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

2026

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.

This guide, in its second version, includes a description of the academic program after updating the subjects and paragraphs of the previous guide in light of the updates and developments of the educational system in Iraq, which included the description of the academic program in its traditional form (annual, quarterly), as well as the adoption of the academic program description circulated according to the letter of the Department of Studies T 3/2906 on 3/5/2026 regarding the programs that adopt the Bologna Process as the basis for their work.

In this regard, we can only emphasize the importance of writing an academic programs and course description to ensure the proper functioning of the educational process.

Concepts and terminology:

Academic Program Description: The academic program description provides a brief summary of its vision, mission and objectives, including an accurate description of the targeted learning outcomes according to specific learning strategies.

Course Description: Provides a brief summary of the most important characteristics of the course and the learning outcomes expected of the students to achieve, proving whether they have made the most of the available learning opportunities. It is derived from the program description.

Program Vision: An ambitious picture for the future of the academic program to be sophisticated, inspiring, stimulating, realistic and applicable.

Program Mission: Briefly outlines the objectives and activities necessary to achieve them and defines the program's development paths and directions.

Program Objectives: They are statements that describe what the academic program intends to achieve within a specific period of time and are measurable and observable.

Curriculum Structure: All courses / subjects included in the academic program according to the approved learning system (quarterly, annual, Bologna Process) whether it is a requirement (ministry, university, college and scientific department) with the number of credit hours.

Learning Outcomes: A compatible set of knowledge, skills and values acquired by students after the successful completion of the academic program and must determine the learning outcomes of each course in a way that achieves the objectives of the program.

Teaching and learning strategies: They are the strategies used by the faculty members to develop students' teaching and learning, and they are plans that are followed to reach the learning goals. They describe all classroom and extra-curricular activities to achieve the learning outcomes of the program.

Academic Program Description Form

University Name: Ninevah University

Faculty/Institute: Information Technology

Scientific Department: Software

Academic or Professional Program Name: Bachelor's

Final Certificate Name: Bachelor of Software

Academic System: Bolonga Process and Course System

Description Preparation Date: **20/5/2026**

File Completion Date: **20/5/2026**

Signature:

Head of Department Name:

Date:

Signature:

Scientific Associate Name:

Date:

The file is checked by:

Department of Quality Assurance and University Performance

Director of the Quality Assurance and University Performance Department:

Date:

Signature:

Approval of the Dean

1. Program Vision

The Software Department is looking forward to being recognized for leadership and excellence in the field of software quality and to be supportive of effective links with local, regional and international bodies in the field of specialization, and to contribute to developing future visions in building the quality of applied software systems, making it a distinguished center in the field of educational, research and community service.

2. Program Mission

Preparing qualified and trained national competencies to start a working life based on an education with an advanced and influential technological base that meets future expectations of science and technology with strong relevance to problem-solving skills that enable them to adapt and adapt to the challenges of tomorrow in all software applications.

3. Program Objectives

- The department's commitment to providing an environment that supports creativity, quality and distinction, and to embrace and encourage development initiatives for its students.
- Preparing generations that are aware, distinguished, and with high skills and sufficient experience, enabling them to participate strongly and effectively in the development of society and develop its capabilities by providing them with the latest knowledge, skills and high moral values.
- Preparing scientific cadres specialized in the field of designing and building large and complex software systems. Providing an appropriate and

comfortable research and teaching environment for the teachers to reach high-quality outputs.

- Seek to open the door for higher studies in the department in a way that suits the needs of the academic community in this specialty.

4. Program Accreditation

None

5. Other external influences

Dean of Information Technology Faculty

6. Program Structure

Program Structure	Number of Courses	Credit hours	Percentage	Reviews*
Institution Requirements	7	15	9%	
College Requirements	7	25	12%	
Department Requirements	42	99	78%	
Summer Training	–	–	–	–
Other	–	–	–	–

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

7. Program Description

Year/Level	Course Code	Course Name	Credit Hours	
			theoretical	practical
First Grade	NVIT1112	Computer Skills	2	2
	NVIT1120	Programming Principles I	2	
	NVIT1116	Calculus	2	2
	NVIT1115	Principles of Logic		
	NVU11	English Language I	2	2
	NVITSW1118	Discrete Structures	1	
	NVIT1118	Statistics and Probabilities	2	
	NVITSW1220	Programming Principles II	2	2
	NVITSW1215	Logic Design	2	2
	NVITSW1217	Computation Theory	2	
	NVITSW1218	Assembly Labguage	2	2
	NVU12	Democracy and Human Rights	2	
	NVU16	Arabic Language 1	2	
Second Grade	NVIT2302	Data Structures and Algorithms I	2	2
	NVIT2304	Object Oriented Programming1 (OOP1)	2	2
	NVITSW2305	Software Engineering (I)	2	2
	NVITSW2306	Software Systems	2	2
	NVU15	English Language II	2	2
	NVITSW2319	Numerical Analysis	2	2
	NVITSW2404	Data Structures and Algorithms II	2	2
	NVIT2306	Database Management Systems	2	
	NVITSW2416	Software Engineering II	2	2
	NVITSW2427	Software Requirements	2	2
	NVU13	Crimes of the Baath regime in Iraq	2	2
	NVU17	Arabic Language 2	2	2
Third Grade	NVITSW3529	Software Engineering Tools	2	2
	NVITSW3523	Web Engineering1	2	2
	NVITSW3531	Data Communications and Computer Networks	2	2

	NVITSW3522	Computer Architecture	2	
	NVITSW3510	Fault Tolerance	2	2
	NVITSW3541	Object Oriented Programming2	2	2
	NVITSW3550	Software Project Management	2	2
	NVITSW3620	Operating System	2	2
	NVITSW3626	Compilers	2	2
	NVIT3556	AI Techniques	2	2
	NVITSW3630	Software Reliability	2	
	NVIT1255	Research methods1	1	2
	NVITSW3640	Software design	2	2
	NVITSW3650	Web Engineering2	2	2
Fourth Grade	NVITSW4720	Information Security	2	2
	NVITSW4724	Mobile Applications1	3	2
	NVITSW4731	Software Maintenance	2	2
	NVITSW4715	Research Methods2	2	
	NVITSW4719	Image processing	2	2
	NVITSW4727	Data Mining	2	
	NVITSW4802	Network Security	2	2
	NVITSW4832	Software Quality Assurance	2	2
	NVITSW4833	Real time systems	2	2
	NVITSW4834	Software Development Techniques	2	2
	NVITSW4853	Mobile Applications2	3	2
	NVITSW4817	Graduation Project	2	

8. Expected learning outcomes of the program

Knowledge	
Learning Outcomes 1	I. Describe and explain core scientific, mathematical, and computing concepts that underpin modern software systems, algorithms, and architectures. II. Integrate relevant knowledge from allied fields—such as data science, cybersecurity, human-computer interaction, and business—to design well-rounded software solutions.
Skills	

Learning Outcomes 2	I. Design and implement robust, maintainable, and secure software systems that meet specified functional and non-functional requirements (performance, scalability, usability, accessibility, sustainability). II. Select and apply appropriate programming languages, development frameworks, testing strategies, and software-engineering methodologies (e.g., Agile, DevOps, CI/CD) in solving real-world problems. III. Design and conduct empirical studies—such as prototyping, simulation, benchmarking, or user testing—interpret the resulting data, and draw evidence-based conclusions.
Ethics	
Learning Outcomes 4	I. Recognise the need for, and possess the ability to engage in, independent and life-long learning to keep pace with rapid advances in software and information technology. II. Communicate ideas, designs, and research findings clearly and persuasively in oral, written, and visual forms to both technical and non-technical audiences.

9. Teaching and Learning Strategies

Teaching and learning strategies and methods adopted in the implementation of the program in general are :

- Lecture Notes
- Tutorial
- Lab experiment
- Individual and group assignments
- Learn via video lecture notes
- Learn by search to vailed information

10. Evaluation methods

Implemented at all stages of the program in general are:

- Daily, weekly, and monthly quizzes.
- Midterm and Final Exam
- Practical exams
- Reports
- Graduation projects.

11. Faculty

Faculty Members

Academic Rank	Specialization		Special Requirements/Skills (if applicable)		Number of the teaching staff	
	General	Special			Staff	Lecturer
Professor	–	–	–	–	–	–
Assistant Professor	Accounting,	Cost Accounting, and Management Accounting			Staff	
Lecturer	Computer Science,	Machine Learning			Staff	
Lecturer	Software Engineering,	Software Engineering			Staff	
Lecturer	Computer Science,	Data Science			Staff	
Lecturer	Mathematics,	Numerical Analysis			Staff	
Lecturer	Computer Science,	Artificial Intelligence			Staff	
Assistant Lecturer	Management Information Systems	Management Information Systems			Staff	
Assistant Lecturer	Computer Science and Mathematics,	Pure Mathematics			Staff	

Assistant Lecturer	Mathematics,	Computational Mathematics			Staff	
Assistant Lecturer	English Language,	Speech Language			Staff	
Assistant Lecturer	Computer Science,	Data Security			Staff	
Assistant Lecturer	Business Administration,	Innovation Management			Staff	
Assistant Lecturer	Software Engineering	Software Engineering			Staff	
Assistant Lecturer	Information Technology	Information Technology			Staff	
Assistant Lecturer	Information Technology Management	Information Technology Management			Staff	

Professional Development

Mentoring new faculty members

Briefly describes the process used to mentor new, visiting, full-time, and part-time faculty at the institution and department level.

Professional development of faculty members

Briefly describe the academic and professional development plan and arrangements for faculty such as teaching and learning strategies, assessment of learning outcomes, professional development, etc.

12. Acceptance Criterion

(Setting regulations related to enrollment in the college or institute, whether central admission or others)

13.	**The most important sources of information about the program**
Website https://uoninevah.edu.iq/it/	

14.	Program Development Plan

Program Skills Outline												
				Required program Learning outcomes								
Year/Level	Course Code	Course Name	Core/ Optional	Knowledge		Skills				Ethics		
				I	II	I	II	III	IV	I	II	III
First Grade 25-26 (First Sem.)	NVU10	Computer Skills	Core	X	X	X	X	X	X	X	X	
	NVIT1120	Programming Principles I	Core	X		X	X	X	X		X	X
	NVIT1116	Calculus	Core		X		X	X	X			
	NVIT1115	Principles of Logic	Core	X	X	X	X				X	X
	NVU11	English Language I	Core	X								
	NVITSW1218	Discrete Structures	Core	X	X	X	X		X	X	X	
First Grade 25-26 (Second Sem.)	NVIT1118	Statistics and Probabilities	Core		X	X	X	X				X
	NVIT1220	Programming Principles II	Core	X	X	X	X	X		X	X	X
	NVIT1215	Logic Design	Core	X	X					X	X	
	NVITSW1217	Computation Theory	Core	X	X	X	X	X	X			X
	NVITSW1218	Assembly Labguage	Core	X	X	X	X	X	X	X	X	
	NVU12	Democracy and Human Rights	Core	X		X	X	X	X		X	X
	NVU16	Arabic Language 1	Core		X		X	X	X			

- **Please tick the boxes corresponding to the individual program learning outcomes under evaluation.**

Program Skills Outline												
				Required program Learning outcomes								
Year/Level	Course Code	Course Name	Core/ Optional	Knowledge		Skills				Ethics		
				I	II	I	II	III	IV	I	II	III
Second Grade 25-26 (First Sem.)	NVIT2302	Data Structures and Algorithms I	Core	X	X							
	NVIT2304	Object Oriented Programming (OOP)	Core	X	X	X	X	X	X		X	X
	NVITSW2305	Software Engineering (I)	Core	X	X							
	NVITSW2306	Software Systems	Core	X	X	X	X				X	X
	NVU15	English Language II	Core	X	X							
	NVITSW0219	Numerical Analysis	Core	X	X	X	X		X	X	X	
Second Grade 25-26 (Second Sem.)	NVIT2404	Data Structures and Algorithms II	Core	X	X	X	X	X				X
	NVIT2306	Database Management Systems	Core	X	X	X	X	X		X	X	X
	NVITSW2416	Software Engineering II	Core	X	X					X	X	
	NVITSW2427	Software Requirements	Core	X	X	X	X	X	X			X
	NVU13	Crimes of the Baath regime in Iraq	Core	X	X			X	X		X	
	NVU17	Arabic Language 2	Core	X	X		X	X	X			

Program Skills Outline												
				Required program Learning outcomes								
Year/Level	Course Code	Course Name	Core/ Optional	Knowledge		Skills				Ethics		
				I	II	I	II	III	IV	I	II	III
Third Grade 25-26 (First Sem.)	NVITSW3529	Software Engineering Tools	Core	X	X							
	NVITSW3523	Web Engineering1	Core	X	X	X	X	X	X		X	X
	NVITSW3531	Data Communications and Computer Networks	Core	X	X							
	NVITSW3522	Computer Architecture	Core	X	X	X	X				X	X
	NVITSW3510	Fault Tolerance	Core	X	X							
	NVITSW3541	Object Oriented Programming2	Core	X	X	X	X		X	X	X	
	NVITSW3550	Software Project Management	Core		X	X	X	X	X	X	X	
Third Grade 25-26 (Second Sem.)	NVITSW3620	Operating System	Core	X	X	X	X	X				X
	NVITSW3626	Compilers	Core	X	X	X	X	X		X	X	X
	NVIT3556	AI Techniques	Core	X	X					X	X	
	NVITSW3630	Software Reliability	Core	X	X	X	X	X	X			X
	NVIT1255	Research methods1	Core	X	X			X	X		X	
	NVITSW3640	Software design	Core	X	X		X	X	X			
	NVITSW3650	Web Engineering2	Core		X	X	X	X	X		X	X

Program Skills Outline												
				Required program Learning outcomes								
Year/Level	Course Code	Course Name	Core/ Optional	Knowledge		Skills				Ethics		
				I	II	I	II	III	IV	I	II	III
Fourth Grade 24-25 (First Sem.)	NVITSW4720	Information Security	Core	X	X	X	X	X	X	X	X	
	NVITSW4724	Mobile Applications1	Core	X		X	X	X	X		X	X
	NVITSW4731	Software Maintenance	Core		X		X	X	X		X	X
	NVITSW4715	Research Methods2	Core	X	X	X	X				X	X
	NVITSW4719	Image processing	Core	X			X	X		X	X	X
	NVITSW4727	Data Mining	Core	X	X	X	X		X	X	X	
Fourth Grade 24-25 (Second Sem.)	NVITSW4802	Network Security	Core		X	X	X	X				X
	NVITSW4832	Software Quality Assurance	Core	X	X	X	X	X		X	X	X
	NVITSW4833	Real time systems	Core	X	X	X	X			X	X	
	NVITSW4834	Software Development Techniques	Core	X	X	X	X	X	X	X		X
	NVITSW4853	Mobile Applications2	Core	X		X	X	X	X		X	X
	NVITSW4817	Graduation Project	Core	X	X	X	X	X	X	X	X	

Course Description Form

1. Module Name:
Computer skills
2. Module Code:
NVIT1112
3. Semester / Year:
1st /2025-2026
4. Description Preparation Date:
1/9/2026
5. Available Attendance Forms:
6. Number of Credit Hours (Total) / Number of Units (Total)
75/3
7. Module's administrator's (mention all, if more than one name)
Name: Haneen Talal Alwazzan Email: haneen.talal@uoninevah.edu.iq
8. Module's Objectives

Module's Objectives	• Identify the basic components of the computer. • Recognize the types of computers and their parts. • Understand the concept of electronic hacking, major harmful files, programs, and viruses. • Learn how to operate and use a computer. • Acquire skills in essential office applications needed by students.
9. Teaching and Learning Strategies	
Strategy	• Prepare students to explore the computer world and keep pace with scientific advancements. • Instill good ethics in dealing with the digital world while maintaining privacy. • Introduce students to the essential components of the computer. • Familiarize students with major programs in the Windows system and how to use them. • Encourage self-learning.

10. Module Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method

1	4	Understand stag of computer evolution & applications	Computer Basics	Lecture + Discussion	Daily & monthly tests, discussions
2	4	Recognize computer components & classifications	Computer Components & Types	Video demonstrations + Lecture	Daily & monthly tests, discussions
3	4	Learn about software entities number systems	Software entity, computer platform, hardware	Reading + Discussions, Lecture	Tests & discussions
4	4	Learn CPU & its parts, input/outp units	CPU, I/O devices	Reading + Discussions	Tests & discussions
5	4	Learn memory types & mouse functions	Memory, Mouse	Reading + Discussions	Tests & discussions
6	4	Learn programm languages & OS types	Programming Languages	Lecture + Practical Discussion	Tests & discussions
7	1	Monthly Exam	—	Exam	10%
8	4	Explore compute platform & syste unit	Platform, System Unit	Lecture + Discussion	Tests & discussions
9	4	Introduction to M Word	Word	Lecture + Practical Work	Tests & discussions
10	4	Using MS Word (files, storage, shortcuts)	Word	Lecture + Practical Work	Tests & discussions
11	4	Introduction to PowerPoint	PowerPoint	Lecture + Practical Work	Tests & discussions
12	4	Using PowerPoir (presentations, slides)	PowerPoint	Lecture + Practical Work	Tests & discussions
13	4	Introduction to Excel	Excel	Lecture + Practical Work	Tests & discussions
14	4	Using Excel (basi functions & formulas)	Excel	Lecture + Practical Work	Tests & discussions
15	3	Final Exam			

11. Module Evaluation

		Time/Number	Weight (Marks)
Formative assessment	**Quizzes**	2	10% (10)
	Assignments	2	10% (10)
	Projects / Lab.	1	10% (10)
	Report	1	10% (10)
Summative assessment	**Midterm Exam**	2hr	10% (10)
	Final Exam	3hr	50% (50)
Total assessment			100% (100 Marks)

12. Learning and Teaching Resources

Required textbo (curricular books, if ar	Ministry of Higher Education official textbook, Parts 1 & 2, First Year
Main referen (sources)	Ministry of Higher Education official textbook, Parts 1 & 2, First Year
Recommended books and references (scientific journals, reports…)	• *Yusr Al-Mustafa Series for Sciences: Fundamentals of Computer and Internet, Office 2010* – Dr. Ziad Mohammed Aboud, 2013
Electronic Referenc Websites	

Course Description Form

1. Course Name:
Calculus
2. Course Code:
NVIT1116
3. Semester / Year:
2025–2026
4. Description Preparation Date:
28/5/2026
5. Available Attendance Forms:
In-person-Online
6. Number of Credit Hours (Total) / Number of Units (Total)
3
7. Course administrator's name (mention all, if more than one name)
Name: Dr. Mohammed Mahmood Salih, Israa Mofak Email: mohammed.salihcs@uoninevah.edu.iq
8. Course Objectives

Course Objectives	**The course aims to introduce the following concepts:** **• Definition of a function and types of functions: graphing them, finding origin and range of each type.** **• Goal and continuity.** **• Derivative laws and their theorems.** **• Applications of the derivative, slope, and tangent equation.** **• Integration and methods for solving them.**

9. Teaching and Learning Strategies

Strategy	Cooperative Learning Theoretical lectures, dialogue and discussions, proble solving, reports, and daily assignments.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
First	3	Function, finding th origin and range of functions, and graphing them	Introduction to the course, definition of the set of numbers, and the univers symbols used in this course. Various practica examples.	Lecture	Discussions with students
Second	3	Limits an Continuit	Definition of Limit and ways find it with multiple examples	Lecture	Daily exams
Third	3	Limits an Continuit	Defining continuity and linking it to Lim with multiple applications	Lecture	Daily exams
Forth	3	derivative	Definition of derivative and applied exampl	Lecture	Daily exams
Fifth	3	derivative	Derivative theories and lav with their application and drawing	Lecture	Daily exams and homewor
Sixth	3	derivative	Derivative application examples	Lecture	Daily exams a
Seventh	3	Integratic and its solutions, indefinite integratic	Constant functi and power functions with various exampl	Lecture	Daily exams and homewor
eighth	3	Integratic and its solutions,	Rational functions and	Lecture	homework

		indefinite integratio	their types with various exampl		
Ninth	3	Integratio and its solutions, indefinite integratio	Logarithmic and Exponential Functions with Various Exampl	Lecture	Daily exams
Tenth	3	Integratio and its solutions, indefinite integratio	Additional examples on the topic of Ninth week	Lecture	Daily exams and homewor
Eleventh	3	Integratio and its solutions, indefinite integratio	Trigonometric functions: writi and applying th laws	Lecture	Daily exams
Twelfth	3	Integratio and its solutions, indefinite integratio	Inverse Trigonometric Functions Writi and Applying Laws	Lecture	Daily exams
Thirteent	3	Integratio and its solutions, indefinite integratio	Integration by parts and substitution	Lecture	Daily exams
Fourteent	3	Integratio and its solutions, indefinite integratio	Integration by fractional divisi	Lecture	Daily exams

11. Course Evaluation

The grade is distributed out of 100 based on the tasks the student completes, such as in-class and extracurricular assignments, daily exams, and report writing.

12. Learning and Teaching Resources

Required textbooks (curricular books any)	Calculus with Analytical Geometry: Part 1 by I. G. Purcell, 1983.
Main references (sources)	Thomas, G. B. Calculus. Reading: Addison-Wesky.2003. Anton, Bivens, Davis. Calculus. Seventh Edition New York, 2002.

Recommended books and references (scientific journals, reports…)	Hintikka, Jaakko. The principles mathematics revisited. Cambridge Univers Press, 1998.
Electronic References, Websites	https://ocw.mit.edu/resources/res-18-001-calculu online-textbook-spring-2005/textbook/ https://www.freebookcentre.net/maths-booksdownload/Calculus-Lecture-Notes.html https://www.freebookcentre.net/maths-booksdownload/Advanced-Calculus-Lecture-Notes for-Mathematics.html https://ocw.mit.edu/courses/mathematics/18-01- single-variablecalculus-fall-2006/lecture-notes/ https://www.math.upenn.edu/~rimmer/math103 otes.html

Course Description Form

1. Course Name:	
Statistics and Probabilities	
2. Course Code:	
NVIT1118	
3. Semester / Year:	
2025−2026	
4. Description Preparation Date:	
28/5/2026	
5. Available Attendance Forms:	
In-person-Online	
6. Number of Credit Hours (Total) / Number of Units (Total)	
3	
7. Course administrator's name (mention all, if more than one name)	
Name: Dr. Mohammed Mahmood Salih, Zina Mohammed Sideek Email: mohammed.salihcs@uoninevah.edu.iq	
8. Course Objectives	
Course Objectives	**Introducing the student to statistics a how to conduct statistical analyses arrive at accurate results.**
9. Teaching and Learning Strategies	

Strategy	Cooperative Learning Theoretical lectures, dialogue and discussio problem-solving, reports, and daily assignments.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
First	3	Statistics	Definition of statistics, definition of descriptive and quantitative da and their graphical representation	Lecture	Discussions with studen
Second	3	Types of statistical da	Definition of classified and unclassified da	Lecture	Daily exams
Third	3	Frequency distribution tables	Types of frequency table and how data is distributed within them	Lecture	Daily exams
Forth	3	Frequency distribution tables	Organizing data in two types of frequency distribution tables How to calculate the frequency of ea	Lecture	Daily exams
Fifth	3	Frequency distribution tables	Organizing data in two types of frequency distribution tables How to calculate the frequency of ea	Lecture	Daily exams and homework
Sixth	3	Cumulative Frequency	Extracting the ascending and	Lecture	Daily exams

			descending cumulative frequency and their graphical representation		
Seventh	3	Cumulative Frequency	Extracting the ascending and descending cumulative frequency and their graphical representation	Lecture	Daily exams and homework
eighth	3	Charts	Types of graph histogram, grap polygon, sector circle.	Lecture	homework
Ninth	3	Measures of central tendency	Calculate the arithmetic mea weighted mean harmonic mean	Lecture	Daily exams
Tenth	3	Measures of central tendency	Calculating the geometric mea median, mode, and the relationship between the arithmetic mea median, and mode	Lecture	Daily exams and homework
Eleventh	3	Measures of central tendency	Calculating the geometric mea median, mode, and the relationship between the arithmetic mea median, and mode	Lecture	Daily exams
Twelfth	3	Dispersion measures	Definition of dispersion, absolute dispersion	Lecture	Daily exams

			measures, rang mean deviation		
Thirteenth	3	Dispersion measures	standard deviation, variance	Lecture	Daily exams
Fourteenth	3	Dispersion measures	Relative dispersion measures, coefficient of variation, standard score	Lecture	Daily exams

11. Course Evaluation

The grade is distributed out of 100 based on the tasks the student completes, such as in-class and extracurricular assignments, daily exams, and report writing.

12. Learning and Teaching Resources

Required textbooks (curricular books, if any)	Statistical Methods, Dr. Sabri Radeef Al-Ani, Ministry of Highe Education and Scientific Research, 1982.
Main references (sources)	Principles of Statistics, Dr. Mahmoud Hassan Al-Mashhada and Mr. Amir Hanna Hormuz, University of Baghdad, 1989.
Recommended books and references (scientific journals, reports…)	
Electronic References, Websites	

Course Description Form

13. Course Name:
Computation Theory
14. Course Code:
NVITSW1217
15. Semester / Year:
Second / 2026
16. Description Preparation Date:
1/6/2026
17. Available Attendance Forms:
In-Person
18. Number of Credit Hours (Total) / Number of Units (Total)
150/6
19. Course administrator's name (mention all, if more than one name)
Mostafa ismail mostafa.i.windi@uoninevah.edu.iq
20. Course Objectives

Course Objectives	**Through this course, students will learn about different models of computation. The learning objectives of the course are to:** **1. Familiarity with the basic concepts of computational theory, the basics of language theory, and general concepts in building programming languages in a mathematical manner.** **2. Knowing and distinguishing the different operations that take place on languages and machines that each language recognizes.** **3. Knowledge of the basics of automata theory and regular expressions as mathematical models help in defining programming languages and formal languages.** **4. The ability to use the concepts of computational theory as basic tools in building programming languages and how to derive them in a logical manner, as its tools represent an important part of the stages of building compilers for each programming language.** **5. Acquisition of basic skills as an introduction to building languages and problem-solving.**

	6. Acquisition of theoretical concepts to know the various processes that take place in languages.

21. Teaching and Learning Strategies

Strategy	The main strategy that will be adopted in delivering this module is to understand which problems can be solved using computational devices and how efficiently those problems can be solved. 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.

22. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
Week 1	3	Understand the foundational elements of computation such as sets, strings, alphabets, and formal languages.	Basic concepts, Set, Strings, alphabets and language	Lecture + Whiteboard Examples	Observation of class participation + short oral Q&A
Week 2	3	Construct and apply regular expressions to define and manipulate simple formal languages.	Regular expressions	Lecture + Hands-on Exercises	In-class quiz on regex + small written assignment
Week 3	3	-Define grammars and use them to specify languages -Classify languages based on Chomsky's hierarchy	Grammars: Definition, Specifying languages by grammars, The Chomsky hierarchy of languages	Lecture + Guided Practice	Homework assignment on grammar construction
Week 4	3	Describe and differentiate between finite automata	Finite state automata: Acceptors,	Lecture + Guided Practice	In-class diagram construction activity +

		models: acceptors, classifiers, and transducers	Classifiers, and Transducers		group work evaluation
Week 5	3	Compare DFA and NFA and convert between them	Deterministic and nondeterministic finite state automata	Lecture + Comparative Analysis	Worksheet-based evaluation + mini quiz
Week 6	3	Apply DFA and NFA to language recognition tasks	Deterministic and nondeterministic finite state automata	Lecture + Guided Practice	Short written exercise + peer review of DFA/NFA designs
Week 7	3	Design Moore and Mealy machines for output-driven automation	Finite State Automata with Output: Moor machine and Mealy machine	Lecture + Guided Practice	Assignment to design output machines + brief presentation
Week 8	3	Assess understanding and application of topics covered in Weeks 1–7	Mid Term Exam	Written Examination	Formal written exam covering Weeks 1–7 topics
Week 9	3	Define and apply context-free grammars to describe programming language constructs	Context Free Grammar and Languages	Interactive Whiteboard Session	In-class grammar analysis + worksheet
Week 10	3	Identify context-free languages and construct grammars accordingly	Context Free Grammar and Languages	Interactive Whiteboard Session	Homework on CFG derivations
Week 11	3	Generate parse trees from grammars to visualize sentence structure and syntax	Generation of Derivation Tree	Lecture + Problem Solving	Draw derivation trees during class + peer assessment
Week 12	3	- Simplify context-free grammars by removing useless symbols and rules	Simplification of context-free grammars, Ambiguity and Unambiguity in	Lecture + Guided Practice	Assignment: simplify grammar and identify ambiguity

		- Identify and resolve ambiguity in grammar definitions	Context Free Grammars		
Week 13	3	Explore closure properties (union, concatenation, Kleene star, substitution) of CFLs	Properties of context free languages, closure properties of context free languages (union, concatenation, Kleene closure and substitutions	Lecture + Guided Practice	Quiz on closure properties + class problem-solving
Week 14	3	Convert grammars into Chomsky Normal Form and explain its advantages	Chomsky normal forms, converting Context Free Grammer into Chomsky Normal Form	Lecture + Guided Practice	In-class transformation exercises + group work evaluation
Week 15	3	Design PDA from context-free grammars	Pushdown Automata (PDA), Convert Context Free Grammar to Push Down Automata	Lecture + Guided Practice	Practical assignment: convert CFG to PDA
Week 16	3	Review all topics and reinforce the integration of theoretical models for problem-solving in computation theory	Preparatory week before the final Exam	Review Session	Mock quiz + feedback session

23. Course Evaluation

Quizzes: 10%
Homework: 10%
Seminar: 10%
Report: 10%
Min Term: 10%
Final exam: 50%

24. Learning and Teaching Resources

Required textbooks (curricular books, if any)	Introduction to Computer Theory, 2nd Edition, by Daniel I. A. Cohen John Wiley & Sons, Inc 1997. ISBN 0-471-13772-3.

Main references (sources)	
Recommended books and references (scientific journals, reports…)	Introduction to Automata Theory, Languages, and Computation, 2/E, by John E. Hopcroft, Rajeev M., Jeffrey D. Ullman, Addison-Wesley 2001. ISBN 0-20144124-1. Introduction to the Theory of Computation, 2nd Edition, Boston, MA: Course Technology, by Michael Sipser 2006. ISBN: 0534950973.
Electronic References, Websites	

Course Description Form

1. Course Name:
Numerical Analysis
2. Course Code:
NVITSW2319
3. Semester / Year:
2025−2026
4. Description Preparation Date:
28/5/2026
5. Available Attendance Forms:
In-person-Online
6. Number of Credit Hours (Total) / Number of Units (Total)
3
7. Course administrator's name (mention all, if more than one name)
Name: Dr. Mohammed Mahmood Salih, Zina Mohammed Sideek, Roa'a Saleh Email: mohammed.salihcs@uoninevah.edu.iq
8. Course Objectives

Course Objectives	**• Introduce students to the sources and ty of errors and mathematical operations.** **• Introduce and teach students methods solving nonlinear equations.**

	• Introduce and teach students numer methods for solving systems of lin equations. • Introduce students to numerical analy algorithms and apply them programm using MATLAB.

9. Teaching and Learning Strategies

Strategy	Cooperative Learning Theoretical lectures, dialogue and discussions, problem-solvi reports, and daily assignments.

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
First	3	Sources errors	Definition of error sources and types	Lecture	Discussions with student
Second	3	Sources errors	Arithmetic operatio on errors	Lecture	Daily exams
Third	3	Solutions of nonlinear equations	Bisection method	Lecture	Daily exams
Forth	3	Solutions of nonlinear equations	Bisection method	Lecture	Daily exams
Fifth	3	Solutions of nonlinear equations	False position meth	Lecture	Daily exams and homewo
Sixth	3	Solutions of nonlinear equations	False position meth	Lecture	Daily exams
Seventh	3	Solutions of nonlinear equations	Newton's method	Lecture	Daily exams and homewo
eighth	3	Solutions of nonlinear equations	Numerical convergence of Newton's method	Lecture	homework

Ninth	3	Numerical Solutions of Linear system	Direct methods/Gaussian elimination method	Lecture	Daily exams
Tenth	3	Numerical Solutions of Linear system	Direct methods/Gaussian elimination method	Lecture	Daily exams and homewo
Eleventh	3	Numerical Solutions of Linear system	Iterative methods/Jacobi method	Lecture	Daily exams
Twelfth	3	Numerical Solutions of Linear system	Iterative methods/Jacobi method	Lecture	Daily exams
Thirteenth	3	Numerical Solutions of Linear system	Iterative methods/Gauss-Seidel method	Lecture	Daily exams
Fourteenth	3	Numerical Solutions of Linear system	Iterative methods/Gauss-Seidel method	Lecture	Daily exams

11. Course Evaluation

The grade is distributed out of 40 based on the tasks the student completes, such as in-class and extracurricular assignments, daily exams, and report writing. In addition, there will be a midterm exam worth 10 points and a final exam worth 50 points.

12. Learning and Teaching Resources

Required textbooks (curricular books, if any)	Principles of Numerical Analysis, by Muhammad Sadiq Saifi and others, I Al-Atheer Printing and Publishi House, 1987.
Main references (sources)	Numerical Methods Using MATLAB, Mathews et.al. Prentice Hall, 1999.
Recommended books and references (scientific journals, reports…)	Numerical Analysis 9th edition, R. Burden et.al. Brooks/Cole, Cenga learning, 2011.
Electronic References, Websites	

Course Description Form

1. Course Name: Software Engineering I

2. Course Code:

NVITSW2305

3. Semester / Year: Semester1

2025-2026

4. Description Preparation Date:

2026/6/7

5. Available Attendance Forms:

6. Number of Credit Hours (Total) / Number of Units (Total)

7 ECTS . 5 hours

7. Course administrator's name (mention all, if more than one name)

Name: Dr. Muyassar Dalli Hamad
Email: muyassar.dalli@uoninevah.edu.iq

8. Course Objectives

Course Objectives	**A thorough definition of software engineering, its objectives, applicability for all types of software, and a synopsis of the concepts, principles, models, and tools of the analysis phase are all covered in this course.**

9. Teaching and Learning Strategies

Strategy	

10. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1st	3		**Introduction in softwar Engineering Definitions, concepts,and principles**		
2nd	3		**- Software application with examples - Complementary & Conflicting Goals in SV**		

3rd	3		**-SW project-planning**		
4th	3		**-Software project Estimation LOC & FP**		
5th	3		- SW Project scheduling		
6th	3		Software development models-1		
7th	3		Software development models-2		
8th	3		Software system requirements		
9th	3		The Mechanics of Structured Analys		
10th	3		The Mechanics of Structured Analys		
11th	3		Structure Charts (block diagram)		
12th	3		Object-Oriented programming Concepts and Principles		
13th	3		Class and object Attribute & Operation & message		
14th	3		Encapsulation Inheritance		
15th	3		Polymorphism Abstraction		

11. Course Evaluation

Distributing the score out of 100 according to the tasks assigned to the student such as daily preparation, daily oral, monthly, or written exams, reports .... etc

12. Learning and Teaching Resources

Required textbooks (curricular books, if any)	**Software Engineering Ninth Edition**
Main references (sources)	**Enterprise Architect help**
Recommended books and references (scientific journals, reports…)	
Electronic References, Websites	**Internet**

Course Description Form

13. Course Name: Software Engineering II
14. Course Code:
NVITSW2416
15. Semester / Year: Semester1
2025-2026
16. Description Preparation Date:
2026/6/7
17.Available Attendance Forms:
18.Number of Credit Hours (Total) / Number of Units (Total)
6 units/5 hours
19. Course administrator's name (mention all, if more than one name)
Name: Dr. Muyassar Dalli Hamad Email: muyassar.dalli@uoninevah.edu.iq
20. Course Objectives

Course Objectives	**The Introduction to Software Engineering (2) course comprises defining its phases, demonstrating how modelling works in each phase, and providing applications for each phase This section provides an overview of software engineering models, methodologies, tools, and relationships across all phases.**

21. Teaching and Learning Strategies

Strategy	

22. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1st	3		**Design principle** **- Abstraction** **- Refinement** **-Modularity**		
2nd	3		**Design principle**		

			-SWE Architecture -Control hierarchy & structure partition		
3rd	3		**Design principle - Data Structure -SW procedure -Information hiding**		
4th	3		**Modularity -Cohesion - Coupling**		
5th	3		**Architecture Design**		
6th	3		**Design Process**		
7th	3		**Architecture Design -program structure**		
8th	3		**User Interface Design - Process**		
9th	3		**Component–level –Des - Box Diagram**		
10th	3		**Component–level –Des - Pdl**		
11th	3		**Software testing -black test - whit test**		
12th	3		**Software testing strate - Unit test -Integration test**		
13th	3		**Software testing strate -Validating test**		
14th	3		**Software testing strate -System test -UAT**		
15th	3		**MVC Architecture Pattern**		

23. Course Evaluation

Distributing the score out of 100 according to the tasks assigned to the student such as daily preparation, daily oral, monthly, or written exams, reports .... etc

24. Learning and Teaching Resources

Required textbooks (curricular books, if any)	**- Software Engineering, Ninth Edition - Software Engineering, Seventh Editi**
Main references (sources)	**- Enterprise Architect help**
Recommended books and references (scientific journals, reports…)	
Electronic References, Websites	**- Internet**

Course Description Form

Course Title	Data Structures and Algorithms 1
Course Code	NVIT2302
Semester / Year	Third Semester / Second Year
Semester Preparation Date	22/09/2026
Available Attendance Forms	In-person (Theoretical and Practical) and Online
Number of Credit Hours (Total) / Number of Units (Total)	150 hours / 6 ECTS units
Course administrator's name	Name: Mustafa Nabeel Salem Email: mustafa.nabeel@uoninevah.edu.iq

Course Objectives

This course aims to achieve a set of cognitive, skill-based, and affective objectives for the student, as follows:

Cognitive Objectives	**Skill Objectives**	**Affective Objectives**
Recognizing the fundamental concepts of various data structures (Arrays, Stacks, Queues, Linked Lists).	Understanding how to analyze the performance and efficiency of algorithms in terms of Time and Space Complexity.	Appreciating the importance of selecting the optimal data structure and algorithm to improve program performance.
Understanding the basic principles of common search algorithms.	Ability to apply appropriate data structures to solve programming problems efficiently.	Developing logical and systematic thinking in analyzing problems and designing solutions.
	Designing and implementing basic search algorithms using the Python programming language.	Adhering to best practices in writing clear, readable, and maintainable code.
	Acquiring the practical skills necessary to write programs that handle complex data structures.	

Teaching and Learning Strategies

Strategy	**Description**
Theoretical Lectures	Presenting basic concepts with illustrative examples using presentations and a whiteboard.

Practical Applications	Lab sessions to apply theoretical concepts and solve programming problems.
Problem-Based Learning	Posing real-world problems and guiding students to choose the most suitable data structure and algorithm to solve them.
Assignments and Projects	Assigning periodic programming assignments and a semester project to enhance understanding and develop skills.
Interactive Learning	Encouraging discussion and asking questions during theoretical and practical lectures.

Course Structure

Week	**Hours**	**Required Learning Outcomes**	**Unit or subject name**	**Learning method**	**Evaluation method**
First	2 Theory + 2 Practical	- Understand the basic concepts of data structures and algorithms. - Set up the Python work environment.	**Theory:** Introduction to Data Structures and Algorithms (ADT). **Practical:** Review of Python basics (variables, loops, functions).	Lecture, Practical Application	Discussion, Practical Assignment
Second	2 Theory + 2 Practical	- Ability to analyze algorithm efficiency. - Write simple Python programs.	**Theory:** Algorithm Analysis (Time and Space Complexity, Big-O). **Practical:** Exercises on Python basics.	Lecture and Examples, Practical Application	Quiz
Third	2 Theory + 2 Practical	- Understand the array structure and its operations. - Handle	**Theory:** Arrays: one-dimensional and two-dimensional,	Lecture, Practical Application	Programming Assignment

		arrays in Python.	basic operations. **Practical:** Implementing array operations in Python.		
Fourth	2 Theory + 2 Practical	- Understand the working principle of a Stack. - Implement a stack programmatically.	**Theory:** Stack: concept (LIFO), operations (Push, Pop). **Practical:** Building a stack using arrays in Python.	Interactive Lecture, Practical Application	Quiz
Fifth	2 Theory + 2 Practical	- Understand the concept of Recursion and its importance.	**Theory:** Stack Applications: Recursion. **Practical:** Solving problems using Recursion (Factorial, Fibonacci).	Lecture and Examples, Practical Application	Programming Assignment
Sixth	2 Theory + 2 Practical	- Understand expression notations (Infix, Prefix, Postfix). - Convert and evaluate expressions.	**Theory:** Mathematical Expressions (Prefix, Infix, Postfix) and their conversion. **Practical:** Program to convert Infix to Postfix and evaluate it.	Interactive Lecture, Practical Application	Discussion
Seventh	-	- Assess student's understanding of previous topics.	Mid-Term Exam	Exam	Exam Grade

Eighth	2 Theory + 2 Practical	- Understand the working principle of a Queue. - Implement a queue programmatically.	**Theory:** Queue: concept (FIFO), operations (Enqueue, Dequeue). **Practical:** Building a queue using arrays in Python.	Lecture, Practical Application	Programming Assignment
Ninth	2 Theory + 2 Practical	- Understand queue applications. - Implement a circular queue.	**Theory:** Queue applications and Circular Queue. **Practical:** Building more queue applications and a circular queue.	Interactive Lecture, Practical Application	Discussion
Tenth	2 Theory + 2 Practical	- Understand Linked Lists. - Implement a simple linked list.	**Theory:** Singly Linked List and its operations (add, delete, search). **Practical:** Building a simple linked list and implementing basic operations in Python.	Lecture, Practical Application	Programming Assignment
Eleventh	2 Theory + 2 Practical	- Understand advanced types of linked lists.	**Theory:** Doubly Linked List. **Practical:** Building a doubly linked list in Python.	Lecture and Examples, Practical Application	Quiz
Twelfth	2 Theory + 2 Practical	- Understand special types and	**Theory:** Circular Linked List and	Interactive Lecture,	Programming Assignment

		applications of linked lists.	applications of linked lists. **Practical:** Practical application on circular lists, stacks, and queues using linked lists.	Practical Application	
Thirteenth	2 Theory + 2 Practical	- Understand the Linear Search algorithm. - Implement and analyze the algorithm.	**Theory:** Search Algorithms (Part 1): Linear Search. **Practical:** Implementing Linear Search.	Lecture, Practical Application	Programming Assignment
Fourteenth	2 Theory + 2 Practical	- Understand the Binary Search algorithm. - Implement and analyze the algorithm.	**Theory:** Search Algorithms (Part 2): Binary Search. **Practical:** Implementing Binary Search.	Lecture and Examples, Practical Application	Quiz
Fifteenth	2 Theory + 2 Practical	- Understand the principles of Hashing. - Comprehensive review of basic concepts.	Introduction to Hashing and comprehensive review. **Practical:** Implementing a simple hash function and practical review.	Interactive Lecture, Practical Review	

Course Evaluation

Assessment Method	**Percentage**	**Notes**
Final Exam	50%	Comprehensive written exam at the end of the semester.

Mid-Term Exam	10%	Written and practical exam in the seventh week.
Quizzes	10%	Periodic quizzes (5 or more) may be given at any time.
Assignments	10%	Programming assignments submitted periodically.
Lab Work	10%	Assessment of performance and participation in the lab.
Project / Report	10%	A programming project where the student applies the concepts learned.
Total	**100%**	

Learning and Teaching Resources

Type	**Source**
Required Textbooks	- "Data Structures and Algorithms in Python" by Goodrich, Tamassia, and Goldwasser.
Main References (Sources)	- "Problem Solving with Algorithms and Data Structures using Python" by Miller and Ranum.
Recommended Books and References	- "Introduction to Algorithms" by Cormen, Leiserson, Rivest, and Stein (CLRS).
Electronic References, Websites	- GeeksforGeeks (https://www.geeksforgeeks.org/data-structures/) - W3Schools Python Tutorial (https://www.w3schools.com/python) - Coursera / edX (Specialized courses in data structures)

Course Description Form

25.	Course Name:

Software Requirements
26. Course Code:
NVITSW2427
27. Semester / Year:
Four/2026
28. Description Preparation Date:
31−5−2026
29. Available Attendance Forms:
On-campus lectures
30. Number of Credit Hours (Total) / Number of Units (Total)
2 hours theoretical, 2 hours lab, 1 hour tutorial /week
31. Course administrator's name (mention all, if more than one name)
Name: Asst. Lect. Safa Sabah Mohammed Email: safa.sabah@uoninevah.edu.iq

32. Course Objectives	
Course Objectives	• Define and explain the concept of requirements engineering and its significance in software development. • Identify and distinguish between user and system requirements for a software system. • Classify and describe functional and non-functional requirements. • Apply requirements elicitation techniques to gather and document software requirements. • Develop a requirements specification document for a software system. • Validate software requirements to ensure accuracy, completeness, and consistency. • Manage changes to software requirements effectively throughout the development lifecycle.
33. Teaching and Learning Strategies	
Strategy	• Presentation • Discussion • Practical computer applications (Lab works) • Brainstorming sessions • Training • Case study • Instructor feedback

34. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	2 hours-theoretical 2 hours-lab	Understanding Course Structure	Introduction to the course	-Presentations -Discussion	Interactive Q&A
2	2 hours-theoretical 2 hours-lab	-Explain RE, its importance, and the complete process. -Understand how abstraction levels increase over time, evolving from low-level coding to modeling using specialized modeling languages. - Explore key features of Enterprise Architect	-Requirements engineering definition and process -Introduction to modeling -Enterprise architect tool overview	-Presentations -Discussion	Short Answer Questions
3	2 hours-theoretical 2 hours-lab 1 hour -tutorial	- Distinguish between user and system requirements - Explore various UML diagrams using Enterprise Architect	- User and system requirements - Introduction to UML	-Presentations -Lab work -Discussion	Short Answer Questions
4	2 hours-theoretical 2 hours-lab 1 hour -tutorial	- Analyze software requirements through a real-world case study. -Use UML modeling language to represent requirements.	- A case study on software requirements - UML diagrams	-Presentations -Brainstorming -Training -Discussion - Case study	Weekly exam
5	2 hours-theoretical 2 hours-lab 1 hour -tutorial	- Identify functional and non-functional requirements - Diving deep into requirements modeling	-Functional and non-functional requirements -modeling functional requirements	- Presentations -Training -Discussion	Weekly exam
6	2 hours-theoretical 2 hours-lab 1 hour -tutorial	- Ability to classify different types of non-functional requirements - Understand the use of use case modeling and its role in software requirements	Types of non-functional requirements - use case diagram	-Presentations -Training -Discussion	Weekly exam
7	2hours-theoretical 2 hours-lab 1 hour -tutorial	-Understand the different stages of requirements engineering process -The use of use case diagrams to model functional requirements	-Requirements engineering process - use case diagram practices	- Presentations -Model demonstration -Discussion	Weekly exam
8	2hours-theoretical 2 hours-lab 1hour -tutorial	-Effectively interact with stakeholders to elicit requirements. - Respecting stakeholders' opinions and interacting with them in a professional manner.	-Requirements elicitation techniques - Requirements diagram	- Presentations - Brainstorming - Training	- Weekly exam - Seminar - Discussions

		-Effective communication skills: listening, and questioning. - Teamwork -Presentation skills self-learning			
9	2 hours-theoretical 2 hours-lab 1 hour -tutorial	- familiarity with the guidelines used in writing requirements using Natural Language - Confidentiality and accuracy in documenting information. - Create and interpret requirement diagrams	-Requirements specification/ Natural language specification - requirements diagram	-Presentations -Training -Discussion	Weekly exam
10	2 hours-theoretical 2 hours-lab 1 hour -tutorial	-Exploring the structure of structured requirements document for example IEEE template - Exploring different features of Requirements diagram	-Requirements specification/ Structured specifications -Requirement diagrams examples	-Presentations -Training -Discussion	Weekly exam
11	2 hours-theoretical 2 hours-lab 1 hour -tutorial	-Specification of requirements using graphical representation such as UML - Create requirements diagrams to represent requirements.	- Requirements specification/ graphical representation - Requirement diagrams	-Presentations -Training -Discussion	Weekly exam
12	2 hours-theoretical 2 hours-lab 1 hour -tutorial	- Understand the purpose and the components of a software requirements document. -Understand the concept of domain modeling and its role in software development	- The software requirements document -introducing domain model	-Presentations -Brainstorming - Training	Midterm exam
13	2 hours-theoretical 2hours-lab 1 hour -tutorial	-Identify and apply techniques for validating requirements - Create and interpret domain models using UML modeling technique	- Requirements validation - domain Model examples	-Presentations -Training -Discussion	Weekly exam
14	2 hours-theoretical 2 hours-lab	- Apply change management processes - Apply best practices for domain modeling	- Requirements change management - Domain Model practices	-Presentations -Discussion - Training	Discussion
15	2 hours-theoretical 2 hours-lab	Consolidating knowledge in software requirements	Review and preparation to final exam	Discussion	

. Course Evaluation

Weekly written exam = 10, seminar= 10 , lab= 10 , in-class assignments= 5, class participation=5, midterm exam =10
final exam= 50

36. Learning and Teaching Resources

Required textbooks (curricular books, if any)	**Software Engineering 10th Edition , Ian Sommerville**
Main references (sources)	**Software Engineering 10th Edition , Ian Sommerville**

Recommended books and references (scientific journals, reports…)	
Electronic References, Websites	-Enterprise architect https://sparxsystems.com/mbse/?gad_source=1&gad_campaignid=12111855332&gbraid=0AAAADu8ML4z2XBEDb-XsgYRamp51Ujp&gclid=CjwKCAjwruXBBhArEiwACBRtHbW5b5gHbJm6B-YYe941yPaLqN8DEAXK4OjQ8ZoZhd2DoRVQ-iVncxoCThoQAvD_BwE -UML specification https://www.omg.org/spec/UML/2.5.1/About-UML deling Language Specification Version 2.5.1

Course Description Form

37. Course Name:
Object Oriented Programming (OOP)
38. Course Code:
NVIT2304
39. Semester / Year:
First Semester / Second
40. Description Preparation Date:
1/9/2025
41. Available Attendance Forms:
Presence
42. Number of Credit Hours (Total) / Number of Units (Total)
Five hours = theoretical (2) + practical (2)
43. Course administrator's name (mention all, if more than one name)
Dr:ali_hashim ali.al-shakarchi@uoninevah.edu.iq
44. Course Objectives

Course Objectives	• Understand the principles of object-oriented design. • Use Java syntax and semantics effectively in programming. • Apply OOP concepts to design and implement programs. • Develop programs using classes, objects, interfaces, and packages.
45. Teaching and Learning Strategies	
Strategy	• Use class time for live-coding sessions that demonstrate how ea concept translates into real Flutter code. • In labs, walk students through a short, working demo (scrolli list, custom navigation flow, etc.). • Immediately challenge them to extend or "remix" the der with new features, explaining every line they add or modify. •

46. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	5		Course introduction + Introduction to Java & OOP I	Theoretical	

2	5		(Lab): Java syantax (Varaibles, loop, condition)	Theoretical	Homework
3	5		(T): Ch 1: Introduction to Java & OOP II (Java syntax)	Theoretical Tutorial	Assignment
4	5		(T): Ch 1: Introduction to Java & OOP II (Java syntax)	Theoretical Tutorial	Assignment
5	5		(T): Ch 2: OOP components II (Constractor + Encapsulation)	Theoretical Practical Tutorial	Lab session and Assignment
6	5		(T): Ch 2: OOP components III (Acess Modifiers)	Theoretical Practical Tutorial	Quiz Exam
7	5		(T): Ch 3: Inhertance I	Theoretical Practical Tutorial	Lab session and Assignment
8	5		Midterm Exam	Theoretical Practical Tutorial	Midterm Exam
9	5		(T): Ch 3: Inhertance II	Theoretical Practical Tutorial	Lab session and Assignment
10	5		(T): Ch 3: Inhertance III (super)	Theoretical Tutorial	Lab session and Assignment
11	5		(T): Ch 3: Inhertance IV (Polymorphism & Method Overriding)	Theoretical Practical Tutorial	Quiz exam
12	5		(T): Ch4: Abstraction (Abstract class)	Theoretical Practical Tutorial	Lab session
13	5		(T): Ch4: Abstraction (Interface)	Theoretical Practical Tutorial	Lab session and Assignment
14	5		(T): Ch5: Exception Handling I	Theoretical Practical Tutorial	Lab session and Assignment
15	5		(T): Ch5: Exception Handling II	Presentation	
16	5		Project Presentation		

47. Course Evaluation

- Participation (5%)
- Theory Quizzes (15%)
- Midterm (10%)
- Lab Activities (10%)

• Project (10%) • Final Exam (50%)

48. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	-
Main references (sources)	Java: How to Program (Late Objects), 11th Edition — Paul Deitel & Harvey Deitel, Pearson.
Recommended books and references (scientific journals, reports…)	Core Java Volume I – Fundamentals by Cay S. Horstmann
Electronic References, Websites	https://www.w3schools.com

Course Description Form

49. Course Name:
Intelligent technique

50. Course Code:
NVIT3556
51. Semester / Year:
First /2025−2026
52. Description Preparation Date:
20/8/2026
53.Available Attendance Forms:
Presence
54.Number of Credit Hours (Total) / Number of Units (Total)
Six
55. Course administrator's name (mention all, if more than one name)
Name: Ali Mohsin Ahmed Email: ali.mohsin@uoninevah.edu.iq
56.Course Objectives

Course Objectives	• **Provide students with a solid foundational understanding of the concepts and theories underpinning intelligent (computational-intelligence) techniques.** • **Enable students to design and develop intelligent models and systems capable of solving complex problems.** • **Connect theoretical concepts to practical applications in classification, prediction, and optimization.** • **Strengthen critical and creative thinking skills in selecting and applying appropriate algorithms.** • **Foster awareness of AI ethics and the responsible use of intelligent technologies.**

57. Teaching and Learning Strategies	
Strategy	**Knowledge** • Knowledge in artificial intelligence and intelligent (computational-intelligence) techniques and their importance. • Employing AI and intelligent techniques in the service of society. • Introducing the applications of intelligent techniques. • Using search algorithms and heuristic and metaheuristic algorithms. • The graduate profile is an **"Agent"** (i.e., a proactive, autonomous problem-solver capable of perceiving, reasoning, and acting). **Skills** • Knowledge-acquisition skills. • Recall and analysis skills. • Application and development skills.

	• Publishing research and participating in local and international conferences. • Participation in seminars and workshops. • Keeping pace with developments in the field of specialization. • Analyzing AI technologies—their benefits and challenges. • Enabling students to write AI programs and algorithms, solve problems, interpret results, and make optimal decisions.

58. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	4	1, 2	• An introduction to artificial intelligence and neural networks, and the similarities and differences between them and the human nervous system.	Analysis and conclusion	Q & A
2	4	3	• NW (MLP)	Analysis and conclusion	Q & A
3	4	3-5	• BACK PROPAGATION NETWORK	Analysis and conclusion	Quiz
4	4	5	• Metaheuristic	Theoretical & practical	Q & A
5	4	5, 6	• PSO	Practical & theoretical	Assignment
6	4	5, 6, 7	• PSO	Lecture Lab.	Assignment
7		8, 9	• GA		
8	4	Midterm Exam			
9	4	8, 9	GA	Lecture Lab.	Assignment
10	4	10, 11	• K-means	Lecture Lab.	daily exams
11	4	12, 13	• Fuzzy logic	Lecture Lab.	Assignment
12	4	14	• Example of fuzzy logic	Lecture Lab.	Assignment
13	4	Project discussion			
14	4	Project discussion			
15	3		• Final exam		

59. Course Evaluation=100	
• **Theoretical:** the score out of 35 according to the tasks assigned to the student such as daily preparation=4, daily oral=2, Term tests=25, reports=4. • **Practical:** the score out of 15 according to the tasks assigned to the student such as daily preparation=3, Term tests =8, project=4. • **Final written exams:** 50	
60. Learning and Teaching Resources	
Required textbooks	• *Artificial Intelligence: A Modern Approach* – Stuart Russell & Peter Norvig • Computational Intelligence Synergies of Fuzzy Logic,Neural Networks, and Evolutionary Computing • Neural Networks, Fuzzy Systems and Other Computational Intelligence Techniques for Advanced process control • *Machine Learning* – Tom Mitchell
Main references	
Recommended books and references	
Electronic References, Websites • https://www.tensorflow.org • https://scikit-learn.org	

Course Description Form

61. Course Name:
Software Engineering tools
62. Course Code:

NVITSW3529	
63. Semester / Year:	
2026	
64. Description Preparation Date:	
1/9/2026	
65. Available Attendance Forms:	
On-campus lectures	
66. Number of Credit Hours (Total) / Number of Units (Total)	
2 hours theoretical, 2 hours lab/week	
67. Course administrator's name (mention all, if more than one name)	
Name: Asst. Lect. Safa Sabah Mohammed Email: safa.sabah@uoninevah.edu.iq	
68. Course Objectives	
Course Objectives	• Articulate the concepts and importance of software engineering tools and their role in the software development process. • Identify and categorize different types of software engineering tools, with a deep dive into two main categories: Software Configuration Management (SCM) and Software Project Management (SPM). • Describe the functionality and application of Upper CASE, Lower CASE, and Cross life-cycle CASE tools, while understanding their building blocks and how they are used in organizations to improve development processes. • Gain hands-on proficiency in using Git for software version control to manage changes on projects, while also understanding the different types of version control systems. • Effectively use a project management tool like ProjectLibre to plan and manage software projects, including constructing and interpreting Gantt and PERT charts. • Understand project planning techniques, including the ability to construct and interpret various charts. • Understand and apply ethical principles related to intellectual property, data security, and the responsible use of software tools in professional settings.
69. Teaching and Learning Strategies	
Strategy	• Presentation • Discussion

	• Practical computer applications (Lab works), • Training • Self-learning • Teamwork

70. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	2 hours-theoretical 2 hours-lab	Understand the course structure and objectives.	Introduction to the course	- Presentations - Discussion	Interactive Q&A
2	2 hours-theoretical 2 hours-lab	Students will be able to identify various types of software applications, recognize software engineering tools, and gain hands-on experience by installing Git and exploring essential CMD commands.	- Software engineering - Software engineering tools - Introduction to Git	- Presentations - Discussion - Lab works	Short Answer Questions
3	2 hours-theoretical 2 hours-lab	Students will be able to explain the significance of software-engineering tools in enhancing productivity, code quality, collaboration and overall efficiency and exploring the key features and commands of Git	- The importance of SE tools - Building blocks for CASE - Categories of Software engineering tools - Git commands	- Presentations - Discussion - Lab works	Short Answer Questions
4	2 hours-theoretical 2 hours-lab	Students will be able to explain the core concepts of Software Configuration Management (SCM), using Git to inspect a project's history, apply commits for file changes, and review code modifications.	- Software Configuration Management Tools -Tracking and Modifying Changes using Git	-Presentations -Training - Discussion	Weekly exam
5	2 hours-theoretical 2 hours-lab	Students will be able to articulate the core principles of version-control systems, critically compare their different models, and apply branching and merging techniques for parallel development.	- Introduction to Version control systems -Introducing various types of Version control systems - Working with Git Branches	- Presentations - Discussion - Training	Weekly exam
6	2 hours-theoretical 2 hours-lab	Students will be able to present software-engineering management tools for project planning and tracking while understanding how to identify, read, and resolve merge conflicts in Git	- Software Engineering Management Tools - Git Merge Conflicts and Resolution	- Presentations -Discussion -Training	Weekly exam

7	2 hours-theoretical 2 hours-lab	Students will be able to apply principles of Time Management and Scheduling to manage and prioritize software development tasks effectively - Applying different undo mechanisms in Git, students will learn to safely revert changes and correct their project history.	-Time Management and Scheduling - Git Undoing and Reverting	-Presentations -Discussion -Training	Weekly exam
8	2 hours-theoretical 2 hours-lab	Students will be able to define project activities and apply the Critical Path Method (CPM) to determine the critical path for a software project, in addition to understanding the basic concepts of ProjectLibre and its role in software project management.	- Defining Activities and Critical Path Method (CPM) - Introduction to ProjectLibre	-Presentations -Discussion -Training	- Weekly exam - Seminar - Discussions
9	2 hours-theoretical 2 hours-lab	Students will be able to create and interpret Gantt charts to effectively schedule and track project activities, dependencies, and progress, in addition to identifying a project's core components: tasks, resources, and costs, using ProjectLibre.	- Gantt Charts - Starting Project in ProjectLibre with the core components of a project	- Presentations - Discussion - Training	- Weekly exam - Seminar - Discussions
10	2 hours-theoretical 2 hours-lab	Students will be able to apply PERT charts to effectively, manage project schedules and assess potential timelines by analyzing task dependencies and accounting for time uncertainty, while also defining project resources and establishing dependencies within a project plan using ProjectLibre.	- PERT Charts - project plan - Identify the project resources, project's high-level tasks and task dependencies	- Presentations - Discussion - Training	Midterm exam
11	2 hours-theoretical 2 hours-lab	Students will be able to present tools from the Requirements, Design, and Construction phases, discussing their roles within the software development lifecycle, in addition to assigning resources and decomposing complex tasks into smaller units.	- Software Requirements Tools - Software Design Tools - Software construction Tools - Assign project resources to appropriate tasks -Elaborate and decompose tasks	- Presentations - Discussion - Training	Weekly exam
12	2 hours-theoretical 2 hours-lab	Students will be able to describe the roles of tools used in the Testing and Maintenance phases, in addition to utilizing various task views such as Gantt, Network, WBS, and Task Usage in ProjectLibre.	- Software Testing Tools - Software Maintenance Tools - Tasks - Views and Operations	-Presentations -Discussion -Training	Weekly exam
13	2 hours-theoretical	Students will be able to present engineering process	- Software Engineering Process Tools	-Presentations -Discussion	Weekly exam

	2 hours-lab	and quality assurance tools, explaining their contributions across the software development lifecycle, in addition to performing common task operations in ProjectLibre.	- Software Quality Tools - Task Operations	-Training	
14	2 hours-theoretical 2 hours-lab	- Reflecting on the shift from traditional software development to modern CASE tool-based approaches - Students will be able to execute common resource operations in ProjectLibre.	- Traditional Systems Development versus CASE - Resource Operations	Presentations Discussion	discussions
15	2 hours-theoretical 2 hours-lab	Consolidating knowledge in software engineering tools	- Review and preparation to final exam	Discussion	

71. Course Evaluation

Weekly written exam = 10, seminar= 10 , lab= 10 , in-class assignments= 5, class participation=5, midterm exam =10
final exam= 50

72. Learning and Teaching Resources

Required textbooks (curricular books, if any)	
Main references (sources)	• Software Engineering Body of Knowledge (SWEBOK), IEEE Computer Society • Software Engineering: A Practitioner's Approach, Roger S. Pressman ,fifth edition • Software Engineering: A Practitioner's Approach, Roger S. Pressman ,nineth edition • Software Engineering 10th Edition , Ian Sommerville
Recommended books and references (scientific journals, reports...)	
Electronic References, Websites	• Git https://git-scm.com/ • ProjectLibre https://www.projectlibre.com/

Course Description Form

Course Title	Object-Oriented Programming 2
Course Code	NVITSW3541
Semester / Level	Fourth Semester / Level 3
Semester Preparation Date	22/09/2026
Available Attendance Forms	In-person (Theoretical, Practical, and Applied)
Number of Credit Hours (Total) / Number of Units (Total)	150 hours / 6 ECTS units
Course administrator's name	Name: Mustafa Nabeel Salem Email: mustafa.nabeel@uoninevah.edu.iq

Course Objectives

This course aims to achieve a set of cognitive, skill-based, and affective objectives for the student, as follows:

Cognitive Objectives	**Skill Objectives**	**Affective Objectives**
Understanding the fundamental principles of Object-Oriented Programming (Encapsulation, Inheritance, Polymorphism).	Designing and implementing classes, defining their properties and functions, and creating objects.	Appreciating the importance of Abstraction in building maintainable and scalable code.
Comprehending the concepts of Inheritance and how to create new classes based on existing ones.	Applying Polymorphism to allow objects of different classes to be treated interchangeably.	Adhering to good practices in hiding implementation details to maintain data integrity.
Recognizing the types of relationships between classes (Association, Aggregation, Composition).	Developing skills in Exception Handling to ensure program robustness.	Fostering logical thinking in designing flexible and extensible software solutions.
Understanding the mechanisms of File I/O and Serialization.	Creating Graphical User Interfaces (GUIs) using object-oriented techniques.	

Teaching and Learning Strategies

Strategy	Description
Theoretical Lectures	Presenting the core concepts and theoretical frameworks of the subject.
Problem-Solving Sessions	Dedicated sessions for discussing and solving problems and exercises to deepen understanding.
Practical Applications	Lab sessions to apply theoretical concepts to real-world programming problems.

Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	2 Theory + 2 Practical	- Understand the basic principles of OOP. - Ability to create and implement a simple class.	Theory: Introduction to Object-Oriented Programming (OOP). Practical: Implementing a Simple Class.	Lecture and Practical Application	Assignment
2	2 Theory + 2 Practical	- Grasp the concepts of Encapsulation and Abstraction. - Apply data hiding.	Theory: Encapsulation and Abstraction. Practical: Applying Encapsulation principles and data hiding.	Lecture and Practical Application	Quiz + Lab Work
3	2 Theory + 2 Practical	- Ability to use abstract classes and interfaces to achieve abstraction.	Theory: Encapsulation and Abstraction.	Lecture and Practical Application	Assignment

			Practical: Implementing Abstraction using abstract classes.		
4	2 Theory + 2 Practical	- Understand the concept of Inheritance. - Build "is-a" relationships between classes.	Theory: Inheritance. Practical: Building a class hierarchy using Inheritance.	Lecture and Practical Application	Quiz + Lab Work
5	2 Theory + 2 Practical	- Apply Method Overriding. - Use the super keyword to access the parent class.	Theory: Inheritance. Practical: Implementing Method Overriding and using super.	Lecture and Practical Application	Assignment
6	2 Theory + 2 Practical	- Understand and apply the concept of Polymorphism in programming.	Theory: Polymorphism. Practical: Applying Polymorphism in practice.	Lecture and Practical Application	Quiz + Lab Work
7	-	- Assess student's understanding of topics covered so far.	Mid-Term Exam	Exam	Mid-Term Exam
8	2 Theory + 2 Practical	- Learn how to handle exceptions and errors to	Theory: Exception Handling.	Lecture and Practical Application	Assignment

		increase program robustness.	Practical: Implementing try-catch blocks to handle errors.		
9	2 Theory + 2 Practical	- Ability to create custom exception types specific to the program.	Theory: Follow-up: Exception Handling. Practical: Creating Custom Exceptions.	Lecture and Practical Application	Lab Work
10	2 Theory + 2 Practical	- Understand how to handle files (File I/O) through reading and writing.	Theory: File I/O and Serialization. Practical: Reading and writing data to and from files.	Lecture and Practical Application	Quiz + Assignment
11	2 Theory + 2 Practical	- Learn how to save and retrieve object states (Object Serialization).	Theory: Follow-up: File I/O and Serialization. Practical: Implementing Object Serialization.	Lecture and Practical Application	Lab Work
12	2 Theory + 2 Practical	- Introduction to GUI programming and building a simple interface.	Theory: Graphical User Interface (GUI) Programming. Practical: Building a simple graphical interface.	Lecture and Practical Application	Assignment

13	2 Theory + 2 Practical	- Add interactive controls (buttons, text boxes) to the GUI.	Theory: Follow-up: GUI Programming. Practical: Adding controls to the graphical interface.	Lecture and Practical Application	Lab Work
14	2 Theory + 2 Practical	- Understand the Event-Driven Programming model and handle user interactions.	Theory: Event-Driven Programming. Practical: Implementing Event Handling.	Lecture and Practical Application	Quiz + Assignment
15	2 Theory + 2 Practical	- Comprehensive review of all OOP concepts and their application in an integrated project.	Theory: Comprehensive review of covered programming principles. Practical: Comprehensive project review.	Comprehensive Review	Report / Project

Course Evaluation

Assessment Method	**Percentage**	**Notes**
Final Exam	50%	Comprehensive written exam at the end of the semester.
Mid-Term Exam	10%	Written and practical exam in the seventh week.
Quizzes	10%	Periodic quizzes (5 or more) may be given at any time.

Assignments	10%	Programming assignments submitted periodically.
Lab Work	10%	Assessment of performance and participation in the lab.
Project / Report	10%	A programming project where the student applies the concepts learned.
Total	**100%**	

Learning and Teaching Resources

Type	**Source**
1- Required Textbooks	- "Head First Object-Oriented Analysis and Design" by Brett D. McLaughlin, Gary Pollice, and David West.
2- Recommended books and references	- "Fluent Python: Clear, Concise, and Effective Programming" by Luciano Ramalho.
3- Electronic references, websites	- GeeksforGeeks (https://www.google.com/search?q=https://www.geeksforgeeks.org/object-oriented-programming-in-python/) - W3Schools (https://www.w3schools.com/python/python_classes.asp) - Coursera / edX (Specialized courses in object-oriented programming)

Course Description Form

73. Course Name:
Web Engineering 1
74. Course Code:
NVITSW3523

75. Semester / Year:
First / 2025-2026
76. Description Preparation Date:
23 / 9 / 2026
77. Available Attendance Forms:
Presense
78. Number of Credit Hours (Total) / Number of Units (Total)
Six
79. Course administrator's name (mention all, if more than one name)
Name: Ali mumtaz Email: ali.aldabbagh@uoninevah.edu.iq
80. Course Objectives

Course Objectives	• Provide students with a solid foundation in web standards, client–server architecture, and protocols. • Introduce requirements engineering, modeling (UML), and software process models adapted for web systems. • Develop practical skills in HTML, CSS, JavaScript, and modern frameworks (Laravel, React, Vue). • Train students to design, implement, and test secure, scalable, and user-centered web applications. • Foster awareness of ethical and professional issues such as accessibility, privacy, and teamwork.

81. Teaching and Learning Strategies	
Strategy	**Knowledge** • Students are encouraged to **engage actively in lectures** by questioning standards, models, and frameworks. • They should **analyze case studies critically**, linking theory to real-world web applications. **Skills**

	• Students are expected to **practice independently in labs**, reinforcing coding and modeling abilities. • They will **collaborate in projects**, showing initiative in design, testing, and deployment. • Through problem-based tasks, they should **adopt a problem-solver mindset**, exploring alternatives before settling on solutions. **Ethics** • Students should **demonstrate respect for privacy and accessibility** in all designs. • They are encouraged to **practice teamwork, integrity, and responsibility** during peer reviews and collaborative activities.

82. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
		A1, A3	roduction to the Web (history, standards, architecture)	Lecture + discussion	Quiz + participation
2		A2, B2	oftware Process Models	ecture + case study	ritten exercise
3		, B2, C2	Requirements Engineering	Lecture + workshop	Assignment uirements doc)
4		A2, B2	UML Modeling	ecture + lab	ab practice + mini-report
5		A1, B1	TML Document Skeleton	cture + coding lab	oding exercise
6		, B1, B3	SS & JavaScript Basics	cture + coding lab	assignment + oral Q&A
7		, B1, B3, C2	vanced Styling & JavaScript sponsive, async)	cture + coding lab	actical lab test
8		-A4, B1–B4	**Midterm Exam**	xam session	ritten midterm exam
9		, A3, B2	Client–Server mmunication & APIs	ecture + case study	Short quiz + analysis

0		, B1, B3	meworks & State Management (React/Vue)	cture + coding lab	b assignment
1		, B1, B4	Laravel Basics MVC + Blade fundamentals)	ecture + lab	ject checkpoint
2		, B3, C2, C3	dvanced Blade Patterns & UI Consistency	cture + peer review	assignment + er evaluation
3		, B2, C1	Security & essibility in Web Apps	Lecture + workshop	signment + lab test
4		, B4, C2	ntegrated Case dies (end-to-end web project)	roject-based learning	oject progress report
5		-A4, B1–, C1–C3	**Final Project, iscussion, and Exam**	oject work + resentations	ect demo + lab ssignments + final exam

83. Course Evaluation

- **Theoretical:** the score out of 30 according to the tasks assigned to the student such as daily preparation=5, daily quiz=5, Term tests=10, reports=4, tasks =6.
- **Practical:** the score out of 20 according to the tasks assigned to the student such as daily preparation=3, Term tests =10, project=7.

Final written exams: 50

84. Learning and Teaching Resources

Required textbooks (curricular books, if any)	• Moreb, M. (2021). *Design and implementation of software engineering for modern web applications*. Academic Press. • Cursa, J. (2022). *HTML, CSS, JavaScript: Become a front-end developer*. Packt Publishing. • Upadhyaya, A. (2023). *Advanced front-end development: Building scalable and high-performance web applications with React*. Packt Publishing. • Stauffer, M. (2023). *Laravel: Up & running* (3rd ed.). O'Reilly Media. • Johnson, R. (2025). *Laravel essentials: Definitive reference for developers and engineers*. HiTeX Press.

Main references (sources)	• Moreb, M. (2021). *Design and implementation of software engineering for modern web applications*. Academic Press.
Recommended books and references (scientific journals, reports…)	• **MDN Web Docs** – HTML, CSS, JavaScript, DOM, APIs. https://developer.mozilla.org/ • **W3C Standards** – HTML, CSS, WCAG accessibility. https://www.w3.org/ • **WebAIM** – Web accessibility guidance. https://webaim.org/
Electronic References, Websites	• **Laravel Documentation** – Official docs for Laravel framework. https://laravel.com/docs/ • **Redux Fundamentals** (Abramov & Clark). https://redux.js.org/tutorials/fundamentals/part-1-overview • **React Documentation** – State management and components. https://beta.reactjs.org/learn/managing-state

Course Description Form

1. Course Name: **Database management system**
2. Course Code: NVIT2306
3. Semester / Year: 2025-2026

4. Description Preparation Date: **31/5/2026**

5. Available Attendance Forms: Attendance-based

6. Number of Credit Hours (Total) / Number of Units (Total) 60 Hours

7. Course administrator's name (mention all, if more than one name)

Name: Inas amjed
Email: inas.amjed@uoninevah.edu.iq

8. Course Objectives

Understanding the fundamentals and models of databases • Designing and analyzing databases. • Writing SQL queries effectively. • Understanding the structure and systems of database management.	• Developing analytical problem-solving skills. • Promoting professional responsibility and work ethics.

9. Teaching and Learning Strategies

Strategy	• Cooperative learning • Problem-based learning • Presentation

10. Course Structure

Week	**Hours**	**Required Learning Outcomes**	**Unit or subject name**	**Learning method**	**Evaluation method**
1	2	The student will be able to recognize the basic concept databases and the importanc DBMS	DB Introduction	Collaborative Learning	Evaluation Form
2	2	Gain an understanding of the tasks of a database administrator and permission management	Database Administrator	Class Discussion	Short Questions

3	2	Able to distinguish between a database schema and table structure	Database Schema	Lectures	Homework
4	2	Able to explain the difference between a Schema and an Instance	Database Instance	Lectures	Short Quiz
5	2	Able to analyze the structure of database management systems.	DBMS Architecture	Collaborative Learning	Group Report
6	2	Able to explain the concept of data independence.	Data Independence	Class Discussion	Applied Homework
7	2	Able to apply normalization steps to reduce redundancy and improve efficiency.	Data Normalization	Labs	Solving Programming Problems
8	2	Able to use ER diagrams to represent data.	Entity-Relational Model (ERD)	Collaborative Learning	Mini Project
9	2	Able to distinguish between strong and weak entities.	Strong & Weak Entity	Lectures	Short Questions
10	2	Able to analyze real-world examples using ERD.	ERD Examples	Labs	Problem Solving
11	2	Able to develop models using EERD.	Enhanced ERD	Lectures + Lab	Practical Exam

12	2	Able to explain the concepts of specialization and generalization.	Specialization & Generalization	Collaborative Learning	Short Report
13	2	Able to convert ER/EER models to a relational schema.	Relational DB Design (Mapping)	Labs	Practical Project

11. Course Evaluation
Distributing the score out of 100 according to the tasks assigned to the student such as daily preparation, daily oral, monthly, or written exams, reports .... etc

12. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	FUNDAMENTALS OF DATABASE SYSTEMS By: Ramez Elmasri&Shamkant B. Navathe
Main references (sources)	Database System Concepts – Abraham Silberschatz et al. SQL for Data Scientists – Renee M. P. Teate
Recommended books and references (scientific journals, reports…)	ACM Transactions on Database Systems (TODS) IEEE Transactions on Knowledge and Data Engineering (TKDE)
Electronic References, Websites	w3schools.com geeksforgeeks.org tutorialspoint.com. coursera.org edx.org

Course Description Form

13. Course Name:
Software Engineering tools
14. Course Code:
NVITSW3529
15. Semester / Year:

2/2026

16. Description Preparation Date:

31−5−2026

17. Available Attendance Forms:

On-campus lectures

18. Number of Credit Hours (Total) / Number of Units (Total)

2 hours theoretical, 2 hours lab/week
Total = 4

19. Course administrator's name (mention all, if more than one name)

Name: Asst. Lect. Safa Sabah Mohammed
Email: safa.sabah@uoninevah.edu.iq

20. Course Objectives

Course Objectives	• The Concepts and Importance of Software engineering tools • The categories of Software engineering tools • Upper CASE tools • Lower CASE tools • Cross life-cycle CASE tools • Software Quality Tools • Building blocks of CASE tools • Version control systems • The use of CASE tools in Organizations

21. Teaching and Learning Strategies

Strategy	• Presentation • Discussion • practical computer applications (Lab works), • training • self-learning • teamwork

22. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	2 hours-theoretical 2 hours-lab	Understanding Course Structure	Introduction to the course	Presentations, Discussion	Interactive Q&A
2	2 hours-theoretical 2 hours-lab	-Identifying types of software applications	- Software engineering - Software engineering tools	Presentations, Discussion,	Short Answer Questions

		- Recognizing tools for software engineering - Installing git and explore CMD commands	- Introducing git		
3	2 hours-theoretical 2 hours-lab	- Explain the significance of software engineering tools in enhancing productivity, code quality, collaboration, and the overall efficiency of the software development process - explore key features of git	- The importance of SWE tools - Categories of Software engineering tools - git commands	- Presentations - Discussion - Lab works	Short Answer Questions
4	2 hours-theoretical 3 hours-lab	-Present tools from Requirements, Design, and construction phases and discuss their roles within the software development lifecycle. - deep dive into git commands	- Software Requirements Tools - Software Design Tools - Software construction Tools - Git commands	-Presentations -training - Discussion	Weekly exam
5	2 hours-theoretical 2 hours-lab	-Present tools from Testing, and Maintenance phases and discuss their roles within the software development lifecycle. -Applying version control system to manage code changes.	- Software Testing Tools - Software Maintenance Tools - using Git in controlling changes of website project	- Presentations - Discussion - Training	Weekly exam
6	2 hours-theoretical 2 hours-lab	- Present Configuration Management tools and discuss their roles within the software development lifecycle - manage code changes using version control system	-Software Configuration Management Tools - Git examples on website project	- Presentations -Discussion -training	Weekly exam
7	2 hours-theoretical 2 hours-lab	Present Engineering Management tools and discuss their roles within the software development lifecycle	- Software Engineering Management Tools - applying various git commands on website project	-Presentations -Discussion -training	Weekly exam
8	2 hours-theoretical 2 hours-lab	-Present tools from Engineering process and discuss their roles within the software development lifecycle -Teamwork, -Presentation skills, -self-learning	- Software Engineering Process Tools - Branching in git	-Presentations -Discussion -training	Weekly exam+ seminar+ discussions
9	2 hours-theoretical 2 hours-lab	-Present Software Quality tools and discuss their roles within the software development lifecycle -Teamwork,	- Software Quality Tools - Branching in git	- Presentations - Discussion - training	Weekly exam+ seminar+ discussions

		-Presentation skills, -self-learning			
10	2 hours-theoretical 2 hours-lab	- Understanding core components for developing software engineering tools -Applying Git branching strategies for development.	- BUILDING BLOCKS for CASE -Branching in git	- Presentations - Discussion - training	Mid exam
11	2 hours-theoretical 2 hours-lab	- Understand the importance of version control systems. - Commits to professional responsibility in using code within GitHub - self-learning	- Introduction to Version control systems - GitHub overview	- Presentations - Discussion - training	Weekly exam
12	2 hours-theoretical 2 hours-lab	- Describes version control systems: local, centralized, and distributed environments - Respects intellectual property and follows licensing rules	- introducing various types of Version control systems -exploring GitHub features	Presentations Discussion training	Weekly exam
13	2 hours-theoretical 2 hours-lab	-Understanding organizational benefits to enhance software development by using software engineering tools. -Maintains integrity and transparency when using collaborative development tools	The Use of CASE in Organizations - GitHub repository options	Presentations Discussion training	Weekly exam
14	2 hours-theoretical 2 hours-lab	-Reflecting on traditional software development methods versus modern CASE tool-based approaches	- Traditional Systems Development and CASE - GitHub repository options	Presentations Discussion	discussions
15	2 hours-theoretical 2 hours-lab	Consolidating knowledge in software engineering tools	Review and preparation to final exam	Discussion	

23. Course Evaluation

Weekly written exam = 10, seminar= 10 , class work = 5,
mid exam =25/ theoretical = 15, lab= 10
final exam= 50

24. Learning and Teaching Resources

Required textbooks (curricular books, if any)	Software Engineering Body of Knowledge (SWEBOK), IEEE Computer Society
Main references (sources)	-Software Engineering Body of Knowledge (SWEBOK), IEEE Computer Society

	-Software Engineering: A Practitioner's Approach, Roger S. Pressman ,fifth edition
Recommended books and references (scientific journals, reports…)	
Electronic References, Websites	-Git https://git-scm.com/ -GitHub https://github.com/

Course Description Form

25. Course Name:
Data Communications and Computer Networks
26. Course Code:
NVITSW3531
27. Semester / Year:

First /Third

28. Description Preparation Date:

31/05/2026

29.Available Attendance Forms:

Presence

30.Number of Credit Hours (Total) / Number of Units (Total)

Six

31. Course administrator's name (mention all, if more than one name)

Name: Inas Amjed Mohammed ALMANI
Email: inas.amjed@uoninevah.edu.iq

32.Course Objectives

Course Objectives	1. . Identify network protocols and models, such as OSI and TCP/IP. 2. 2. Understand the functions and responsibilities of each layer in the OSI model. 3. 3. Explore different transmission media, including guided (twisted pair, coaxial, optical fiber) and unguided (electromagnetic spectrum, line-of-sight, satellite, wireless LANs).

33. Teaching and Learning Strategies

Strategy	A- Knowledge and Understanding A1 includes an understanding of theoretical concepts and their application A2 a comprehensive understanding of data communications A3 an understanding of networking protocols and network management B - Subject-Specific Skills B1 - The student applies the simulation model to a real-life situation B2 - The student is able to conduct the simulation C- Thinking Skills C1 - The student will address a real-life problem C2 - The student will combine theoretical knowledge with practical application C3 - The student will develop the skills and knowledge necessary for success in the field of data communications and networks.

34. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	4	Data communications Basics	• Introduction, Data Components, Simplex, Duplex, Half-Duplex, Signals: Analog and Digital	Analysis and conclusion	Assignment

			• Modulation and Types		
2	4	Introduction to computer networks	• Definition of networks, their uses, types of networks: LAN, WAN, MAN, Extra-Net, Intra-Net, Inter-Net, network model	Theoretical	Writing reports
3	4	Data communications and networks	• Data Communications and Networking • Introduction to Protocols and Network Models (OSI and TCP/IP)	Theoretical	Homework
4	4	Physical layer	• Introduction, design issues, and physical layer tasks. • ● Transmission media: • Guided: twisted-pair cables, coaxial cables, optical fibers. Unguided: electromagnetic spectrum, line-of-sight, satellites, IEEE 802.11 wireless LAN standards.	Lecture Lab.	Weekly exams
5	4	Data Link Layer	• Introduction to the Data Link Layer • Error Detection and Correction • Example of data link protocols: HDLC and PPP.	Lecture Lab.	Assignment
6	4	Network Layer Network Routing and IP Addressing	• Introduction to the Network Layer and its Design Issues • Hardware: Routers, Gateways	Lecture Lab.	Assignment

			• Internet Protocol: IPv4 Frame Format, IP Addresses and Classes, Subnetting and Subnet Masks		
7		Midterm Exam	•		
8	4	Implementing TCP and UDP socket programming	Services Provided to the Upper Layer ● Transport Protocols: TCP, UDP, SCTP ● Ports and Sockets	Lecture Lab.	Assignment
9	4	Application Layer Implementing TCP and UDP socket programming	DHCP, DNS, HTTP, SMTP PROXY, FTP	Lecture Lab.	Assignment
10	4	Application Layer-Continued Application Layer Protocols	Application Layer Protocols • ● Implementing client and server applications using socket programming	Lecture Lab.	daily exams
11	4	Network Management	Network Management Introduction, components, and the Internet management framework. • Configuring network security measures (e.g., access control lists, firewall rules)	Lecture Lab.	Assignment
12	4	Network Security	• Introduction, targets, attacks and countermeasures: mapping, packet sniffing, spoofing, DDoS attacks, hijacking.	Lecture Lab.	Assignment
13	4	Network layer security	• Configure VPN tunnels for secure connections • IPsec, VPN.	Lecture Lab.	daily exams

14	4	Wireless LAN Security	• Wireless Network Performance and Interference Analysis • WEP, WPA. Firewalls	Lecture Lab	
15		Final Exam	•		

35. Course Evaluation=100
• **Theoretical:** the score out of 35 according to the tasks assigned to the student such as daily preparation=4, daily oral=2, Term tests=25, reports=4. • **Practical:** the score out of 15 according to the tasks assigned to the student such as daily preparation=2, Term tests =8, project=4. • **Final written exams:** 50

36. Learning and Teaching Resources	
Required textbooks	1. A Computer Networking -1 Top-Down Approach 2 Computer Networks
Main references	"Introduction to Data Communications Networking
Recommended books and references	
Electronic References, Websites	https://www.netacad.com

Course Description Form

37. Module Name:
Computer Architecture
38. Module Code:
NVITSW3522

39. Semester / Year:
2025-2026
40. Description Preparation Date:
11/08/2026
41. Available Attendance Forms:
Excel Sheet prepared by the Dep
42. Number of Credit Hours (Total) / Number of Units (Total)
43. Module's administrator's (mention all, if more than one name)
Name: Radwan Basheer Email :radwanbasher@uoninevah.edu.iq
44. Module's Objectives

Module's Objectives	Understanding the computer architecture which contains: • The basic memory unit (flip-flop) • The memory design and addressing • The microprocessor basic internal units • Basic memory addressing between the CPU the memory banks • The microprocessor addressing modes form 8086 to Pentium processor • The 80386 to Pentium processor protected mode addressing.

45. Teaching and Learning Strategies	
Strategy	• Students are taught how the CPU and memory interact duri memory allocation, including the roles of cache and registers optimizing performance. The course emphasizes understandi memory protection mechanisms and privilege levels to ensu secure and efficient system operation. These concepts a demonstrated through theoretical explanations, analogies, a hands-on lab exercises that simulate real hardware behavior. • Understand and comprehend the mechanism of designing lar memory from smaller available memories. • Learn the addressing mechanism in computers and methods memory protection.

46. Module Structure **Theory and Lab**

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	4	Understand Boolean algebra basics and Karnaugh maps	Review of Boolean Algebra & K-map	Lecture, discussion	Quiz / Class activity
2	4	Explain logic gates and design combinational logic circuits	Logic Circuits & Combinational Logic	Lecture, examples, demo	Quiz / Assignment
3	4	Describe flip-flops and their role as basic memory units	Sequential Logic Circuits (Flip-flops)	Lecture, lab practice	Lab exercise
4	4	Understand multiplexers, demultiplexers, decoders, and encoders	Multiplexers and Decoders	Lecture, practical examples	Quiz / Assignment
5	4	Explain shift registers and types (SISO, PISO, SIPO, PIPO)	Registers	Lecture, lab	Lab exercise
6	4	Understand RAM organization, address decoding, and memory expansion	Memory Units	Lecture, hands-on lab	Lab exercise
7	4	Describe memory hierarchy, cache memory, multiprogramming, and cache performance	Memory Organization & Cache	Lecture, examples, demo	Quiz / Assignment
8	4	Evaluate knowledge of weeks 1-7	Mid-term Exam	Written & practical exam	Mid-term exam
9	4	Understand cache memory organization and cache mapping techniques	Cache Memory Organization	Lecture, demo, discussion	Quiz / Assignment
10	4	Explain cache write policies and associative memory concepts	Cache Write Policies & Associative Memory	Lecture, examples, lab	Lab exercise
11	4	Describe virtual memory concepts including paging and segmentation	Virtual Memory, Paging & Segmentation	Lecture, coding practice	Assignment
12	4	Practice virtual memory management techniques	Virtual Memory Tutorial	Lecture, practical exercises	Lab exercise
13	4	Apply memory management models and paging/segmentation addressing	Programming Model & Memory Management	Lecture, coding lab	Lab exercise
14	4	Review and prepare for the final exam	Exam Preparation	Review sessions, Q&A	—
15	4	Assess ove knowledge and sl	Final Exam	Written practical exam	Final exa

47. Module Evaluation

Formative Assessment	Quizzes	2 quizzes	10%
	Assignments	2 assignments	10%

	Projects / Labs	2 projects/labs	20%
Summative Assessment	Midterm Exam	2 hours	10%
	Final Exam	3 hours	50%

48. Learning and Teaching Resources	
Required textbooks (curricular books, if any)	Mano, M. M., Abel, P. (2005). Comput System Architecture. United Kingdo Pearson Education, Limited.
Main references (sources)	Instructor-Prepared Materials
Recommended books and references (scientific journals, reports…)	Abd-El-Barr, M., & El-Rewini, H. (200 Fundamentals of computer organizati and architecture. John Wiley & Sons.
Electronic References, Websites	Geeksforgeeks: https://www.geeksforgeeks.org

Course Description Form

85. Course Name:
Software Systems
86. Course Code:
NVITSW2306

87. Semester / Year:
First/ Second
88. Description Preparation Date:
22/09/2026
89.Available Attendance Forms:
Presence
90.Number of Credit Hours (Total) / Number of Units (Total)
Five
91. Course administrator's name (mention all, if more than one name)
MohammedTarik mohammed.tariq@uoninevah.edu.iq
92.Course Objectives

Course Objectives	• To understand the relationship between system software and machine architecture. • To understand the processing of an HLL program for execution on a computer. • To understand the process of scanning and parsing. • To know the design and implementation of assemblers, macro processor, linker and compiler • To have an understanding of loader, system software tools. • To understand and know the working of device drivers • Be able to compare various system software related to the given system • Be able to understand the concepts required to develop the system software

93. Teaching and Learning Strategies	
Strategy	• At the start of course, the course delivery pattern, prerequisite of the subject will be discussed. • Lectures will be conducted with the aid of multi-media projector, black board, OHP etc. • Attendance is compulsory in lecture and laboratory which carries 10 marks in overall evaluation. • One internal exam will be conducted as a part of internal theory evaluation. • Assignments based on the course content will be given to the students for each unit and will be evaluated at regular interval evaluation. • Surprise tests/Quizzes/Seminar/tutorial will be conducted having a share of five marks in the overall internal evaluation. • The course includes a laboratory, where students have an opportunity to build an appreciation for the concepts being taught in lectures. • Experiments shall be performed in the laboratory related to course contents.

94. Course Structure

Week	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	5	Be able to compare various system software related to the given system	**Introduction to System Software**	Analysis and conclusion	Homework
2	5	Be able to understand the concepts required to develop the system software	**Data Structures for Language Processing:** • Allocation Data Structures.	Theoretical	Open source quiz

3	5	Be able to make proper use of system software tools	**Assemblers:** • Elements of Assembly Language Programming	Theoretical Tutorial	Homework
4	5	Be able to make proper use of system software tools)	**Assemblers:** • A Simple Assembly Scheme	Theoretical Practice Tutorial	Weekly exams
5	5	Be able to make proper use of system software tools	**Assemblers:** • Pass Structure of Assemblers	Theoretical Practice Tutorial	Assignment
6	5	Be able to make proper use of system software tools	**Assemblers:** • Design of a Two Pass Assembler	Theoretical Practice Tutorial	Assignment
7	5	Be able to make proper use of system software tools	**Assemblers:** • A single pass Assembler for IBM PC.	Theoretical Practice	Weekly exams
8	5	Understand the concepts of Macro Expansions and Gain the knowledge of Editing Processes	**Macros and Macro Processors:** • Macro Definition and Call • Macro Expansion	Theoretical Practice	Assignment
9	5	Understand the concepts of Macro Expansions and Gain the knowledge of Editing Processes	**Macros and Macro Processors:** • Nested Macro Calls	Theoretical Practice Tutorial	Assignment
10	5	Understand the concepts of Macro Expansions and Gain the knowledge of Editing Processes	**Macros and Macro Processors:** • Advanced Macro Facilities	Theoretical Practice Tutorial	Term test
11	5	Understand the concepts of Macro Expansions and Gain the knowledge of Editing Processes	**Macros and Macro Processors:** • Design of a Macro Preprocessor.	Theoretical Practice Tutorial	Assignment
12	5	Understand the concepts like interrupts, deadlock, memory management and file management	**Linkers and Loaders:** • Introduction to linkers • Relocation and Linking Concepts	Theoretical Practice Tutorial	Assignment

13	5	Understand the concepts like interrupts, deadlock, memory management and file management.	**Linkers and Loaders:** • Design of a Linker	Theoretical Practice Tutorial	Weekly exams
14	5	Understand the concepts like interrupts, deadlock, memory management and file management.	**Linkers and Loaders:** • Self-Relocating Programs	Theoretical Practice Tutorial	Assignment
15	5	To understand and know the working of device drivers	**Linkers and Loaders:** • A Linker for MS-DOS • Linking for Overlays and Loaders	Theoretical Practice Tutorial	Assignment
16	5	Review	• **Preparatory week before the final Exam**	Tutorial	Evaluation form

95. Course Evaluation=100

- **Theoretical:** the score out of 35 according to the tasks assigned to the student such as daily preparation=4, daily oral=2, Term tests=25 , reports=4.
- **Practical:** the score out of 15 according to the tasks assigned to the student such as daily preparation=2, Term tests =8, project=4.
- **Final written exams:** 50

96. Learning and Teaching Resources

Required textbooks	• D. M. Dhamdhere, "Systems Programming and Operating Systems", Second Revised Edition, Tata McGraw-Hill, 1999.
Main references	• Leland L. Beck, "System Software – An Introduction to Systems Programming", 3rd Edition, Pearson Education Asia, 2000. • Santanu Chattopadhyay, "System Software", Prentice-Hall India, 2007. • Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman, "Compilers: Principles, Techniques, and Tools",2nd Edition, Pearson Education Asia.
Recommended books and references	• Achy8utS.Godbole,OperatingSystems,TMH,2002. • JohnJ.Donovan,SystemsProgramming,TMH,1991.

Electronic References, Websites	▪ https://online.stanford.edu/programs/software-systems-graduate-certificate

Course Description Form

49. Course Name:
Information security
50. Course Code:
NVITSW4720
51. Semester / Year:
First /Fourth

52. Description Preparation Date:
20/8/2025
53.Available Attendance Forms:
Presence
54.Number of Credit Hours (Total) / Number of Units (Total)
Six
55. Course administrator's name (mention all, if more than one name)
Name: Ali Mohsin Ahmed Email: ali.mohsin@uoninevah.edu.iq
56.Course Objectives

Course Objectives	**Teaching students how to protect data** 4. Either by using different encryption methods or by selecting the appropriate protection technique. 5. Describing possible attacks and studying defense mechanisms as needed.

57. Teaching and Learning Strategies	
Strategy	**Knowledge** a. Understand the fundamental concepts of information security, such as encryption, digital signatures, protection against viruses, and cyberattacks. b. Ability to identify security threats and protect against them, including applying the necessary measures to reduce risks to information systems. c. Ability to analyze and test network security, respond to cyberattacks, and solve potential security threat issues within an information system. **Skills** a. Ability to design a reliable and secure information system, identifying the appropriate security requirements for its use. b. Ability to establish and implement systems for managing security policies and practices in a systematic way to ensure compliance with security requirements and related constraints. **Values** • **Integrity:** Do not manipulate or misuse information. Reject the use of acquired skills for illegal or malicious purposes. • **Confidentiality:** Respect the privacy of the information handled, whether belonging to individuals or institutions. Commit to not disclosing information without authorization. • **Responsibility:** Take ethical and professional responsibility when dealing with sensitive systems and data. Recognize the consequences of actions that may harm security or users. • **Respect:** Respect the rights of users, colleagues, and local and international laws.

58. Course Structure					
Week	**Hours**	**Required Learning Outcomes**	**Unit or subject name**	**Learning method**	**Evaluation method**
1	4	1	• Introduction to data security	Analysis and conclusion	Q & A
2	4	2, 3, 4	• Security attack	Theoretical	Q & A
3	4	5	• Classical transposition method for cryptography	Theoretical & practical	training
4	4	6	• Classical substitution method for cryptography	Theoretical & practical	training
5	4	7	• Classical substitution method for cryptography	Lecture Lab.	Assignment
6	4	8	• Classical substitution method for cryptography	Lecture Lab.	Assignment
7		Midterm Exam	•		
8	4	9	Block cipher	Lecture Lab.	Q & A
9	4	9	Data Encryption Standard (DES)	Lecture Lab.	Assignment
10	4	10	• Public Key Cryptography	Lecture Lab.	daily exams
11	4	10	• Public Key Cryptography	Lecture Lab.	Assignment
12	4	11	• RSA algorithm	Lecture Lab.	Assignment
13	4	12	• Authentication in RSA	Lecture Lab.	daily exams
14	4		• Revision	Lecture Lab	
15			• Final exam		
59. Course Evaluation=100					

• **Theoretical:** the score out of 35 according to the tasks assigned to the student such as daily preparation=4, daily oral=2, Term tests=25, reports=4. • **Practical:** the score out of 15 according to the tasks assigned to the student such as daily preparation=2, Term tests =8, project=4. • **Final written exams:** 50

60. Learning and Teaching Resources

Required textbooks	cryptography-and-network-security_-principles-and-practice-7th-global-edition 2. Increasing the accuracy of Melanom 3- Handbook of Applied Cryptography
Main references	
Recommended books and references	
Electronic References, Websites	