 4, 6 and 7
	Projects / Lab.	1	10% (10)	Continuous	
	Report	1	10% (10)	13	LO # 5, 8 and 10
Summative assessment	Midterm Exam	2 hr	10% (10)	7	LO # 1-7
	Final Exam	2 hr	50% (50)	16	All
Total assessment			100% (100 Marks)		

Delivery Plan (Weekly Syllabus)	
	Material Covered
Week 1	Introduction to Computer Systems • Basic components of a computer system • Overview of computer architecture and organization
Week 2	Number Systems and Digital Logic • Binary, decimal, and hexadecimal number systems • Logic gates and Boolean algebra • Combinational and sequential logic circuits
Week 3	Basic Computer Organization • Von Neumann architecture • CPU, memory, and I/O subsystems • Instruction execution cycle
Week 4	Machine Language and Assembly Programming • Machine language instructions • Assembly language programming concepts • Introduction to an assembly language (e.g., MIPS, x86)
Week 5	Central Processing Unit (CPU) Design • CPU components and their functions • Instruction set architecture (ISA) • CPU Datapath and control unit
Week 6	1. Memory Systems • Memory hierarchy • Cache memory organization and mapping techniques • Virtual memory concepts

Week 7	Mid-term Exam
Week 8	Microprocessors and Microcontrollers • Introduction to microprocessors and microcontrollers • Architecture and features of popular microprocessors (e.g., Intel 8086, ARM Cortex-M)
Week 9	Instruction Set Architecture (ISA) • Types of instruction formats • Addressing modes • Assembly language programming for the chosen ISA
Week 10	Input/Output Systems • I/O interfaces and devices • Interrupts and DMA (Direct Memory Access) • I/O programming techniques
Week 11	Computer Arithmetic • Binary and hexadecimal arithmetic • Integer and floating-point representations • Arithmetic operations and algorithms
Week 12	Pipelining and Superscalar Techniques • Pipelined CPU architecture • Instruction pipelining and hazards. • Superscalar and out-of-order execution
Week 13	Advanced Topics in Computer Architecture • Parallel processing and multiprocessors
Week 14	Advanced Topics in Computer Architecture • Memory management and protection • Performance evaluation and optimization techniques
Week 15	General Discussion
Week 16	Preparatory week before the final Exam

Delivery Plan (Weekly Lab. Syllabus)	
	Material Covered
Week 1	Lab 1:
Week 2	Lab 2:
Week 3	Lab 3:
Week 4	Lab 4:
Week 5	Lab 5:
Week 6	Lab 6:
Week 7	Lab 7:

Learning and Teaching Resources		
	Text	Available in the Library?
Required Texts	"Computer Organization and Architecture: Designing for Performance" by William Stallings:	

	➢ This textbook provides a comprehensive introduction to computer organization and architecture, with a focus on performance design principles. It covers topics such as CPU organization, memory hierarchy, instruction set architecture, and I/O systems. The book includes numerous examples, illustrations, and exercises to reinforce concepts.	
Recommended Texts	"Structured Computer Organization" by Andrew S. Tanenbaum and Todd Austin: ➢ This book provides a structured approach to computer organization and architecture. It covers fundamental concepts, including digital logic, data representation, CPU organization, memory systems, and I/O systems. The text emphasizes the importance of hierarchical organization in computer systems and includes numerous examples and exercises to reinforce learning.	
Websites		

Grading Scheme			
Group	Grade	Marks (%)	Definition
Success Group (50- 100)	A- Excellent	90- 100	Outstanding Performance
	B- Very Good	80- 89	Above average with some errors
	C- Good	70- 79	Sound work with notable errors
	D- Satisfactory	60- 69	Fair but with major shortcomings
	E- Sufficient	50- 59	Work meets minimum criteria
Fail Group (0 – 49)	FX – Fail	(45-49)	More work required but credit awarded
	F – Fail	(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.

Fourth Year - First Semester

Mobile Applications Programming

1. Course Name:
Mobile Applications Programming
2. Course Code:
3. Semester / Year:
First semester
4. Description Preparation Date:
13/9/2025
5. Available Attendance Forms:
6. Number of Credit Hours (Total) / Number of Units (Total)
7. Course administrator's name (mention all, if more than one name)
Name: Dr. Shatha Falih Hendi Email: shatha,falih@uobasrah,edu.iq
8. Email: Course Objectives

Course Objectives	1. Learn about different smartphone platforms and the structure and framework of the phone. 2. Learn programming and mobile app development. 3. Learn how to upload an app to the store to make it available for use. 4. Learn how to monetize apps.
9. Teaching and Learning Strategies	
Strategy	1- Monthly exams 2- Daily exams 3- Reports 4- Projects through which the student applies what they have learned from the course 5- Daily lecture participation
10. Course Structure	

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
		1- The blackboard 2- A display screen to display the lecture using a presentation application 3- Showing short clips of films related to the topics being covered			

		4- Motivating students by asking questions and opening up the possibility of diverse answers from everyone to create interactive discussions during the lecture 5- Making students feel the importance of studying the material - Encouraging team spirit among them			

11. Course Evaluation

12. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	
Main references (sources)	1-Valentino Lee, Heather Schneider, and Robbie Schell, *Mobile Applications: Architecture,* *Design, and Development*, Prentice Hall, 2004. 2- Brian Fling, *Mobile Design and Development*, O'Reilly Media, 2009. 3- Maximiliano Firtman, *Programming the Mobile Web*, O'Reilly Media, 2010. 4- Christian Crumlish and Erin Malone, Designing Social Interfaces, O'Reilly Media, 2009.
Recommended books and references (scientific journals, reports...)	1-Valentino Lee, Heather Schneider, and Robbie Schell, *Mobile Applications: Architecture,* *Design, and Development*, Prentice Hall, 2004. 2- Brian Fling, *Mobile Design and Development*, O'Reilly Media, 2009. 3- Maximiliano Firtman, *Programming the Mobile Web*, O'Reilly Media, 2010. 4- Christian Crumlish and Erin Malone, Designing Social Interfaces, O'Reilly Media, 2009.
Electronic References, Websites	1- https://developer.android.com/training/index.html 2- www.coders-hub.com 3- http://www.javacodegeeks.com/category/android/ 4- www.wrox.com

Computational Intelligence

1. Course Name:
Computational Intelligence
2. Course Code:

3. Semester / Year:
First semester
4. Description Preparation Date:
13 / 9 / 2025
5. Available Attendance Forms:
6. Number of Credit Hours (Total) / Number of Units (Total)
75
7. Course administrator's name (mention all, if more than one name)
Name: Email:
8. Email: Course Objectives

Course Objectives	The course is a research-based course and therefore focuses on leading students to investigate the current state of research in Computational Intelligent areas as well as to gain comprehensive theoretical knowledge from scientific research about the basic concepts and features of CI methodologies and approaches.

9. Teaching and Learning Strategies

Strategy	The course is a research-based course and therefore focuses on leading students to investigate the current state of research in Computational Intelligent areas as well as to gain comprehensive theoretical knowledge from scientific research about the basic concepts and features of CI methodologies and approaches.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
		Provide the student with key vocabulary and help to understand artificial intelligence and Computational intelligence by understand: Optimization Constrained, unconstrained optimization Parameter space, function space, and fitness space Local			

		and global optima Multi-objective optimization Classification / Learning Classification (Supervised Learning) Clustering (Unsupervised Learning) Reinforcement Learning Control Systems			

11. Course Evaluation	
12. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	1. James M. Keller et al.," Fundamentals of Computational Intelligence: Neural Networks, Fuzzy Systems, and Evolutionary Computation", Wiley-IEEE Press, 2016. 2. Jiangjun Tang et al. "Simulation and Computational Red Teaming for Problem Solving", ch12: Computational Intelligence, Wiley-IEEE Press, pp. 219 – 240, 2020. 3. Jan Peters, "Computational Intelligence: Principles, Techniques and Applications", Computer Journal, 2007. 4. Mircea Eremia et al.," Advanced Solutions in Power Systems: HVDC, FACTS, and Artificial Intelligence', ch17: Fuzzy Systems, Wiley-IEEE Press, pp. 785 - 818, 2016.
Main references (sources)	
Recommended books and references (scientific journals, reports...)	
Electronic References, Websites	

Fourth Year - Second Semester

Computer Security

1. Course Name:
Computer Security

2. Course Code:
3. Semester / Year:
Second semester
4. Description Preparation Date:
13 / 9 / 2025
5. Available Attendance Forms:
6. Number of Credit Hours (Total) / Number of Units (Total)
75
7. Course administrator's name (mention all, if more than one name)
Name: Email:
8. Email: Course Objectives

Course Objectives	This course provides students with the most common cryptographic algorithms and protocols and how to use cryptographic algorithms and protocols to secure distributed applications and computer networks: - Explain the objectives of information security. - Explain the importance and application of each of confidentiality, integrity, authentication and availability. - Understand various cryptographic algorithms. - Understand the basic categories of threats to computers and networks.

9. Teaching and Learning Strategies

Strategy	Type something like: The main strategy that will be adopted in delivering this module is to encourage students' participation in the exercises, while at the same time refining and expanding their critical thinking skills. This will be achieved through classes, interactive tutorials and by considering types of simple experiments involving some sampling activities that are interesting to the students.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
		By the end of the course, students will be able to:			

		Understand the Cryptography principles and types. Describe the computer systems security issues. Student will be able to understand basic cryptographic algorithms, message and security issues. Ability to identify information system requirements for both of them, such as, client and server. Ability to understand the current issues towards information security. Apply security principles to system design.			

11. Course Evaluation

12. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	William Stallings, "Cryptography and Network Security. Principle and Practice", Fourth Edition, Principle Hall, USA, 2006.
Main references (sources)	Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, "Handbook of Applied Cryptography", Fifth Edition , CRC Press, 2001.
Recommended books and references (scientific journals, reports...)	
Electronic References, Websites	

Knowledge Engineering

1. Course Name:
Knowledge Engineering

2. Course Code:
3. Semester / Year:
Second semester
4. Description Preparation Date:
14 / 9 / 2025
5. Available Attendance Forms:
Attendance / Blended Learning (In-person, Online)
6. Number of Credit Hours (Total) / Number of Units (Total)
75 Hour
7. Course administrator's name (mention all, if more than one name)
Name: Lec. Suhaib Abdulatif Abdulqader **Email**: Suhaib.alansarry@uobasrah.edu.iq
8. Email: Course Objectives

Course Objectives	- Ability to demonstrate knowledge of the principles of Knowledge Engineering and its applications. - Students should have an understanding of Artificial Intelligence concepts in a practical environment supported by applications.
9. Teaching and Learning Strategies	
Strategy	- Readings, self-learning, and discussion sessions. - In-class exercises and activities. - Directing students to selected websites to enhance their skills. - Organizing research seminars to explain and analyze problems and explore solution methods. - The main strategy in teaching this unit is to encourage students to participate in exercises, while developing and expanding their critical thinking skills. - This will be achieved through classroom sessions, interactive lessons, and studying simple experiments that include sample activities relevant to students' interests.
10. Course Structure	

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
		- Define the concept of knowledge, its types, and how it is represented in intelligent systems. - Explain the difference between data, information, and knowledge.			

		- Acquire knowledge from experts and different sources. - Apply knowledge representation techniques such as rules, semantic networks, frames, and ontologies. - Demonstrate reasoning methods such as forward and backward chaining. - Build simple expert systems. - Identify issues of consistency, ambiguity, and complexity in knowledge representation. - Use software tools related to knowledge engineering. - Analyze real-world problems and select suitable knowledge representation methods. - Develop teamwork skills through projects in designing knowledge-based systems.			

11. Course Evaluation	
12. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	
Main references (sources)	
Recommended books and references (scientific journals, reports...)	Kendal, Simon L., and Malcolm Creen. *An introduction to knowledge engineering*. London: Springer London, 2007.
Electronic References, Websites	

Communication Skills

1. Course Name:
Communication Skills

2. Course Code:
3. Semester / Year:
Fourth Year
4. Description Preparation Date:
14/9/2025
5. Available Attendance Forms:
In-person & Online
6. Number of Credit Hours (Total) / Number of Units (Total)
3 Credit Hours / 2 Units
7. Course administrator's name (mention all, if more than one name)
Name: Dr.Murtaja Ali Saare Email:murtaja.sari@uobasrah.edu.iq
8. Email: Course Objectives

Course Objectives	- Develop students' oral and written communication skills for academic and professional contexts. - Enhance listening and speaking abilities through interactive discussions and presentations. - Foster teamwork and collaboration skills in group settings. - Build confidence in public speaking and argumentation. - Introduce students to digital tools for effective communication.
9. Teaching and Learning Strategies	
Strategy	- Lectures and interactive discussions - Group projects and collaborative tasks - Case studies and role-playing - Class presentations and debates - Use of multimedia and ICT platforms

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	3	Understand basics of communication	Introduction to Communication	Lecture + Discussion	Quiz
2–3	6	Apply verbal/non-verbal communication skills	Verbal & Non-Verbal Communication	Role Play + Practice	Participation + Assignment

4–5	6	Develop effective writing skills	Academic & Professional Writing	Writing Workshops	Written Assignment
6–7	6	Practice oral communication	Public Speaking & Presentations	Presentations	Oral Presentation
8	3	Demonstrate listening & feedback	Active Listening	Interactive Activities	Participation
9–10	6	Collaborate in groups effectively	Teamwork & Group Dynamics	Group Projects	Group Report + Peer Review
11	3	Use ICT for communication	Digital Communication Tools	Lab/Workshop	Practical Task
12–13	6	Apply skills in real context	Case Studies & Simulations	Case Studies	Assignment
14	3	Review & Integration	Course Review	Seminar	Final Exam

11. Course Evaluation
- Class Participation & Attendance: 10% - Assignments & Reports: 20% - Presentations & Oral Activities: 20% - Group Project: 20% - Final Exam: 30%

12. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	Adler, R. B., & Elmhorst, J. M. (2019). Communicating at Work: Strategies for Success in Business and the Professions. McGraw-Hill.
Main references (sources)	Lucas, S. E. (2020). The Art of Public Speaking. McGraw-Hill.
Recommended books and references (scientific journals, reports...)	Journals on communication studies and applied linguistics; Reports and case studies from professional organizations (e.g., Toastmasters International).
Electronic References, Websites	www.communicationtoday.com www.toastmasters.org www.coursera.org/communication