Module Handbook
for Software Engineering

April 2026

Hengxing University

Contents

Introduction to Software Engineering Program 1
C Language Programming 4
Discrete Mathematics 7
Data Structures and Algorithms 9
Principles of Operating Systems 11
Principles of Database Systems 14
Digital Logic 17
Principles of Computer Composition 19
Computer Networks 22
Web Front-End Development Technology 24
Software Engineering 27
Object-Oriented Programming 30
Principles of Compilation 32
Software Architecture 35
System Analysis and Design 37
Software Testing 39
Software Project Management 41
Java Web Application Development 43
C++ Object-Oriented Programming 46
Enterprise-level Java Development 48
Android Mobile Application Development 50
Python Programming 52
Big Data Collection and Processing 55
Introduction to Artificial Intelligence 58
Computer Science English 60
Fundamentals of Information Security Technology 64
Internet of Things Applications 67
Innovative Practices in AIGC Applications 70
Internet+ Innovative Application 73
Basic Training in Programming 76
Basic Training in Data Structures and Algorithms 78

Basic Training in Software Front-end Development 81
Basic Training in Database Application 84
Linux System Programming Training 87
Comprehensive Training in Software Requirement Analysis 89
Comprehensive Training in Object-Oriented Programming 91
Comprehensive Training in Java Web Application Development 94
Software Engineering Practice 97
Comprehensive Training in System Analysis and Design 100
Comprehensive Training in Software Testing 102
Comprehensive Training in Enterprise-level Java Application Development 104
Software Engineering Project Management Practice 106
Comprehensive Training in System Development 109
Graduation Internship 112
Graduation Thesis (Design) 115

Introduction to Software Engineering Program

Module name	Introduction to Software Engineering Program		
Code	3080902270		
Semester	Fall, 1st Semester		
Contact person	Liu Zunren,Sun Ge		
Language	Chinese		
Relation to curriculum	Compulsory, Professional Foundation Courses		
Type of teaching, contact hours	Lecture Method, Case Analysis Method, Online Learning Method Teaching Hours:2 hours per week, 8 weeks total,16 hours in total		
Work load	Total Workload (estimated):28 class hours Teaching Hours:16 hours Self-study Hours:12 hours, including: after-class assignments, exam preparation time, etc.		
Credit points	1		
Recommended prerequisites	None		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Through learning the basic concepts and development history of software engineering, students will master the essence of software, the core ideas of software engineering, and software lifecycle models; combined with cases of domestic software (such as HarmonyOS, industrial software) breaking through technological blockades, inspire students' national pride and sense of responsibility to contribute to the development of the nation's software industry.	**R1**
	CLO2	Through this course, students will be able to understand core processes such as software requirements, design, construction, testing, and maintenance; master modern engineering concepts such as Agile development and DevOps; enable students to preliminarily apply engineering thinking to analyze common problems in software projects and enhance their awareness of solving complex engineering problems.	**R2**

	CLO3	By introducing software engineering tools and environments (such as version control with Git, project management tools), preliminarily integrate theory with practice; enable students to select appropriate development models and formulate preliminary project plans based on simple task descriptions; cultivate basic software engineering literacy and lay a solid foundation for further in-depth study of Professional Core Course.	**R3**
	CLO4	Students will be able to communicate effectively in software engineering case seminars and group collaborations; understand the professional ethics and social responsibilities of software engineers (such as intellectual property rights, privacy protection, engineering ethics); demonstrate team collaboration spirit to complete simple software project simulations; cultivate students' expression abilities and teamwork skills, promoting the improvement of students' comprehensive qualities.	**R9**
Content	This course is a professional foundation course designed to strengthen the cultivation of software engineering thinking and professional literacy. This course enables students to understand the disciplinary system and evolutionary history of software engineering, master the basic principles of software lifecycle models and core engineering activities (requirements, design, coding, testing), become familiar with modern engineering methods such as Agile development and DevOps, and based on case studies, select appropriate engineering processes according to project characteristics. It cultivates students' ability to comprehensively apply engineering knowledge to analyze and solve complex software problems; and fosters and enhances students' team collaboration spirit, engineering ethics awareness, and comprehensive qualities of integrating theory with practice. **Knowledge Module 1:** Software Engineering Fundamentals and Process Models (Weight: 3/16, Level: Remember + Understand) **Knowledge Module 2:** Requirements Engineering and Analysis (Weight: 3/16, Level: Understand + Apply) **Knowledge Module 3:** Software Design and Architecture (Weight: 4/16, Level: Understand + Analyze) **Knowledge Module 4:** Software Construction, Testing, and Maintenance (Weight: 4/16, Level: Apply + Analyze) **Knowledge Module 5:** Software Project Management and Professional Ethics (Weight: 2/16, Level: Understand + Evaluate)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is in the form of a course report, accounting for 50% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 15%; (2) regular assignments 15%; and (3) interim tests 20%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.		
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.		

Reading list	1.*Software Engineering: A Practitioner's Approach* (9th Edition), by Roger S. Pressman, Bruce R. Maxim, translated by Zheng Renjie et al., China Machine Press, 2022. 2.*Software Engineering* (10th Edition), by Ian Sommerville, translated by Cheng Cheng et al., China Machine Press, 2018. 3.*Software Engineering*, edited by Li Tong, Higher Education Press, 2019. 4.No Silver Bullet — Essence and Accidents of Software Engineering, Frederick P. Brooks, Jr., IEEE Computer, 1987. 4.Software Engineering in the Era of Artificial Intelligence: Challenges and Opportunities, by Mei Hong, Communications of the Chinese Computer Federation, 2020.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

C Language Programming

Module name	C Language Programming		
Code	3080900242		
Semester	Fall, 1st Semester		
Contact person	Liu Yanfang, Guo Shaosheng, Liu Xing		
Language	Chinese		
Relation to curriculum	Compulsory，Professional Foundational Course		
Type of teaching, contact hours	Lecturing method；Demonstration method；Laboratory experimental method Teaching hours: 4 hours per week, 12 weeks in total, with a total of 48 hours.		
Work load	Total workload (estimated): 84 teaching hours Teaching hours: 48 hours Self-study hours: 36 hours, including after-class assignments, exam preparation time, etc.		
Credit points	3		
Recommended prerequisites	None		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Course Objective 1: Through learning C language programming, students will master the basic knowledge of computer languages, grasp the fundamental methods and thinking of program design, form the basic concepts of programming, and acquire the essential skills of program debugging. This course will enable students to develop preliminary programming capabilities and lay a solid foundation for further study of other professional courses and future engagement in software development.	**R1**
	CLO2	Course Objective 2: Through learning this course, students will master the basic principles of C language and programming methods. They will be able to use mathematics and basic software engineering knowledge to identify and analyze complex programming problems, draw valid conclusions through program design and debugging, and develop the ability to analyze and solve problems.	**R2**

	CLO3	Course Objective 3: Master C language programming methods, be able to design and implement program modules and algorithmic flows that meet requirements, and form solutions to complex problems. Integrate a sense of standardization and innovative thinking into design, take engineering factors such as security and specification into account, and improve software design and development capabilities.	**R3**
	CLO4	Course Objective 4: Master C language development environments and debugging tools. Be able to rationally select programming technologies and tools to complete program analysis, design, implementation and testing. Understand the applicable scenarios and limitations of common development tools, and possess the basic ability to solve software engineering problems using modern tools.	**R5**
Content	Fundamentals of Computer Languages is one of the essential basic skills for computer science students. It plays an indispensable role in cultivating abstract thinking, improving computer literacy, and enhancing understanding of subsequent courses. The C language is platform-independent and powerful, allowing programs to be written for various operating systems such as Windows, Linux, and Unix. It plays a key role in developing students' programming abilities. Through this course, students will master the basic knowledge of computer languages, grasp the fundamental methods and thinking of programming, form basic programming concepts, and acquire essential program debugging skills. It enables students to develop preliminary programming capabilities and lays a solid foundation for further study of other professional courses and future careers in software development. Knowledge Module 1: Program Design and C Language (Weight: 2/48, Level: Memorization + Comprehension) Knowledge Module 2: Algorithms – The Soul of Programs (Weight: 4/48, Level: Comprehension + Analysis + Evaluation) Knowledge Module 3: Basic C Program Design – Sequential Program Design (Weight: 4/48, Level: Memorization + Comprehension + Application) Knowledge Module 4: Selection Structure Program Design (Weight: 6/48, Level: Memorization + Comprehension + Application) Knowledge Module 5: Loop Structure Program Design (Weight: 6/48, Level: Memorization + Comprehension + Application) Knowledge Module 6: Processing Batch Data Using Arrays (Weight: 6/48, Level: Memorization + Comprehension + Application) Knowledge Module 7: Modular Program Design Using Functions (Weight: 6/48, Level: Comprehension + Application + Analysis) Knowledge Module 8: Pointers (Weight: 8/48, Level: Comprehension + Application + Analysis) Knowledge Module 9: Structures and Unions (Weight: 4/48, Level: Memorization + Comprehension + Application) Knowledge Module 10: File Operations (Weight: 2/48, Level: Memorization + Application)		

Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 60% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 40% of the overall grade and includes: (1) class participation 12%; (2) regular assignments 12%; and (3)Lab results: 16%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] C Programming, edited by Tan Haoqiang, Tsinghua University Press, 2023, 5th Edition. [2] C Language Programming, chief edited by Zhao Ji and Wang Caixia, Tsinghua University Press, 2023. [3] Niu Guofeng, Zhu Miaomiao, Li Xin. Process Design and Research on Blended Teaching of C Language Course[J]. Journal of Suzhou University of Technology, 2026, 40(02): 118-124. DOI:10.16101/j.cnki.cn32-1749/z.2026.02.013. [4] Liu Zhihai, Liu Ruijie, Wang Chenglong, et al. C Language Teaching Design Integrating Ideological and Political Education and Case Teaching[J]. China Educational Technology & Equipment, 2026, (06): 83-86. [5] Liang Xiaojie. Design and Application of Micro-lessons Based on Knowledge Integration Teaching Theory in C Language Flipped Classroom in Secondary Vocational Schools[D]. Central China Normal University, 2025.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Discrete Mathematics

Module name	Discrete Mathematics		
Code	3080900229		
Semester	Spring ,2nd Semester		
Contact person	Diao Tianru、Guo Shaosheng		
Language	Chinese		
Relation to curriculum	Compulsory，Professional Foundational Course		
Type of teaching, contact hours	Lecture Method; Group Discussion Method; Laboratory Experiment Method Teaching Hours: 4 hours per week for 12 weeks, totaling 48 hours		
Work load	Total Workload (Estimated): 84 hours Teaching Hours: 48 hours Self-Study Hours: 36 hours, comprising: Pre-class preparation；Homework and assignments；Exam preparation and review		
Credit points	3		
Recommended prerequisites	Advanced Mathematics A I		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Students are able to understand the fundamental concepts of mathematical logic, set theory, graph theory, and other areas within discrete mathematics, and to preliminarily describe, abstract, and represent discrete structures in discrete mathematical problems.	**R1**
	CLO2	Students are able to comprehend the fundamental methods of mathematical logic, set theory, graph theory, etc., analyze their applications in computer science and related fields, and formulate specific problems into computer-recognizable data models.	**R2**
	CLO3	Students are able to perform derivations and proofs on specific problems using mathematical tools, design feasible solutions through abstract thinking, generalization analysis, and logical reasoning, and derive reasonable conclusions.	**R4**
	CLO4	Through the study of the development history of mathematical logic and classic paradoxes in set theory such as the "Barber Paradox," students will recognize the tortuous yet progressive development of mathematics, establish a correct view of learning, be able to generalize existing knowledge and draw inferences from examples, thereby correctly understanding the necessity of independent and lifelong learning.	**R11**

Content	This course is a fundamental core course for the four-year undergraduate program in Software Engineering. Discrete mathematics is a branch of modern mathematics that studies the structure of discrete quantities and their interrelationships. The main content of this course includes mathematical logic (propositional logic and first-order logic), set theory (sets, relations, and functions), and graph theory (graphs and trees).Upon completion of this course, students are expected to master the basic concepts and fundamental analytical methods of discrete mathematics, understand and acquire the descriptive tools and methods necessary for handling discrete structures, and apply common discrete mathematical thinking methods to solve practical problems in the field of computer science, thereby providing the necessary mathematical foundation for subsequent courses in the software engineering curriculum. Knowledge Module 1: Propositional Logic (Weight: 12/32, Levels: Understand + Apply + Analyze + Evaluate) Knowledge Module 2: First-Order Logic (Weight: 6/32, Levels: Understand + Apply + Analyze) Knowledge Module 3: Sets (Weight: 6/32, Levels: Understand + Apply + Analyze) Knowledge Module 4: Binary Relations (Weight: 10/32, Levels: Understand + Apply + Analyze) Knowledge Module 5: Functions (Weight: 2/32, Levels: Understand + Apply) Knowledge Module 6: Graph Theory (Weight: 12/48, Levels: Understand + Apply)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 50% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 15%; (2) regular assignments 15%; and (3) computer experiment 20%. Classroom performance is comprehensively evaluated based on student topic discussions in class, in-class exercises, and learning of digital courses.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	1. Qu Wanling, Liu Tian, Geng Suyun, et al. Discrete Mathematics (4th edition) [M]. Beijing: Tsinghua University Press,2022. 2. Qu Wanling, Liu 1. 2. Qu Wanling, Liu Tian, Geng Suyun, et al. Discrete Mathematics Exercise Solution and Learning Guide (4th edition) [M]. Beijing: Tsinghua University Press,2022. 3. Wang Qingxian, Gu Xiaofeng, Wang Lijie. Discrete Mathematics (Micro-course Edition) [M]. Beijing: Posts and Telecommunications Press,2023. 4. Ma Dianfu. Discrete Mathematics and its Application: Python Modeling and implementation [M]. Beijing: Higher Education Press,2021. 5. Aigner M. Discrete mathematics[M]. American Mathematical Society, 2023. 6. Chen Lijun, Cheng Li. Mixed teaching reform of discrete Mathematics based on OBE concept [J]. Journal of Chifeng University (Natural Science Edition),2024,40(02):78-81. 7. Wu Nan. Reflections on the construction of Ideological and political teaching system of Discrete Mathematics courses for computer majors [J]. Journal of Higher Education, 2019,9(34):174-177.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Data Structures and Algorithms

Module name	Data Structures and Algorithms		
Code	3080900230		
Semester	Spring, 2nd Semester		
Contact person	Liu Yanfang, Guo Ning, Liu Zunren		
Language	Chinese		
Relation to curriculum	Compulsory，Professional Foundational Course		
Type of teaching, contact hours	Lecture Method, Group Discussion Method, Case Analysis Method, Simulation Experiment Method. Teaching Hours: 4 hours per week for 16 weeks, totaling 64 hours		
Work load	Total Workload (Estimated): 112 hours Teaching hours: 64 hours Self-study hours:48 hours (assignments, exam preparation, etc.)		
Credit points	4		
Recommended prerequisites	C Language Programming, Discrete Mathematics		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Understand basic concepts of data structures and algorithms; master logical/storage structures and basic operations; model software engineering problems using mathematical and engineering fundamentals.	**R1**
	CLO2	Analyze time and space complexity; identify core requirements in complex engineering problems; conduct literature research to draw valid conclusions.Apply algorithmic design techniques to develop efficient solutions; evaluate trade-offs between correctness, efficiency, and scalability; validate approaches through systematic testing and empirical analysis.	**R2**
	CLO3	Design solutions for complex software engineering problems; select/design appropriate data structures and algorithms (sorting, searching, graph algorithms); incorporate innovation, social and safety considerations.	**R3**
	CLO4	Conduct engineering research using scientific methods; design experiments; implement and validate algorithms; analyze performance and reach reliable conclusions.	**R4**

Content	This course systematically introduces the fundamental theories and application methods of data structures and algorithms, and cultivates students' ability to select appropriate data structures and design efficient algorithms for practical problems. The main contents include: linear structures (sequential lists, linked lists, stacks, queues), trees and binary trees, graph structures, searching and sorting algorithms, algorithm complexity analysis, etc. Through the integration of theory and practice, students will master the core technologies of data organization and processing in software engineering, improve programming and problem-solving skills, and lay a solid foundation for subsequent software development and system design. Knowledge Module 1: Introduction (Weight: 2/64, Level: Understanding) Knowledge Module 2: Linear Lists (Weight: 8/64, Level: Understanding + Application) Knowledge Module 3: Stacks and Queues (Weight: 8/64, Level: Understanding + Application) Knowledge Module 4: Strings (Weight: 6/64, Level: Understanding + Application) Knowledge Module 5: Arrays and Generalized Lists (Weight: 6/64, Level: Understanding + Application) Knowledge Module 6: Trees and Binary Trees (Weight: 8/64, Level: Understanding + Application + Analysis) Knowledge Module 7: Graphs (Weight: 8/64, Level: Application + Analysis) Knowledge Module 8: Searching (Weight: 8/64, Level: Application + Analysis + Evaluation) Knowledge Module 9: Sorting (Weight: 10/64, Level: Application + Analysis + Evaluation)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 60% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 40% of the overall grade and includes: (1) class participation 12%; (2) regular assignments 12%; and (3) laboratory work 16%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Yan, W. M. Data Structures (C Language Edition). Beijing: Tsinghua University Press. [2] Jin, Y. D. Hello Algorithm. Beijing: Posts & Telecom Press, 2024. [3] Watanabe, Y., Milenkov, N. Illustrated Algorithms and Data Structures. Trans. Zheng, M. Z. Beijing: Posts & Telecom Press, 2024. [4] Akiba, T., Iwata, Y., Kitagawa, M. Challenges in Programming Contests (2nd ed.). Beijing: Posts & Telecom Press, 2013. Papers [5] Wen, Y. K. Research on the Discreteness of Computer Algorithm Design and Data Structures. Information and Computer (Theoretical Edition), 2024, 36(04): 59–61. [6] Zhang, X. L., Zhang, C., Liu, W. R., et al. Research on Key Data Structures in Private Set Operations. Journal of Cryptologic Research, 2024, 11(02): 263–281. DOI: 10.13868/j.cnki.jcr.000679.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Principles of Operating Systems

Module name	Principles of Operating Systems		
Code	3080903255		
Semester	Fall ,3rd Semester		
Contact person	Cao Xiuping, Liu Zunren		
Language	Chinese		
Relation to curriculum	Compulsory，Professional Foundational Course		
Type of teaching, contact hours	Lecture method, demonstration method, group discussion method, case analysis method Teaching duration:4 hours per week for 12 weeks, totaling 48 hours		
Work load	Total Workload (Estimated): 84 hours Teaching duration: 48 hours Self-study hours: 36 hours, including homework and exam preparation time.		
Credit points	3		
Recommended prerequisites	C Language Programming、Data Structures and Algorithms		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1：** Through the study of "Operating System Principles," students will master the fundamental components and operating principles of operating systems, the basic methods for core algorithm design, and operating system interface technologies. By examining the historical development of operating systems and considering current challenges such as difficulties in China's independent operating system development and foreign technological monopolies, this course aims to inspire students' patriotism and their determination to contribute to the advancement of domestically controlled operating systems, thereby solidifying their core engineering knowledge in software engineering.	**R1**
	CLO2	**Course Objective 2：** Through this course, students will master the structure and working principles of operating system kernel resources, as well as methods for expanding system resources. This enables them to design operating system-related functional solutions accurately based on practical engineering requirements, enhances their problem analysis and resolution skills, and cultivates their ability to analyze system-level issues and perform logical reasoning.	**R2**

	CLO3	**Course Objective 3：** By writing core algorithm code within the operating system environment, this approach systematically and deeply integrates theoretical knowledge of operating systems with software programming. It enables students to design appropriate operating system functional modules and develop corresponding programs according to practical engineering requirements, equipping them with the capability to design and develop relevant operating system modules. This lays a solid foundation for students to further pursue advanced studies in complex system development.	**R3**
	CLO4	**Course Objective 4：** During the design and debugging of the operating system course project, students can communicate effectively, learn from each other's strengths, and fully leverage the capabilities of every team member to collaboratively complete operating system-related projects (such as a simple process management system and a memory allocation simulation program). This process enhances students' hands-on practical skills, fosters their innovative thinking, develops their ability to comprehensively apply operating system knowledge to solve real-world engineering problems, and contributes to the overall improvement of their competencies.	**R4**
Content	"Principles of Operating Systems" is a required course for computer science majors, designed to provide a comprehensive and systematic introduction to the architecture, design principles, implementation methods, and technologies of operating systems. The course covers system calls and interfaces, processor scheduling and thread control, synchronization and communication mechanisms, deadlock resolution, memory management based on partitioning, paging, and segmentation along with virtual memory, device management, and file systems. Through this course, students will systematically master the fundamental concepts, key functionalities, operating principles, core technologies, and implementation approaches of operating systems. They will develop the ability to utilize and analyze operating systems, acquire a solid theoretical foundation and technical competence in operating system design, lay a robust groundwork for advanced professional studies, and prepare them to develop various application and system software on operating system platforms. Knowledge Module 1：Operating System Overview (Weight: 2/48, Level: Memory & Understanding) Knowledge Module 2：User Interface (Weight: 2/48; Levels: Memory, Understanding, Evaluation) Knowledge Module 3：Process Management (Weight: 10/48; Level: Understanding-Evaluation) Knowledge Module 4：Processor scheduling (Weight: 12/48, Level: Memory-Evaluation) Knowledge Module 5：Memory Management (Weight: 10/48, Level: Understand-Creat) Knowledge Module 6：File Management (Weight: 6/48, Level: Memory-Analysis) Knowledge Module 7：Device Management (Weight: 6/48, Level: Understanding & Application)		

Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 60% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 40% of the overall grade and includes: (1) class participation 12%; (2) regular assignments 12%; and (3) interim tests 16%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Tang Xiaodan, Liang Hongbing, Zhe Fengping, Tang Ziying. Computer Operating Systems (4th Edition) [m]. Xi 'an: Xi' an University of Electronic Science and Technology Press, 2021. [2] Pang Liping, Yang Fumin. Computer Operating Systems (3rd Edition) [m]. Beijing: People's Posts and Telecommunications Press, 2017. [3] Wu Fan, Liu Gongshen, Wu Chentao. Principles and Implementation of Operating Systems [m]. Beijing: People's Posts and Telecommunications Press, 2024. [4] Ge Yan, Liu Guozhu, Du Junwei, Cao Ling. Research on Diversified Teaching Models for the "Principles of Operating Systems" Course [j]. 2024. [5] Liu Qiang, Zhang Hua. Teaching Reform and Practical Exploration of Operating System Principles [j]. Computer Education, 2022, (10):123–126. [6] Zhang Lei, Wang Xiaochun, Liu Zhiyong. Exploration and Practice of an Industry-Education Integrated Teaching Model for the Operating Systems Course [d]. Computer Education, 2023.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Principles of Database Systems

Module name	Principles of Database Systems		
Code	3080900211		
Semester	Fall ,3rd Semester		
Contact person	Wan Ming'ao， Guo Ning， Zhu Xiaoxia		
Language	Chinese		
Relation to curriculum	Compulsory,Professional Foundational Course		
Type of teaching, contact hours	Lecture Method、Group Discussion Method、Case Analysis Method、Simulation Experiment Method Class hours: 4 hours per week, 12 weeks in total, totaling 48 hours		
Work load	Estimated Total Workload: 84 class hours Teaching Hours: 48 hours Independent Study Hours: 36 hours, including homework, exam preparation and other self-study tasks.		
Credit points	3		
Recommended prerequisites	C Language Programming, Data Structures and Algorithms		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Master the basic concepts, relevant principles, and usage methods of relational databases; grasp the core functionalities of SQL language; understand the usage of commonly used database objects; and be proficient in commonly used database design methods.	**R1**
	CLO2	Able to create databases and their objects, and perform CRUD (Create, Read, Update, Delete) operations; capable of selecting tools based on requirements to model databases and conduct normalized database design.	**R2**
	CLO3	Through database experiments and database application system design, students' professional ethics and craftsmanship spirit are cultivated; through discussions on database security technologies, students' patriotism and sense of mission to contribute to the country through science and technology are inspired.	**R3**

Content	This course introduces and elaborates on the basic theories, fundamental techniques and methodologies of database systems. It requires students to master knowledge of relational data theory, database protection, database design and database management systems. The specific contents include: the history of database technology, data models, architecture of database systems, composition of database systems; relational databases; SQL language; relational systems and query optimization; relational data theory and normalization; database design; transactions, failure recovery and concurrency control; database security and integrity constraints.Through the study of this course, students will acquire theoretical knowledge and practical application abilities in basic database knowledge, database design, database management and maintenance. **Knowledge Modules** **Module 1:** Overview of Database SystemsWeight: 4/48Competence Level: Memorization & Comprehension **Module 2:** Relational DatabasesWeight: 4/48Competence Level: Memorization, Comprehension & Application **Module 3:** Fundamentals of SQL Server DatabaseWeight: 4/48Competence Level: Memorization & Comprehension **Module 4:** Standard Database Language SQLWeight: 24/48Competence Level: Memorization, Comprehension, Application & Analysis **Module 5:** Relational Database Design TheoryWeight: 4/48Competence Level: Memorization, Comprehension & Application **Module 6:** Database DesignWeight: 4/48Competence Level: Memorization, Comprehension, Application & Analysis **Module 7:** Database ManagementWeight: 4/48Competence Level: Memorization, Comprehension & Application
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 60% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 40% of the overall grade and includes: (1) class participation 12%; (2) regular assignments 12%; and (3) laboratory experiments 16%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.

Reading list	1.Wang Fengling. Principles and Applications of Database Systems [M]. Xidian University Press, 2022. 2.Wang Shan,Du Xiaoyong,Chen Hong. Introduction to Database Systems (6th Edition) [M]. Higher Education Press, 2023. 3.Li Guoliang, Feng Jianhua, Chai Chengliang. Database System Theory: From Fundamental Principles to System Construction[M]. Higher Education Press, 2024. 4.Wang Xuhui. Database Technology and Its Application in the Big Data Environment [J]. Information Recording Materials, 2023, 24(03). 5.Wu Dan. The Application Value and Methods of Distributed Database Technology [J]. Digital Communication World, 2023(04).
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Digital Logic

Module name	Digital Logic		
Code	3080900202		
Semester	Spring, 4th Semester		
Contact person	Qian Yuying, Jiang Shanshan, Ren Xiangjun		
Language	Chinese		
Relation to curriculum	Compulsory, Professional Foundational Course		
Type of teaching, contact hours	Lecture Method,Group Discussion, MethodCase Analysis, MethodProject-Based Learning Method Teaching hours: 2 hours/week, 16 weeks, total 32 hours.		
Work load	Total workload (estimated): 56 hours Teaching hours: 32 hours Self-study hours: 24 hours (including homework, exam preparation, etc.)		
Credit points	2		
Recommended prerequisites	Advanced Mathematics AⅠ, Advanced Mathematics AⅡ		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Ability to analyze characteristics and functions of digital logic devices; describe and analyze combinational/sequential logic circuits; analyze typical pulse circuits, semiconductor memories, ADC/DAC structures and principles; select devices and parameters for engineering problems. Problem-oriented scientific thinking.	R2
	CLO2	Ability to perform preliminary design of digital logic circuits; apply fundamental principles to complete designs of combinational/sequential logic circuits according to specifications; comprehensively consider constraints and understand dialectical unity of contradictions.	R3
	CLO3	Through experimental teaching, integrate theory with practice; research and verify knowledge of digital logic circuits, embrace the principle that "practice is the sole criterion for testing truth".	R4

Content	Digital Logic is an important hardware technology fundamental course for Information Engineering majors. It lays a solid hardware foundation for subsequent courses such as Computer Organization and Compiler Principles. Combining theory and practice, the course introduces basic knowledge, theories, and commonly used digital ICs, with emphasis on methods of digital logic circuit analysis and design. Students are expected to understand various digital circuits that compose digital computers/systems, use logic algebra as a tool, master logic analysis and design of combinational circuits, synchronous sequential circuits, and asynchronous sequential circuits. Students will also design/debug functional blocks using modern digital devices, complete logic designs with suitable ICs, and acquire essential skills for analyzing, designing, and developing digital system hardware. Knowledge modules (with cognitive levels): • Module 1: Fundamentals of Digital Logic (weight 2/32, level: memory, comprehension) • Module 2: Logic Gate Circuits (weight 2/32, level: comprehension, application) • Module 3: Boolean Algebra (weight 4/32, level: comprehension, application) • Module 4: Analysis & Design of Combinational Logic Circuits (weight 4/32, level: application, analysis) • Module 5: Common Combinational Modules & Applications (weight 4/32, level: application, analysis, creation) • Module 6: Flip-flops (weight 4/32, level: application, analysis, creation) • Module 7: Sequential Logic Circuit Analysis (weight 4/32, level: application, analysis, creation)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 60% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 40% of the overall grade and includes: (1) class performance 12%; (2) regular assignments 12%; and (3) periodic tests 16%. Class performance is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	Textbook: [1] Yan Shi (Ed.), Fundamentals of Digital Electronic Technology (6th Ed.), Higher Education Press, 2016. [2]Huang Xu et al., Experimental Tutorial on Digital Electronic Technology, Soochow University Press, 2016. [3] Kang Huaguang (Ed.), Electronic Technology Fundamentals (Digital Part) (6th Ed.), Higher Education Press, 2014. [4] Yan Shi, Wang Hong, Instructor's Manual for Digital Electronic Technology (5th Ed.), Higher Education Press, 2006. [5] Thomas L. Floyd (trans. by Lou Shuqin, Sheng Xinzhi, Shen Yan), Digital Fundamentals: A Systems Approach, China Machine Press, 2014. [6] William Kleitz (trans. by Tao Guobin, Zhao Yufeng), Digital Electronics, Science Press, 2008. [7] Wu Chunming, Lei Yuling, Li Lei (Eds.), Digital Circuits and Logic Design (2nd Ed.), Tsinghua University Press, 2019. [8] Zhu Dinghua, Digital Circuits and Logic Design, Tsinghua University Press, 2011.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Principles of Computer Composition

Module name	Principles of Computer Composition		
Code	3080900209		
Semester	Fall, 5th Semester		
Contact person	Dong Ling, Gao Wei, Wang Jing, Liu Qianqian		
Language	Chinese		
Relation to curriculum	Compulsory, Professional Foundational Courses		
Type of teaching, contact hours	Lecture method, demonstration method, group discussion method, case analysis, laboratory experiment method Teaching duration: 4 hours per week, totaling 12 weeks and 48 hours in total.		
Work load	Total workload (estimated): 84 credit hours Teaching hours: 48 hours Self-study hours: 36 hours, including homework and exam preparation time		
Credit points	3		
Recommended prerequisites	Digital Logic, Data Structures and Algorithms		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Course Objective 1: To achieve a precise understanding of the overall architecture of computer systems and the composition and functions of core hardware components (CPU, memory, bus, I/O interfaces); to master the operating principles, key technical parameters, and interaction mechanisms of each component; to clearly distinguish between computer organization and computer architecture concepts; and to accurately describe fundamental concepts such as binary encoding, instruction sets, and addressing modes, thereby establishing a solid theoretical foundation for advanced studies.	**R1**
	CLO2	Course Objective 2: By applying core knowledge of computer architecture principles, analyze the hardware connectivity logic, instruction execution flow, and data transmission processes of simple computer systems; integrate prior knowledge in digital electronics and assembly language to interpret instruction formats, perform assembly coding and machine code conversion for simple instructions, develop preliminary skills in analyzing and solving fundamental computer hardware issues, and independently complete basic hardware logic analysis tasks.	**R2**

	CLO3	Course Objective 3: To comprehensively apply theoretical knowledge of computer architecture principles to analyze the design concepts and implementation methods of typical computer hardware modules (such as arithmetic units, controllers, and memory); to develop preliminary designs for simple hardware logic modules or optimize workflows of existing modules based on specified functional requirements; to acquire fundamental capabilities in hardware system analysis, design, and optimization; and to understand the collaborative relationship between hardware and software.	**R3**
	CLO4	Course Objective 4: To develop a holistic understanding of computer systems, establish the conceptual framework that "hardware forms the foundational support for software operation," and cultivate rigorous logical reasoning skills and engineering practice competence; to stay informed about trends in computer architecture technologies (such as multi-core architectures and advanced memory technologies), possess the ability for self-directed learning and continuous exploration of new hardware technologies, integrate acquired knowledge with practical engineering applications, and lay a solid foundation for subsequent studies in operating systems, embedded development, and related fields, as well as for careers in computer hardware-related work.	**R3**
Content	This course serves as a core foundational comprehensive course designed to enhance students 'understanding of fundamental computer technologies and develop their hardware application skills. It enables students to comprehend the overall architecture and operational mechanisms of computer systems, master the composition principles, operating mechanisms, and design concepts of major hardware components, and become familiar with the structures and interconnection techniques of key hardware modules such as arithmetic units, memory, controllers, and input/output systems. Through case studies and practical applications, students will understand instruction execution processes, data storage methods, and bus interaction logic. The course aims to cultivate students' ability to apply theoretical knowledge in electronics and digital logic to analyze hardware operation mechanisms and solve practical hardware problems, while strengthening their logical analysis capabilities for hardware systems and architectural design thinking. It also enhances the integration of theoretical knowledge with engineering practice, laying a solid foundation for subsequent courses such as Operating Systems, Embedded Development, and Microcomputer Principles. Knowledge Module 1: Overview of Computer Systems and Number Systems and Encoding (Weight: 2/48; Level: Memory – Understanding) Knowledge Module 2: Operators and Fundamentals of Data Operations (Weight: 4/48; Level: Understanding – Application) Knowledge Module 3: Memory Architecture and Design Principles (Weight: 12/48; Level: Understanding – Analysis) Knowledge Module 4: Instruction Set and Central Processing Unit (CPU) (Weight: 16/48; Level: Application – Analysis) Knowledge Module 5: Bus Systems and Input/Output Interfaces (Weight: 14/48; Level: Understanding – Application)		

Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 70% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 30% of the overall grade and includes: (1) class participation 10%; (2) regular assignments 10%; and (3) interim tests 10%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Feng Jianwen, Zhang Fujia, Zhao Jianyong, et al. Principles of Computer Organization and System Architecture [M]. 3rd ed. Beijing: Higher Education Press, 2024. [2] Tang Shuofei. Principles of Computer Organization [M]. 2nd ed. Beijing: Higher Education Press, 2013. [3] Von Neumann J. First draft of a report on the EDVAC [R]. Philadelphia: University of Pennsylvania, 1945. [4] Liu L. Innovation and Practice in Blended Teaching of Computer Organization Principles with AI Empowerment [J]. Journal of Computer Technology and Education, 2025,13(1):45–52. [5] Xu Weidong, Lou Mian, Li Li, et al. Bit-serial SRAM in-memory computation based on original complement encoding [J]. Journal of Beijing Institute of Technology, 2024,44(10):1095–1104.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Computer Networks

Module name	Computer Networks		
Code	3080900206		
Semester	Fall ,5th Semester		
Contact person	Su Nannan, Ren Xiangjun, Su Wanli, Xu Ying		
Language	Chinese		
Relation to curriculum	Compulsory， Professional Foundational Course		
Type of teaching, contact hours	Lecture; group discussion; laboratory experiments. Teaching hours: 2 hours per week, 16 weeks, total 32 hours.		
Work load	Total workload (estimated): 56 hours Teaching hours: 32 hours Self-study hours: 24 hours (including assignments, exam preparation, etc.)		
Credit points	2		
Recommended prerequisites	C Language Programming, Data Structures and Algorithms		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Master the architecture of computer networks, protocol principles, fundamentals of data communication; understand network layering, establish network engineering thinking, and build network security awareness and information literacy.	R1
	CLO2	Master the TCP/IP protocol suite, routing and switching principles; be able to analyze network failures and performance bottlenecks; possess network problem analysis and solving capabilities.	R2
	CLO3	Be able to complete local area network (LAN) setup, device configuration, protocol debugging and simple network application development; achieve hardware-software collaboration; possess network system design and implementation skills.	R3
	CLO4	Communicate and collaborate effectively in network projects and experiments, complete network planning, deployment, and optimization; cultivate engineering practice and innovative application abilities.	R3

Content	This course systematically explains computer network architecture, communication principles, network protocols, LAN and WAN technologies, network interconnection, network security, etc., emphasizing the combination of theory and practice, and cultivating abilities in network planning, configuration, debugging, and application development. Knowledge modules (with weights and cognitive levels): • Module 1: Introduction to Computer Networks (weight 2/32, level: comprehension) • Module 2: Physical Layer & Fundamentals of Data Communication (weight 4/32, level: comprehension + application) • Module 3: Data Link Layer & Local Area Networks (weight 8/32, level: comprehension + application + analysis) • Module 4: Network Layer & Routing Protocols (weight 8/32, level: comprehension + application + analysis) • Module 5: Transport Layer & Application Layer Protocols (weight 8/32, level: comprehension + application + analysis) • Module 6: Network Security & Network Practice (weight 2/32, level: application + evaluation)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 60% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 40% of the overall grade and includes: (1) class participation 12%; (2) regular assignments 12%; and (3) interim tests 16%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	Textbook: [1] Wu Gongyi, Wu Ying, Computer Networks (5th Edition), Tsinghua University Press, July 2022. Reference books: [1] Xie Xiren, Computer Networks (8th Edition), Publishing House of Electronics Industry, 2019. [2] Chen Hong, Xiao Chenglong, Computer Networks, China Machine Press, 2022. [3] Yuan Jiazheng, Zhang Jingzun, Computer Networks, Xidian University Press, 2016. [4] Zhou Ge, Fundamentals of Computer Network Technology, People's Posts and Telecommunications Press, 2022.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Web Front-End Development Technology

Module name	Web Front-end Development Technology		
Code	3080902227		
Semester	Spring, 2nd Semester		
Contact person	Sun Yingying, Jiang Zhijun		
Language	Chinese		
Relation to curriculum	Compulsory，Professional Core Course		
Type of teaching, contact hours	Lecture Method, Demonstration Method,Group Discussion Method,Project-Based Learning Method Teaching hours: 2 hours per week for 16 weeks, totaling 32 hours.		
Work load	Total workload (estimated): 56 hours Teaching hours: 32 hours Self-study hours: 24 hours, including homework, exam preparation, etc.		
Credit points	2		
Recommended prerequisites	C Language Programming, Introduction to Software Engineering Program		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Through the teaching of this course, students should master the basic knowledge of Web technology, become familiar with the fundamental elements of HTML, and understand the commonly used techniques in website development. They should grasp the essential methods of web front-end development technology, be able to design and develop web pages using technologies such as HTML, CSS, and JavaScript. Students should become familiar with the B/S development model and basically acquire the ability to develop websites. They should be capable of comprehensively applying HTML and JavaScript technologies to develop the front end of websites, thereby enhancing their analytical and problem-solving skills. They should master the techniques of web front-end development technology, laying a solid theoretical and technical foundation for subsequent courses and future practical applications, while also improving their self-learning abilities to ensure they can further expand their knowledge in this field.	Supported PLOs
	CLO1	**Course Objective 1:** Be able to apply the basic principles of mathematics, natural sciences, and software engineering to identify, articulate, and analyze complex engineering problems in the field of software engineering through literature research, in order to draw valid conclusions.	**R2**

	CLO2	**Course Objective 2:** Be able to design solutions for complex engineering problems in the field of software engineering, develop software systems, modules, or algorithm processes that meet specific requirements, and demonstrate innovative thinking in the design process while considering social, health, safety, legal, cultural, and environmental factors.	**R3**
	CLO3	**Course Objective 3:** Be able to develop, select, and use appropriate technologies, resources, modern engineering R&D tools, and information technology tools for complex engineering problems in the field of software engineering, including the analysis, design, testing, implementation, application, and maintenance of such problems, while also understanding their limitations.	**R5**
	CLO4	**Course Objective 4:** Be able to effectively communicate and exchange ideas with industry peers and the public on complex engineering issues in the field of software engineering, including writing reports and design documents, delivering presentations, clearly expressing or responding to instructions; and possess a certain international perspective, enabling communication and interaction in cross-cultural contexts.	**R9**
Content	The course "Web Front-end Development Technology" primarily covers the fundamental principles of website operation, as well as the basic knowledge and core techniques of web front-end development. Through this course, students are expected to master essential knowledge of web technologies, become familiar with basic HTML elements, and understand commonly used website development techniques. They will acquire the fundamental methods of web front-end development technology, enabling them to design and develop web pages using technologies such as HTML, CSS, and JavaScript. Students will gain familiarity with the B/S (Browser/Server) development model and develop foundational abilities in website development. They will learn to comprehensively apply HTML and JavaScript technologies for front-end website development, thereby enhancing their analytical and problem-solving skills. Moreover, they will grasp key techniques in web front-end development, laying a solid theoretical and technical foundation for subsequent courses and future practical applications. The course also aims to improve students' self-learning capabilities, enabling them to further expand their knowledge in this field. Knowledge Module 1: Basics of Web Pages and Websites (Weight: 2/32, Level: Remember) Knowledge Module 2: Fundamentals of HTML (Weight: 4/32, Level: Remember + Understand + Apply) Knowledge Module 3: Fundamentals of CSS (Weight: 4/32, Level: Remember + Understand + Apply) Knowledge Module 4: CSS Layout Design (Weight: 4/32, Level: Apply - Analyze) Knowledge Module 5: Fundamentals of JavaScript (Weight: 4/32, Level: Remember + Understand + Apply) Knowledge Module 6: Objects in JavaScript (Weight: 4/32, Level: Understand + Apply + Analyze) Knowledge Module 7: Event Handling (Weight: 4/32, Level: Apply + Analyze) Knowledge Module 8: Comprehensive Case Study (Weight: 6/32, Level: Analyze - Create)		

Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 50% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 15%; (2) regular assignments 15%; and (3) computer experiment 20%. Classroom performance is comprehensively evaluated based on student topic discussions in class, in-class exercises, and learning of digital courses.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] "Web Front-end Development Technology Tutorial", Ou Jingmin, Kan Dianyu, Northwestern Polytechnical University Press, 2020. [2] "HTML5+CSS3+JavaScript: From Entry to Project Practice", Jumuke Education R&D Center, Tsinghua University Press, 2022. [3] "Web Website Design and Development Tutorial (HTML5, JSP Edition, 2nd Edition)", Wen Haoyu, Li Hui, Xidian University Press, 2018. [4] "Web Front-end Development Technology Tutorial (HTML+CSS+JavaScript) 2nd Edition", Liu Ruixin, Zhang Bingyi, Mechanical Industry Press, 2022. [5] Wang Na. Research on the Teaching Reform of the "Web Front-end Development" Course Guided by the Integration of "Post, Course, Competition, and Certification" [J]. Journal of Yuzhang Normal University, 2025(3). [6] Dong X H, Zhang X X, Zhang K X. Exploration and Practice of Hybrid Teaching Mode of Web Front-end Development Technology Based on OBE [C]//Proceedings of the 2023 2nd International Conference on Science Education and Art Appreciation (SEAA 2023). SHS Web of Conferences, 2023, 174: 01010. DOI:10.1051/shsconf/202317401010
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Software Engineering

Module name	Software Engineering		
Code	3080902210		
Semester	Spring, 4th Semester		
Contact person	Cao Xiuping, Wei Liangping		
Language	Chinese		
Relation to curriculum	Compulsory，Professional Core Course		
Type of teaching, contact hours	Lecture method, demonstration method, group discussion method, case analysis Teaching duration: 4 hours per week for 12 weeks, totaling 48 hours		
Work load	Total workload (estimated): 84 credit hours Teaching duration: 48 hours Self-study hours: 36 hours, including homework and exam preparation time.		
Credit points	3		
Recommended prerequisites	C Language Programming、Data Structures and Algorithms、Principles of Operating Systems		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1：** Through the study of Software Engineering, students will master the fundamental concepts of software engineering, the software life cycle, core development models, requirements engineering, software design, as well as the basic theories and methods of testing and maintenance. By integrating the current status of China's software industry development and its independent R&D needs, this course solidifies the core engineering knowledge essential for the software engineering discipline.	**R1**
	CLO2	**Course Objective 2：** Through this course, students will master the fundamental approaches and methods for software problem analysis, learn to identify practical issues such as requirement deviations, design flaws, and quality risks in software projects, and apply software engineering principles to decompose, analyze, and pinpoint complex software problems, thereby enhancing their system-level problem analysis and logical reasoning skills.	**R2**

	CLO3	**Course Objective 3：** Through practical training in software design and development, this program systematically integrates software engineering theory with program development. Students will be capable of completing preliminary design, detailed design, and functional module development based on software requirements, acquiring fundamental skills in software system design and development, and laying a solid foundation for subsequent research and development of complex software systems.	**R3**
	CLO4	**Course Objective 4：** Through learning and practicing software project management knowledge, one masters core project management methodologies such as project planning, task decomposition, schedule control, quality assurance, and risk management. This enables participation in and execution of the full-process management of small-scale software projects, equipping individuals with fundamental software project management literacy and practical skills.	**R10**
Content	"Software Engineering" is a core course for the software engineering major, designed to systematically introduce the theoretical framework, development methodologies, design specifications, and project management techniques of software engineering. It covers essential topics such as an overview of software engineering, software life cycle models, requirements engineering, software preliminary and detailed design, software coding standards, software testing techniques, software maintenance, and software project management. Through this course, students will master the fundamental principles and practical methods of software engineering, develop the ability to analyze, design, and develop software systems as well as perform basic project management tasks, cultivate a standardized and engineering-oriented mindset for software development, and establish a solid theoretical and practical foundation for careers in software research and development or project management. Knowledge Module 1: Introduction to Software Engineering (Weight: 4/48, Level: Remembering + Understanding) Knowledge Module 2: Software Life Cycle and Development Models (Weight: 6/48, Level: Remembering + Understanding + Application) Knowledge Module 3: Requirements Engineering (Weight: 10/48, Level: Understanding - Analysis) Knowledge Module 4: Software Design (Weight: 12/48, Level: Understanding - Creation) Knowledge Module 5: Software Coding and Specifications (Weight: 6/48, Level: Understanding + Application) Knowledge Module 6: Software Testing (Weight: 6/48, Level: Understanding + Application) Knowledge Module 7: Software Project Management (Weight: 4/48, Level: Remembering + Understanding + Application)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 60% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 40% of the overall grade and includes: (1) class participation 12%; (2) regular assignments 12%; and (3) computer experiment 16%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.		
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.		

Reading list	[1] Zhang Haifan, Mou Yongmin. Introduction to Software Engineering (8th ed.)[M]. Beijing: Tsinghua University Press, 2022. [2] Roger S. Pressman. Software Engineering: A Practitioner's Approach (9th ed.)[M]. Beijing: China Machine Press, 2021. [3] Zhou Su, Wang Wen. Practical Tutorial of Software Engineering[M]. Beijing: Science Press, 2023. [4] Li Li, Wang Yanqiu. Research on Optimization of Teaching System for Software Engineering Course[J]. Computer Education, 2024. [5] Chen Yang, Zhang Min. Exploration on Teaching Reform of Software Engineering for Engineering Accreditation[J]. Computer Engineering & Science, 2023, 45(S1): 112-116. [6] Liu Qiang, Zhang Hua. Practice and Research on Project-based Teaching Mode of Software Engineering[J]. China Information Technology Education, 2022.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Object-Oriented Programming

Module name	Object-Oriented Programming		
Code	3080902228		
Semester	Spring, 4th Semester		
Contact person	Sun Fang, You Haidong		
Language	Chinese		
Relation to curriculum	Compulsory, Professional Core Course		
Type of teaching, contact hours	Lecture Method, Demonstration Method, Group Discussion Method and Laboratory Experiment Teaching Hours: 4 hours per week for 16 weeks, totaling 64 hours		
Work load	Total estimated workload: 112 hours Contact hours: 64 hours Independent Study: 48 hours (assignments, exam preparation, etc.)		
Credit points	4		
Recommended prerequisites	C Language Programming		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	CLO description	Supported PLOs
	CLO1	**CLO1:** Explain the core principles of object-oriented programming; apply Java syntax to implement classes, objects, encapsulation, inheritance, and polymorphism; demonstrate foundational programming skills and software engineering literacy.	**R1**
	CLO2	**CLO2:** Apply object-oriented analysis to decompose real-world problems; define appropriate classes, objects, attributes, and methods; model business requirements as software solutions.	**R2**
	CLO3	**CLO3:** Be able to design reasonable class structures according to requirements, implement encapsulation, inheritance, polymorphism, complete the design, coding and debugging of small object-oriented programs, and have modular programming development capabilities.	**R3**
	CLO4	**CLO4:** Configure and use JDK for Java program compilation and execution; operate IDEs (Eclipse, IntelliJ IDEA) for integrated development tasks; implement version control workflows and debugging strategies; integrate modern development tools into software engineering practice.	**R5**

Content	This course is a fundamental core course for computer-related majors. Using Java as the programming language, it systematically introduces object-oriented programming concepts and development skills. Topics include classes and objects, encapsulation, inheritance, polymorphism, interfaces, abstract classes, packages, exception handling, common class libraries, the collection framework, I/O fundamentals, multithreading, and a comprehensive capstone project. Through theoretical instruction and extensive programming practice, students will master object-oriented analysis, design, and implementation methodologies; develop modular thinking, coding standards awareness, and engineering practice capabilities; and establish a solid foundation for advanced studies in framework technologies, Web development, and mobile application development. Knowledge Module 1: Java Basics and Object-Oriented Overview (Weight 8/64, Level: Understanding) Knowledge Module 2: Classes, Objects, Encapsulation and Constructors (Weight 12/64, Level: Application) Knowledge Module 3: Inheritance, Method Overriding and super Keyword (Weight 12/64, Level: Application) Knowledge Module 4: Polymorphism, Abstract Classes and Interfaces (Weight 12/64, Level: Application + Analysis) Knowledge Module 5: Exception Handling, Common Classes and Collection Framework (Weight 12/64, Level: Application) Knowledge Module 6: I/O, Multithreading and Comprehensive Project Practice (Weight 8/64, Level: Creation)
Forms of examination	Final exam (closed-book): 50% of total scoreDaily assessment: class performance, assignments, programming experiments (50% of total score) The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 50% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 15%; (2) regular assignments 15%; and (3)computer experiment 20%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Joshua Bloch. Effective Java[M]. 3rd Edition. Boston: Addison-Wesley, 2018. [2] Cay S. Horstmann. Core Java Volume I – Fundamentals[M]. 12th Edition. Beijing: China Machine Press, 2022. [3] Sun Qinqin. Java Object-Oriented Programming[M]. 2nd Edition. Beijing: Publishing House of Electronics Industry, 2019. [4] Kathy Sierra. Head First Java[M]. 3rd Edition. Beijing: China Electric Power Press, 2023. [5] Qin B. Comparative Teaching of C++ and Java in Object-Oriented Programming Courses[J]. Computer Education, 2024(04): 144-148+153. [6] Tao J. Experimental Research on Encapsulation of OOP in Java[J]. Information & Computer, 2024(02): 115-118.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Principles of Compilation

Module name	Principles of Compilation		
Code	3080903243		
Semester	Fall, 5th Semester		
Contact person	Chen Linong,Wang Fengqin,Zhao Xia		
Language	Chinese		
Relation to curriculum	Compulsory，Professional Core Course		
Type of teaching, contact hours	This course adopts an integrated implementation of three categories of teaching methods: teacher-centered methods, interactive methods, and individualized methods. Teacher-centered methods include: lectures (concept explanation, algorithm derivation, principle elucidation), case teaching (analysis of typical compiler examples, grammar design examples), questioning and in-class exercises; Interactive methods include: group discussions (centered on topics such as grammar analysis and optimization strategies), teaching seminars (analysis of key and difficult points and sharing of ideas); Individualized methods include: tiered assignments (combining basic and advanced exercises), literature reading and reporting (independently selecting topics to read papers in the compilation field and writing reading notes), personalized programming practice (independent implementation of compiler components such as lexical analyzers or syntax analyzers based on personal interest); Teaching Hours: 2 hours per week, 16 weeks in total, 32 hours in total.		
Work load	Total Workload (estimated): 56 hours. Teaching Hours: 32 hours. Self-Study Hours: 24 hours, including: after-class assignments, exam preparation, etc.		
Credit points	2		
Recommended prerequisites	Discrete Mathematics, Data Structures and Algorithms, Principles of Operating Systems		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes	Description	Supported PLOs
	CLO1	Master the fundamental concepts, formal description methods, and core algorithm principles of compiler theory, including finite automata theory for lexical analysis, LL/LR parsing methods for syntax analysis, syntax-directed translation techniques, etc., and be able to apply these theories and methods to formal modeling and problem-solving for each stage of compilation.	**R1**
	CLO2	Be able to apply core algorithms and design methods of compiler theory to identify and determine key issues in the stages of lexical, syntax, and semantic analysis within the compilation process, analyze the complexity of problems, and design corresponding compilation processing schemes for specific grammars.	**R2**
	CLO3	Be able to independently design and implement front-end components of a compiler (such as lexical analyzer, syntax analyzer, semantic analyzer) for a specific high-level language grammar, complete the full development process from	**R3**

		requirements analysis, scheme design to coding implementation, and be able to optimize the design scheme.	
	CLO4	Be able to read and understand relevant literature in the field of compilation, understand the development history and current research hotspots of compilation technology, and master the basic principles and development trends of cutting-edge technologies such as trusted compilation, deep learning compilation, and parallel compilation.	**R4**
	CLO5	Be able to use modern compilation tools such as lexical analyzer generators (Lex/Flex), syntax analyzer generators (Yacc/Bison), and the LLVM framework for compiler development and testing, understand the technical principles and limitations of these tools, and be able to communicate effectively and divide work reasonably within team collaboration.	**R5**
Content	Compiler Principles is a core specialized course for the Software Engineering major. This course not only covers the basic structure and workflow of a compilation system but also introduces in detail the design principles of each component of a compiler and commonly used compilation techniques and methods. Through learning, students will master the entire process of converting source code to executable code, including key stages such as lexical analysis, syntax analysis, semantic analysis, intermediate code generation, code optimization, and target code generation. The course content not only reflects the problem-solving approaches and formal problem methods in the field of computer science but also provides important inspiration and guidance for the design and development of application software and system software. Additionally, the principles and techniques covered in this course are widely applied in fields such as software engineering, reverse engineering, software re-engineering, and language conversion. Through this course, students will build a solid theoretical and practical foundation for engaging in the development of application software and system software, enhancing their professional competence and comprehensive ability. This course includes the following knowledge modules: Knowledge Module 1: Overview of Compilation (Weight 2/32, Level: Remembering + Understanding) Knowledge Module 2: Language and Grammar Fundamentals (Weight 2/32, Level: Remembering + Understanding) Knowledge Module 3: Design of Lexical Analysis Programs (Weight 2/32, Level: Remembering - Analyzing) Knowledge Module 4: Syntax Analysis: Recursive Descent Parsing (Weight 2/32, Level: Remembering - Evaluating) Knowledge Module 5: Symbol Tables (Weight 2/32, Level: Remembering + Understanding) Knowledge Module 6: Runtime Storage Organization and Management (Weight 2/32, Level: Remembering + Understanding + Applying) Knowledge Module 7: Syntax-Directed Translation Techniques (Weight 2/32, Level: Remembering + Understanding + Applying) Knowledge Module 8: Semantic Analysis and Intermediate Code Generation (Weight 2/32, Level: Remembering + Understanding + Applying) Knowledge Module 9: Control Flow and Data Flow (Weight 2/32, Level: Remembering + Understanding + Applying) Knowledge Module 10: Code Optimization Techniques (Weight 2/32, Level: Remembering - Evaluating) Knowledge Module 11: Target Code Generation and Optimization (Weight 2/32, Level: Remembering - Evaluating) Knowledge Module 12: Automatic Generation Techniques for Lexical Analyzers (Weight 2/32, Level: Remembering + Understanding + Applying)		

	Knowledge Module 13: Automatic Generation Techniques for Syntax Analyzers (Weight 2/32, Level: Remembering + Understanding + Applying) Knowledge Module 14: Syntax-Directed Translation Techniques II (Weight 2/32, Level: Remembering + Understanding + Applying) Knowledge Module 15: LLVM Compiler Architecture (Weight 2/32, Level: Remembering + Understanding) Knowledge Module 16: Advanced Compilation Techniques (Weight 2/32, Level: Remembering + Understanding)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 50% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 15%; (2) regular assignments 15%; and (3) interim tests 20%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	Recommended Textbooks: [1] Compiler Principles and Techniques, Zhang Li et al., Higher Education Press, 2024, 1st Edition. [2] Compiler Methods, Techniques and Practice, Xu Chang et al., China Machine Press, 2024. [3] Compilers: Principles, Techniques, and Tools (2nd Edition), Alfred V. Aho et al., translated by Zhao Jianhua et al., China Machine Press, 2009. [4] Modern Compiler Implementation in C, Andrew W. Appel et al., Posts and Telecom Press, 2018. [5] Compiler Principles (3rd Edition), Wang Shengyuan, Dong Yuan, Zhang Suqin et al., Tsinghua University Press, 2015. [6] Principles of Programming Language Compilation (3rd Edition), Chen Huowang, Liu Chunlin et al., National Defense Industry Press, 2000. Recommended Reading Papers: [7] Xuejun Yang, Yang Chen, Eric Eide, John Regehr. Finding and Understanding Bugs in C Compilers[C]. Proceedings of the 32nd ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2011), June 2011. [8] Vu Le, Mehrdad Afshari, Zhendong Su. Compiler Validation via Equivalence Modulo Inputs[C]. Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2014), June 2014.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Software Architecture

Module name	Software Architecture		
Code	3080902222		
Semester	Fall, 5th Semester		
Contact person	Wei Jinping, Sun Fang, Zong Chengjie		
Language	Chinese		
Relation to curriculum	Compulsory，Professional Core Course		
Type of teaching, contact hours	Lecture Method, Demonstration Method, Group Discussion Method, Case Analysis Method, Laboratory Experiment Method Teaching Hours: 4 hours per week for 8 weeks, totaling 32 hours		
Work load	Total Workload (estimated): 56 hours. Teaching Hours: 32 hours. Self-study Hours: 24 hours, including: after-class assignments, examination preparation time, etc.		
Credit points	2		
Recommended prerequisites	Principles of Database Systems, Data Structures and Algorithms		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1:** Guide students to apply basic principles of software engineering, mathematics and natural sciences to identify complex engineering problems in software architecture design, master methods to describe key problem links using flowcharts, class diagrams and other tools, analyze problem roots in different scenarios, form scientific and reasonable problem analysis conclusions, and improve the ability to identify and analyze complex problems.	**R2**
	CLO2	**Course Objective 2:** Through studying this course, enable students to design architecture solutions that meet requirements for complex engineering problems in the field of software architecture, complete architecture selection, module division, component design and database adaptation, integrate innovative concepts into design, fully consider relevant constraints such as security and law, and have the ability to transform design schemes into implementable software architecture models.	**R3**
	CLO3	**Course Objective 3:** Enable students to proficiently master modern tools related to software architecture design, including architecture design tools, code development platforms, testing tools and database management systems, be able to select appropriate tools according to the needs of complex engineering problems, complete architecture analysis, design and verification, understand application scenarios and limitations of various tools, and improve tool application and optimization capabilities.	**R5**

	CLO4	**Course Objective 4:** Help students master project management methods related to software architecture, understand economic and management factors in architecture design, be familiar with core processes and technologies of architecture management in the whole lifecycle of software projects, and be able to apply project management principles in multidisciplinary collaboration scenarios, preliminarily possessing management and coordination capabilities for complex software architecture projects.	**R10**
Content	This course is an important research field in computer science, aiming to explore how to quickly and cost-effectively build reasonable and reliable software systems, especially in aerospace, national defense and other fields that often need to handle large and complex software systems. As an introductory course to software architecture, this course will introduce relevant basic theories, engineering practices and latest research trends in detail. Through studying this course, students will master the basic concepts of software architecture, understand its necessity, and learn formal language description methods for software architecture. In addition, the course will also discuss the practical application of different software architecture styles and their important role in the software system development process. This will lay a solid foundation for students' future study and work. Knowledge Module 1: Software Architecture Concepts (Weight 2/32, Level: Remember, Understand) Knowledge Module 2: Software Architecture Styles and Patterns (Weight 8/32, Level: Remember, Apply, Analyze) Knowledge Module 3: Software Architecture Description (Weight 8/32, Level: Remember - Evaluate) Knowledge Module 4: Software Architecture Development Environment (Weight 8/32, Level: Apply - Evaluate) Knowledge Module 5: Software Architecture Evaluation (Weight 4/32, Level: Analyze, Evaluate) Knowledge Module 6: Prospects of Software Architecture (Weight 2/32, Level: Analyze, Create)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 60% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 40% of the overall grade and includes: (1) class participation 12%; (2) regular assignments 12%; and (3) experiment grade 16%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.		
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.		
Reading list	[1] Software Architecture, edited by Qin Zheng et al., Tsinghua University Press, 2021, 5th Edition. [2] Software Architecture Design: Patterns, Methods and Practice, edited by Zhou Zhiming, Publishing House of Electronics Industry, 2022. [3] Software Architecture in Practice, edited by Software Engineering Institute of Carnegie Mellon University, Posts & Telecom Press, 2020, 4th Edition. [4] Tutorial for System Architecture Designer, edited by Ye Hong, Tsinghua University Press, 2022, 2nd Edition. [5] State of Practice: LLMs in Software Engineering and Software Architecture, Jahic J., Sami A, 2024 IEEE 21st International Conference on Software Architecture Companion (ICSA-C), pp. 311-318, 2024 年.		
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.		

System Analysis and Design

Module name	System Analysis and Design		
Code	3080902226		
Semester	Spring, 6th Semester		
Contact person	Diao Tianru, Yang Weiqing		
Language	Chinese		
Relation to curriculum	Compulsory,Professional Core Course		
Type of teaching, contact hours	Lecture Method, Group Discussion Method,Case Analysis Method Teaching Hours:4 hours per week for 8 weeks,totaling 32 hours		
Work load	Total Workload (Estimated):56 hours Teaching Hours:32 hours Self-study Hours: 24 hours, including: After-class assignments, Exam preparation time, etc.		
Credit points	2		
Recommended prerequisites	Object-Oriented Programming,Data Structures and Algorithms, Computer Networks, Software Engineering, Software Architecture, Principles of Database Systems		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1:** Master the basic theories and core concepts of UML, be familiar with the use of UML modeling tools, and understand the application of UML in the software development life cycle. This helps students flexibly use UML for modeling in practical projects and improve the efficiency and quality of engineering implementation.	**R1**
	CLO2	**Course Objective 2:** Through learning UML diagrams and models, be able to use UML to conduct structured and systematic analysis of complex problems. Students will learn to identify the key points and difficulties of problems, master the method of problem decomposition using UML, thereby cultivating rigorous logical thinking and critical thinking.	**R2**
	CLO3	**Course Objective 3:** Be able to use UML for modeling static structures and dynamic behaviors, and design software systems that meet requirements and are easy to implement. At the same time, learn to optimize design schemes using UML to improve system performance, stability and maintainability.	**R3**
	CLO4	**Course Objective 4:** Improve students' communication efficiency and effectiveness through learning UML diagram expression skills. Students will learn to draw UML diagrams clearly and accurately, and intuitively express the ideas of system design and development through diagrams. This will help students better cooperate with others in teams and improve the efficiency and quality of team collaboration.	**R4**

Content	This course mainly introduces the Unified Modeling Language (UML) and object-oriented analysis and design knowledge. It enables students to master the use of UML tools for object-oriented analysis and design, how to model in analysis and design, and at the same time enables students to master UML analysis and design in the Rose environment, learn to use UML modeling tools, cultivate students' ability to analyze and solve problems, and lay a good foundation for systematic and scientific software development. Knowledge Module 1: UML Overview and Basics (Weight: 2/32, Level: Memory + Understanding) Knowledge Module 2: Use of UML Modeling Tools (Weight: 2/32, Level: Understanding + Application) Knowledge Module 3: Use Case Diagrams (Weight: 4/32, Level: Application + Analysis + Evaluation) Knowledge Module 4: Class Diagrams and Object Diagrams (Weight: 4/32, Level: Application + Analysis + Evaluation) Knowledge Module 5: Sequence Diagrams and Collaboration Diagrams (Weight: 4/32, Level: Application + Analysis + Evaluation) Knowledge Module 6: State Diagrams and Activity Diagrams (Weight: 6/32, Level: Application + Analysis + Evaluation) Knowledge Module 7: Component Diagrams and Deployment Diagrams (Weight: 4/32, Level: Application + Analysis + Evaluation) Knowledge Module 8: Application of UML in the Software Development Life Cycle (Weight: 6/32, Level: Application + Analysis)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 50% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 15%; (2) regular assignments 15%; and (3) interim tests 20%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] 《UML Object-Oriented Analysis, Modeling and Design》, Lü Yunxiang, Zhao Tianyu, Tsinghua University Press, 2021. [2]《UML Software Modeling Task-Driven Tutorial》, Chen Chenghuan, People's Posts and Telecommunications Press, 2022. [3] 《UML Basics, Modeling and Application》, Cao Hanhua, Yi Yang, Guan Chunxi, et al., China Machine Press, 2022. [4] 《UML Modeling Example Tutorial (3rd Edition)》, Feng Xiangke, Liu Zhicheng, Electronic Industry Press, 2024. [5] A UML-Based Modeling Analysis and Design of Common Business for College Students [J]. Sun Xingfu, Yan Mengru, Cao Jiachi, et al., China Science and Technology Information, 2024, (15): 65-68.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Software Testing

Module name	Software Testing		
Code	3080902224		
Semester	Spring ,6th Semester		
Contact person	Zong Chengjie、Zhou Ying、Cui Zhenzhen		
Language	Chinese		
Relation to curriculum	Compulsory, Professional Core Course		
Type of teaching, contact hours	Lecture Method、Demonstration Method、Group Discussion Method、Case Analysis Method Teaching Hours: 4 hours per week for 8 weeks, totaling 32 hours.		
Work load	Total Workload (Estimated): 56 credit hours Teaching Hours: 32 hours Self-study Hours: 24 hours, including after-class assignments, exam preparation and other related learning activities.		
Credit points	2		
Recommended prerequisites	《Data Structures and Algorithms》《Software Engineering》		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Course Objective 1: Capable of designing targeted test cases for functional testing, performance testing and other non-functional testing based on software projects.	**R3**
	CLO2	Course Objective 2: Master black-box testing, white-box testing and their corresponding test case design.	**R5**
	CLO3	Course Objective 3: Equip students with practical competencies in software testing, enable them to adapt to the working environment and industry standards of software enterprises, and keep pace with advanced international software development concepts and testing technologies.	**R6**
	CLO4	Course Objective 4: Possess sound physical and mental quality, understand relevant laws and regulations, recognize the professional ethics and norms of software engineering featuring honesty, impartiality and integrity, and consciously abide by them in software engineering practices.	**R8**

Content	his course is a core and essential course for the Software Engineering major. It mainly introduces the basic theories, technologies and methods of software testing. Through the study of this course, students will master the fundamental concepts of software testing, learn common testing methods and the use of software testing tools. Meanwhile, students will develop sound logical thinking and analytical capabilities, as well as the abilities to design and compile test cases, together with strong teamwork and communication skills. Knowledge Module 1: Fundamentals of Software Testing (Weight: 2/32, Level: Comprehension) Knowledge Module 2: Black-box Testing (Weight: 10/32, Level: Comprehension + Application + Analysis) Knowledge Module 3: White-box Testing (Weight: 10/32, Level: Comprehension + Application + Analysis) Knowledge Module 4: Web System Testing Cases (Weight: 10/32, Level: Application + Analysis + Creatio
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 50% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 15%; (2) regular assignments 15%; and (3) laboratory work 20%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Wei Nadi, Li Wenbin. Training on Software Testing Technology and Test Case Design[J]. Tsinghua University Press, 2022. [2] Song Xiaojing, Liu Shiyu, Su Mingyue, et al. Research and Practice on FPGA Black-box Testing Method Based on Code Review[J]. Application of Electronic Technology, 2026. [3] Ru Bingsheng, Chen Lei, Zhu Shaomin. Authoritative Guide to Modern Software Testing Technology[M]. Publishing House of Electronics Industry, 2025: 856. [4] Zhu Yanjun, Fu Yuzhen, Fang Xiaodong. Project-based Course of Software Testing[M]. China Railway Publishing House, 2024: 197. [5] Wang Honggang, Xie Bingxian, Wang Zhengfeng. Software Engineering: Theory and Practice[M]. Northwest University Press, 2024: 262.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Software Project Management

Module name	Software Project Management		
Code	3080902223		
Semester	Fall, 7th Semester		
Contact person	Liu Qianqian, Guo Ning		
Language	Chinese		
Relation to curriculum	Compulsory, Professional Core Course		
Type of teaching, contact hours	Lecture Method, Demonstration Method, Case Analysis Method, Situational Experiment Method, Project-based Learning Method Teaching hours: 4 hours per week, 8 weeks in total, 32 hours		
Work load	Total estimated workload: 56 hours Contact hours: 32 hours Self-study hours: 24 hours, including homework, exam preparation, etc.		
Credit points	2		
Recommended prerequisites	Software Engineering, Software Architecture, Software Testing		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Knowledge Objectives 1. Master the concepts of software project management; 2. Be familiar with the lifecycle and management processes of software projects; 3. Understand the responsibilities and competencies of project managers, and master characteristics of different project types; 4. Know the characteristics of human resource requirements in software projects, and master the process and features of project team development; 5. Be familiar with effective communication methods and understand conflict management; 6. Understand the purpose of feasibility studies in software projects, and master the process and content of feasibility studies.	**R3**
	CLO2	Ability Objectives 1. Ability of self-learning and knowledge acquisition; 2. Ability to analyze given software project cases, identify key issues, and propose basic solutions; 3. Ability to manage and control each phase of software development lifecycle, reduce rework, lower development costs, and shorten development cycles.	**R5**

	CLO3	Quality Objectives 1. Cultivate students'awareness of integrating theory with practice and applying knowledge to social practice; 2. Foster students' rigorous and practical work attitude.	**R9**
	CLO4	Certificate Objectives 1. Help students obtain Computer Rank Examination certificates; 2. Lay a solid foundation for future PMP certification in software project management.	**R10**
Content	This course is a compulsory core professional course for undergraduate students majoring in Software Engineering. It covers: concepts of software project management, requirements management, lifecycle, feasibility study, human resource management, communication management, conflict management, contract management, time management, cost management, risk management, quality management, configuration management, and closure management. It is an interdisciplinary course of technology and management, guiding the organization and management of software development projects, aiming to cultivate software project management talents. Students will master basic theories, methods, processes and tools of software project management; develop basic abilities to manage software development projects; apply theories into practice; and improve problem-analysis and problem-solving abilities.		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is a closed-book exam with a duration of 100 minutes, accounting for 50% of the overall grade. It comprehensively assesses students' ability to learn, analyze, and solve complex problems. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 15%; (2) regular assignments 15%; and (3) interim tests 35%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.		
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.		
Reading list	[1] Zhu Shaomin, Cheng Lijun, Yan Ruiyu, Huo Yujie. Software Project Management (3rd Edition). People's Posts and Telecommunications Press, 2025. [2] Li Wei, Liu Hui, Tai Weipeng, Li Jie. Software Project Management and Cases. Tsinghua University Press, 2023. [3] Tian Geng. Software Project Management Based on PMBOK. Tsinghua University Press, 2023. [4] Liang Shilian. Software Project Management. Tsinghua University Press, 2017. [5] Han Wanjiang, Jiang Lixin. Software Project Management Case Tutorial. China Machine Press, 2019. [6] Wu Yanbin, Shi Jinyu, Xie Xiong, et al. Teaching Design and Implementation of "Software Project Management" Course Based on PMO. Journal of Maritime Education Research, 2024, 41(02):26-34+44.		
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.		

Java Web Application Development

Module name	Java Web Application Development		
Code	3080902229		
Semester	Fall ,5th Semester		
Contact person	Chai Yan, Cui Zhenzhen, Wang Jing		
Language	Chinese		
Relation to curriculum	Elective, Professional Elective Course		
Type of teaching, contact hours	Lecture Method, Demonstration Method, Group Discussion Method, Case Analysis Method,Laboratory Experiment Method Teaching Hours: 4 hours per week, 8 weeks in total, 32 hours in total		
Work load （Including teaching hours and self-study hours）	Total Workload (Estimated): 56 hours Teaching Hours: 32 hours Self-study duration: 24 hours, including: pre-class preparation, post-class review and consolidation, assignment completion, error identification and correction, supplementary resource exploration for enhancement, summary and gap identification/addressing, and exam preparation time.		
Credit points	2		
Recommended prerequisites	Object-Oriented Programming, Data Structures and Algorithms, Principles of Operating Systems, Software Engineering, Web Front-end Development Technology, Principles of Database Systems		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1**: Be able to proficiently master the basic principles of mathematics (such as logical operations, data structures, basic algorithms), natural sciences (such as the basics of computer architecture) and software engineering (such as modular design, layered architecture, object-oriented thinking, etc.), and understand the application scenarios of these principles in the field of Java Web application development, so as to provide theoretical support for identifying and analyzing complex engineering problems.	**R2**
	CLO2	**Course Objective 2**: Be able to proficiently master the core methods of Java Web application design and development (such as layered architecture design, modular development, object-oriented design, etc.), understand the internal connection between software requirement analysis and solution design, and master the design specifications and development processes of Java Web systems, modules and algorithm processes, so as to provide technical and methodological support for designing solutions to complex engineering problems.	**R3**

	CLO3	**Course Objective 3:** Be able to design complete solutions based on the results of requirement analysis in combination with complex engineering problems in Java Web application development (such as multi-module system integration, data interaction security, high-concurrency access, cross-scenario adaptation, etc.); Be able to clarify the design ideas, core architecture, module division and algorithm processes of the solution, ensuring that the solution can effectively solve complex engineering problems and meet specific business needs.	**R3**
	CLO4	**Course Objective 4:** Be able to proficiently use basic information technology tools and R&D tools in the field of Java Web development, including but not limited to integrated development environments (such as IntelliJ IDEA, Eclipse), version control tools (such as Git), and database management tools (such as MySQL Workbench, Navicat), master the basic operation processes of the tools, and be able to use the tools to complete the writing, debugging, version management and data interaction of simple Java Web programs, laying a foundation for solving complex engineering problems.	**R5**
Content	This course is a professional elective course for undergraduate students majoring in Software Engineering. The main content of the course includes technologies such as JSP, Servlet, Filter, Listener, EJB, CORBA, JSF, as well as currently popular lightweight framework technologies such as Struts, Hibernate, and Spring. Through systematic theoretical study, course experiments and project development practice, it helps students fully understand the entire technical framework of J2EE, grasp the essence of J2EE programming ideas, comprehensively understand various basic technologies of J2EE programming, proficiently master programming skills, and improve engineering development capabilities. The course requires proficient mastery of Servlet/JSP principles and development, basic XML concepts and the use of XML programming interfaces; understanding of EJB programming models and Web Service principles, and understanding of their basic development steps; understanding of application server installation and management and application deployment, full implementation of project development, and assessment based on practical engineering problem-solving capabilities. Knowledge Module 1: Basics of Dynamic Web Page Development (Weight: 4/32, Level: Memory - Evaluation) Knowledge Module 2: JSP Technology (Weight: 6/32, Level: Memory - Evaluation) Knowledge Module 3: Servlet Basics (Weight: 6/32, Level: Memory - Evaluation) Knowledge Module 4: Request and Response (Weight: 6/32, Level: Memory - Evaluation) Knowledge Module 5: Session and Session Technology (Weight: 6/32, Level: Memory - Evaluation) Knowledge Module 6: EL Expressions and JSTL (Weight: 4/32, Level: Memory - Evaluation)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is conducted as an assessment component, accounting for 50% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class performance 15%; (2) regular assignments 15%; and (3) computer-based laboratory exercises 20%. Class performance is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.		

Study and examination requirements	Following the examination regulations,students must have selected and registered for the courses.
Reading list	[1] Getting Started with Java Web Easily, Cloud Shang Technology, Tsinghua University Press, 2022, 1st Edition. [2]Lightweight Java EE Web Framework Technology - Spring MVC + Spring + MyBatis + Spring Boot, Li Donghai, Tsinghua University Press, 2022, 1st Edition. [3]Java Web and Its Framework Technology, Edited by Chen Zhenxing, Electronic Industry Press, 2024. [4] Ren Rongjun. Research on the Application of Java Language in Computer Software Development [J]. Office Automation, 2026, 31(06): 120-122. [5]Wang Yunhua. Research on the Application of Project-Based Teaching in the Course of Web Back-End Development Technology [J]. Journal of Contemporary Teaching and Research, 2025, 11(12): 23-26.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

C++ Object-Oriented Programming

Module name	C++ Object-Oriented Programming		
Code	3080717264		
Semester	Fall, 5th Semester		
Contact person	Jiang Shanshan, Wan Ming'ao		
Language	Chinese		
Relation to curriculum	Elective, Professional Elective Course		
Type of teaching, contact hours	Teacher-centered methods: Lecture Method, Demonstration Method Interactive methods: Group Discussion Method, Case Analysis Method Practical methods: Laboratory Experiment Method Total contact hours: 4 hours/week, 8 weeks, 32 hours		
Work load	Total estimated workload: 56 hours Teaching hours: 32 hours Self-study hours: 24 hours (including homework, exam preparation, etc.)		
Credit points	2		
Recommended prerequisites	C Language Programming, Data Structures and Algorithms, Discrete Mathematics		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	The core mission of this course is to enable students to: Develop software design capabilities using object-oriented programming languages. Choose appropriate development languages for specific problems in software development.Solve practical problems using object-oriented techniques. Cultivate rigor, practicality, and an awareness of object-oriented software development.Integrate theory with practice and apply knowledge to social practice. Enhance the ability to observe society, identify requirements, and establish positive academic values, thereby improving comprehensive professional qualities.	Supported PLOs
	CLO1	**Course Objective 1：** Apply mathematics, natural science, and software engineering knowledge to analyze complex engineering problems in C++ object-oriented programming, design solutions based on object-oriented features, and complete program development, debugging, and optimization.	**R1**
	CLO2	**Course Objective 2：** Analyze problems using the concepts and creation methods of classes and objects; identify classes, objects, and object initialization; design class methods, call methods, debug programs; and compare syntax differences with C language.	**R2**
	CLO3	**Course Objective 3：** Design base classes and derived classes; define access permissions for class members; design data members and function members; understand name conflicts in inheritance and master resolution methods.	**R3**

	CLO4	**Course Objective 4：** Select and use appropriate modern engineering and information technology tools for C++ object-oriented programming tasks (analysis, design, coding, testing, debugging, maintenance); understand tool applicability and limitations to improve efficiency and quality.	**R5**
Content	This course is a professional elective for software engineering majors. Object-oriented software development integrates abstraction, encapsulation, inheritance, and polymorphism to create modular, highly abstract, reusable, extensible, and maintainable software. It focuses on object-oriented programming principles and mechanisms. Prerequisite: C Language Programming. Follow-up courses: Java Programming, .NET Development, etc. Knowledge Modules & Cognitive Levels： 1. Classes and Objects (weight: 2/32, level: Understand) 2. Special Functions and Members (weight: 4/32, level: Remember + Understand + Apply) 3. Inheritance and Derivation (weight: 9/32, level: Understand + Apply + Analyze) 4. Templates (weight: 6/32, level: Understand + Apply + Analyze) 5. Polymorphism and Virtual Functions (weight: 9/48, level: Understand + Apply + Analyze + Evaluate) 6. C++ Files and I/O Control (weight: 2/48, level: Remember + Understand + Apply)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is conducted as an assessment component, accounting for 50% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class performance 15%; (2) regular assignments 15%; and (3) computer-based laboratory exercises 20%. Class performance is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.		
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.		
Reading list	[1] Zhao X H. C++ Object-Oriented Programming and Project Practice [M]. Beijing: Tsinghua University Press, 2024. [2] Zhou F G, Gao W C. Basic Course of C++ Object-Oriented Programming [M]. Beijing: Tsinghua University Press, 2020. [3] Lin S, Deng N. Construction and practice of competency-oriented object-oriented programming course [J]. Journal of Computer Technology and Education, 2024, 12(4): 74-79. [4] Pei Z, Tian X X, Li B X. Teaching reform and practice of object-oriented programming C++ enabled by knowledge graph [J]. Journal of East China Normal University (Natural Science), 2024(5): 112-122. [5] Li H L, Xue H L, et al. Teaching reform of object-oriented programming course with “one-drive, multi-integration, four-progress and six-stage” [J]. Computer Education, 2024(12): 112-118. [6] Holton I, Van Waelde P. Getting Started with C++20 (6th ed.) [M]. Trans. Zhou B S. Beijing: Tsinghua University Press, 2022.		
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.		

Enterprise-level Java Development

Module name	Enterprise-level Java Development		
Code	3080902416		
Semester	Spring, 6th semester		
Contact person	Wei Jinping, Wang Xin, Qiao Qun		
Language	Chinese		
Relation to curriculum	Elective，Professional Elective Course		
Type of teaching, contact hours	Lecture-based teaching、Demonstration method、Group discussion method、Case analysis method Teaching hours: 2 hours per week for 16 weeks, totaling 32 hours.		
Work load	Total workload (estimated): 56 hours Instructional hours: 32 hours Self-study hours: 24 hours, including: homework assignments, exam preparation, etc		
Credit points	2		
Recommended prerequisites	Object-Oriented Programming, Data Structures and Algorithms, Computer Networks, Software Engineering, Software Architecture, Principles of Database Systems		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1**: Ability to systematically analyze enterprise-level business logic and data flow, identify key challenges such as concurrency and transactions, and decompose them into implementable solutions and architecture selection justifications based on the Java technology stack.	**R2**
	CLO2	**Course Objective 2**: Master mainstream frameworks such as Spring Boot, efficiently complete coding, debugging, and refactoring using IntelliJ IDEA, and build highly cohesive, loosely coupled, and scalable distributed enterprise-level applications.	**R3**
	CLO3	**Course Objective 3:** With IntelliJ IDEA as the core tool, proficiently integrate Maven, Git, Docker and other tools, and utilize their advanced features for project building, version control, and containerized deployment.	**R5**

Content	**Course Description:** This course serves as a bridge for students to advance toward enterprise-level Java development. It systematically covers Servlets, Spring, Spring MVC, MyBatis, Spring Boot (core concepts, Web development, data access, security, messaging, deployment, monitoring), and Vue.js 3 fundamentals and advanced topics. Students complete three progressive case studies: SSM+JSP, Spring Boot+Thymeleaf+MyBatis, and Spring Boot+Vue.js 3+MyBatis, developing practical skills in building frameworks for diverse industry scenarios.To align with current technological trends, each teaching unit integrates AI engineering application topics, including large-model-assisted requirement analysis, code generation, automated testing, and system performance tuning. Outdated monolithic training cases have been replaced with university-enterprise collaborative projects that combine Java backend development with the full AI lifecycle, ensuring adherence to modern industry standards and equipping students with job-ready competencies. **Knowledge Module 1:** Dynamic Web Development Fundamentals (Weight: 4/32, Level: Comprehension + Application) **Knowledge Module 2:** JSP Technology (Weight: 4/32, Level: Memory + Comprehension + Application) **Knowledge Module 3:** Servlet Fundamentals (Weight: 6/32, Level: Memory + Comprehension + Application) **Knowledge Module 4:** Request and Response (Weight: 6/32, Level: Memory + Comprehension + Application + Creation) **Knowledge Module 5:** Session and Session Technologies (Weight: 6/32, Level: Comprehension + Application) **Knowledge Module 6:** EL Expressions and JSTL (Weight: 6/32, Level: Memory + Comprehension + Application)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is in the form of a course report, accounting for 50% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 10%; (2) regular assignments 10%; and (3) experiment report 30%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] LI Donghai, JIN Zongxin. Lightweight Java EE Web Framework Technologies. Beijing: Tsinghua University Press, 2022. [2] Ishu Anand Jaiswal. (2025). Integrating ai into enterprise java applications for secure high performance and scalable systems. *International Journal of Computational and Experimental Science and Engineering*, 11(4).
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Android Mobile Application Development

Module name	Android Mobile Application Development		
Code	3080902225		
Semester	Spring, 6th Semester		
Contact person	Wei Jinping, Yuan Qianmin		
Language	Chinese		
Relation to curriculum	Elective, Professional Elective Course		
Type of teaching, contact hours	Lecture Method, Demonstration Method, Group Discussion Method, Laboratory Experiment, Online Learning Method Teaching hours: 4 hours per week for 8 weeks, totaling 32 hours		
Work load	Total Workload (Estimated): 56 hours Teaching hours: 32 hours Self-study hours: 24 hours, including after-class assignments, exam preparation, etc.		
Credit points	2		
Recommended prerequisites	Web Front-end Development Technology, Object-Oriented Programming, Java Web Application Development		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**CLO1:** Master basic knowledge of Android programming, program examples, design methods and steps, and development techniques for Android applications.	**R2**
	CLO2	**CLO2**: Independently design and develop mobile applications using mainstream tools and languages (Java/Kotlin, XML); solve common issues in Android development, including performance optimization, UI design, data storage, and network security.	**R3**
	CLO3	**CLO3:** Able to use modern development tools such as Android Studio, combine relevant technologies and resources to complete the analysis, design, testing, implementation and basic maintenance of Android applications, and understand the applicable scenarios and limitations of such tools in the development of complex applications.	**R5**
	CLO4	**CLO4**: Can clearly write Android application development reports and design documents, accurately elaborate on development ideas and achievements, communicate technical issues with peers, possess cross-cultural communication awareness, and respond to development-related instructions and requirements promptly and appropriately.	**R9**

Content	This course is a specialized elective for Software Engineering majors, aiming to develop students' Android mobile application development capabilities. It covers Android OS architecture, Android App project setup, usage of Android API components, communication mechanisms, process/thread control, network programming, and database programming. Guided by real-world cases, students learn UI design, resource usage, database programming, network programming, multimedia, and game development. Upon completion, students understand the system architecture of mobile devices and core mobile development techniques, laying a solid foundation for careers and further study in Android App development. Knowledge Module 1: Android System Evolution and Architecture (Weight 2/32, Level: Understanding + Application) Knowledge Module 2: Development Basics (Weight 2/32, Level: Understanding + Application) Knowledge Module 3: Common UI Components (Weight 12/32, Level: Memory + Understanding + Application + Analysis + Evaluation) Knowledge Module 4: Activity (Weight 4/32, Level: Memory + Application + Analysis) Knowledge Module 5: Data Storage (Weight 4/32, Level: Understanding + Application) Knowledge Module 6: Messages, Notifications, Broadcasts, and Alarms (Weight 4/32, Level: Understanding + Application + Analysis) Knowledge Module 7: Animation and Multimedia (Weight 2/32, Level: Application + Analysis) Knowledge Module 8: Network Programming (Weight 2/32, Level: Understanding + Application + Analysis + Evaluation)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is in the form of a course report, accounting for 60% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 40% of the overall grade and includes: (1) class participation 12%; (2) regular assignments 12%; and (3) laboratory work 16%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Ouyang Shen. Android App Development: Introduction and Practical Projects. Tsinghua University Press, 2021, 1st Edition. [2] Peng Tao, Sun Lianying, Liu Chang. Android Application Development. Tsinghua University Press, 2021, 1st Edition. [3] Li Bin. Android Studio Mobile Application Development: A Task-Based Tutorial. Posts & Telecom Press, 2024, 1st Edition. [4] Nisha Kumari, Teerath Das, Tommi Mikkonen. An Empirical Study on Understanding Performance Issues and Causes in Android Apps. IEEE International Conference on Source Code Analysis and Manipulation (SCAM), 2024. [5] Souani B, Khanfir A, Bartel A, et al. Android Malware Detection Using BERT. International Conference on Applied Cryptography and Network Security (ACNS), Springer, 2022.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Python Programming

Module name	Python Programming		
Code	3080717219		
Semester	Spring, 6th Semester		
Contact person	Zhou Ying, Liu Yanfang		
Language	Chinese		
Relation to curriculum	Elective, Professional Expansion Course		
Type of teaching, contact hours	Lecture method, demonstration method, group discussion method, case analysis method, laboratory experiment method, tool-assisted teaching, personalized tutoring Teaching duration: 2 hours per week, totaling 16 weeks and 32 hours in total.		
Work load	Total workload (estimated): 56 credit hours Total class hours: 32 hours Self-study hours: 24 hours, including homework and exam preparation time		
Credit points	2		
Recommended prerequisites	C Language Programming, Data Structures and Algorithms, Principles of Operating Systems		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Course Objective 1: Through this course, students will master Python's syntax conventions, program flow control, data types, functions, file and data formatting, object-oriented programming, exception handling, as well as the Python computing ecosystem and essential libraries. They will develop proficiency in applying Python's programming mindset to solve real-world problems.	**R2**
	CLO2	Course Objective 2: To cultivate students' innovative abilities and their capacity to identify, analyze, and solve problems; to apply fundamental computer knowledge and programming skills in designing solutions, constructing theoretical models, conducting practical implementations, processing data, and addressing real-world application challenges.	**R3**
	CLO3	Course Objective 3: Become proficient in using common Python development environments such as PyCharm, Anaconda, and Visual Studio Code 2022; learn program debugging techniques, perform debugging operations on code, and carry out related tasks. Development of practical projects.	**R5**

Content	Python Programming is a core professional course that introduces the Python programming language and its applications. This course employs a teaching approach that integrates theory with practical application, enabling students to comprehend Python's programming paradigms, verify, understand, and ultimately master the knowledge acquired in class. Students will become proficient in using the Python development environment, leveraging fundamental data types—including lists, tuples, dictionaries, and sets—as well as features like list derivations and slicing to solve real-world problems; mastering Python's branching structures, loop constructs, function design, and class creation and usage; skillfully handling text and binary file operations; understanding Python debugging techniques; writing object-oriented programs; and becoming familiar with the basic usage of Python extension modules across various domains, thereby preparing them for future careers and academic pursuits. Knowledge Module 1: Overview of Python (Weight: 4/48; Level: Understanding) Knowledge Module 2: Fundamentals of Python Programming (Weight: 6/48; Level: Memory + Understanding) Knowledge Module 3: Programming Control Structures (Weight: 6/48; Level: Memory + Understanding + Application) Knowledge Module 4: Common Data Structures (Weight: 8/48; Level: Memory + Understanding + Application) Knowledge Module 5: Functions (Weight: 8/48; Level: Memory + Understanding + Application) Knowledge Module 6: Object-Oriented Programming (Weight: 8/48; Level: Memory + Understanding + Application) Knowledge Module 7: Files and Exceptions (Weight: 6/48; Level: Memory + Understanding + Application) Knowledge Module 8: Function Extension Module (Weight: 2/48; Level: Understanding + Application)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is conducted as an assessment component, accounting for 50% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class performance 15%; (2) regular assignments 15%; and (3) computer-based laboratory exercises 20%. Class performance is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Python Programming, edited by Chen Chunhui, Weng Kai, and Ji Jiangmin, Zhejiang University Press, 2021 [2] Introduction to Python Programming: Experimental Guide, edited by Zheng Jiangchao, Tsinghua University Press, 2021 [3] Task-Based Tutorial on Python Programming, edited by Zhang Yingxin, Tsinghua University Press, 2021 [4] Fundamentals and Applications of Python Programming, edited by Dong Fuguo, China Machine Press, 2019 [5] He Caixia, Lü Yanan, Cen Weifu. Practice and exploration of the PBL teaching model for "Python Programming" in the context of new engineering education [J]. Journal of Zunyi Normal University, 2025,27(05):144–149.

Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Big Data Collection and Processing

Module name	Big Data Collection and Processing		
Code	3080717260		
Semester	Spring, 6th Semester		
Contact person	Li Yao, Zhu Min, Sun Jilyu		
Language	Chinese		
Relation to curriculum	Elective,Professional Expansion Course		
Type of teaching, contact hours	Lecture method, demonstration method, case analysis method, laboratory experiment method Teaching duration: 2 hours per week, totaling 16 weeks, for a cumulative period of 32 hours.		
Work load	Total workload (estimated): 56 credit hours Total class hours: 32 hours Self-study duration: 24 hours, including: pre-class preparation, post-class review and consolidation, assignment completion, error identification and correction, supplementary resource exploration for enhancement, summary and gap identification/addressing, and exam preparation time.		
Credit points	2		
Recommended prerequisites	Python Programming, Principles of Database Systems, Data Structures and Algorithms		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Course Objective 1: Through the study of Big Data Collection and Processing, students will master the fundamental principles and engineering methods of data collection, preprocessing, storage, and analysis; be capable of applying mathematical, natural science, and engineering science principles to identify, articulate, and analyze complex engineering problems in artificial intelligence; draw effective conclusions through literature review and information organization; and develop rigorous data thinking skills and scientific research literacy.	**R2**
	CLO2	Course Objective 2: Through this course, students will master methods for multi-source data acquisition, data cleaning, data transformation, and system process design. They will be able to develop data processing models, design functional modules and system solutions tailored to real-world scenarios, demonstrate innovative thinking in their designs, comprehensively consider constraints such as social, safety, health, legal, cultural, and environmental factors, and enhance their ability to solve complex engineering problems.	**R3**

	CLO3	Course Objective 3: Through practical training in data collection, cleaning, analysis, and system development, students will develop strong technical communication skills, document preparation abilities, and interpersonal communication competencies; be capable of effectively addressing complex big data-related engineering challenges with peers and the public; possess a fundamental international perspective and cross-cultural communication skills; respect technical standards and cultural differences; and comprehensively enhance their overall engineering competence and teamwork capabilities.	**R9**
Content	This course is a comprehensive applied course designed to enhance students 'technical application skills. It enables students to understand the fundamental processes of Big Data Collection and Processing, master methods for multi-source data collection, data cleaning, data transformation, data standardization, and data preprocessing, become familiar with common big data collection tools and data processing frameworks, and, through practical experiments, perform data collection, cleaning, processing, and analysis tasks tailored to business requirements. The course aims to cultivate students' ability to integrate theoretical knowledge to analyze and solve real-world problems, while fostering and enhancing their innovative design capabilities and overall competence in combining theory with practice. Knowledge Module 1: Overview of Big Data and Fundamentals of Collection (Weight: 2/32; Level: Understanding) Knowledge Module 2: Multi-source Data Collection Methods and Tools (Weight: 6/32; Level: Understanding + Application) Knowledge Module 3: Data Cleaning and Exception Handling (Weight: 8/32; Level: Application + Analysis) Knowledge Module 4: Data Transformation, Integration, and Standardization (Weight: 10/32; Levels: Application + Analysis + Evaluation) Knowledge Module 5: Comprehensive Data Processing Practice (Weight: 6/32; Level: Application − Creation)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is in the form of a course report, accounting for 60% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 40% of the overall grade and includes: (1) class participation 12%; (2) regular assignments 12%; and (3) interim tests 16%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.		
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.		
Reading list	[1] Data Acquisition and Preprocessing (2nd Edition), edited by Lin Ziyu, People's Post and Telecommunications Press, 2025. [2] Big Data Collection and Preprocessing Technology, edited by Tang Shiwei, Tian Feng, Gai Xuan, and Li Xuegui, Tsinghua University Press, 2022. [3] Big Data Collection and ETL Technology Applications, edited by Li Junhua, Beijing Institute of Technology Press, 2025. [4] Introduction to Big Data (2nd ed.), edited by Mei Hong, Higher Education Press, 2024. [5] Wang Haofen, Chen Huajun. Research Progress on Key Technologies for Big Data Collection and Governance [J]. Journal of Computer Science, 2024.		

Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Introduction to Artificial Intelligence

Module name	Introduction to Artificial Intelligence		
Code	3080717245		
Semester	Fall, 6th Semester		
Contact person	Amjad Ali Amjad、DR AITIZAZ ALI, Shao Xiaofang		
Language	Chinese		
Relation to curriculum	Elective,Professional Expansion Course		
Type of teaching, contact hours	Lecture method, demonstration method, group discussion method, case analysis, laboratory experiment method Teaching duration: 4 hours per week, totaling 12 weeks and 48 hours in total.		
Work load	Total workload (estimated): 56 credit hours Teaching hours: 32 hours Self-study hours: 24hours, including homework and exam preparation time		
Credit points	2		
Recommended prerequisites	Linear Algebra, Advanced Mathematics		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Course Objective 1: To apply mathematics, natural sciences, fundamental engineering principles, and specialized knowledge to solve complex engineering problems in artificial intelligence-related fields.	**R1**
	CLO2	Course Objective 2: To apply fundamental principles from mathematics, natural sciences, and engineering sciences to identify, articulate, and analyze complex engineering problems in artificial intelligence-related fields; to retrieve relevant information through literature review, organize, summarize, and analyze it to derive valid conclusions.	**R2**
	CLO3	Course Objective 3: When addressing complex engineering problems, students shall be able to analyze and evaluate the impacts of engineering practices on health, safety, environment, legal considerations, and economic and social sustainability based on relevant engineering background knowledge, while understanding their associated responsibilities.	**R6**
	CLO4	Course Objective 4: To cultivate a sound political stance, worldview, outlook on life, and values; to practice the core socialist values; to develop competencies in humanities and social sciences along with a strong sense of social responsibility; and to understand and adhere to engineering ethics and standards in artificial intelligence practice while fulfilling professional responsibilities.	**R7**

	CLO5	Course Objective 5: Understand and master engineering management principles and decision-making methods in artificial intelligence-related fields, comprehend project life cycles, and apply them across multidisciplinary and cross-intelligence environments.	**R10**
Content	This course is a discipline with strong theoretical and practical components. Its theoretical sessions cover a wide range of fundamental principles, methods, and techniques in artificial intelligence; however, these concepts can seem abstract to students when taught solely through classroom lectures and are difficult to relate to solving real-world problems. Therefore, the teaching approach must consistently adhere to the principle of integrating theory with practice and concrete case studies. Specifically, key concepts from the textbook should be thoroughly understood and mastered by addressing specific problems in laboratory sessions, with the primary goal being to apply what has been learned practically. Knowledge Module 1: Overview of Artificial Intelligence (Weight: 2/32, Level: Understanding) Knowledge Module 2: Knowledge Representation and Reasoning (Weight: 6/32, Level: Understanding) Knowledge Module 3: Search and Inference Techniques (Weight: 8/32; Level: Memory) Knowledge Module 4: Deep Learning (Weight: 8/32, Level: Memory) Knowledge Module 5: Machine Learning and Knowledge Discovery (Weight: 8/32; Level: Applied)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is in the form of a course report, accounting for 60% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 40% of the overall grade and includes: (1) class participation 12%; (2) regular assignments 12%; and (3) interim tests 16%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.		
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.		
Reading list	[1] Fundamentals of Artificial Intelligence, edited by Yang Jie et al., Machinery Industry Press, 2020 [2] Principles and Methods of Artificial Intelligence, Wang Yongqing, Xi'an Jiaotong University Press, 2017 [3] Introduction to Artificial Intelligence, by Ma Shaoping and Zhu Xiaoyan, Higher Education Press, 2022 [4] Artificial Intelligence, by Nils J. Nilsson (USA), translated by Zheng Kuogen and Zhuang Yueting, Machinery Industry Press, 2018 [5] New Developments in Artificial Intelligence: From Large Models to General Intelligence [J], Wang Feiyue, Pattern Recognition and Artificial Intelligence, 2024		
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.		

Computer Science English

Module name	Computer Science English		
Code	3080900245		
Semester	Spring，7th Semester		
Contact person	Fang Xiaoxue, Amjad Ali Amjad, DR AITIZAZ ALI, AMEEN FADHL HASAN ALKASEM, Wei Liangping,Sun Jilu		
Language	English		
Relation to curriculum	Elective, Professional Expansion Course		
Type of teaching, contact hours	Lecture method, case analysis method, situational teaching method, group discussion method Teaching Hours: 2 hours per week for 16 weeks, totaling 32 hours		
Work load	Total workload (estimated): 56 credit hours Teaching hours: 32 hours Self-study hours: 24 hours, including pre-class preparation, post-class review and consolidation, assignment completion, supplementary material exploration for advancement, and exam preparation time.		
Credit points	2		
Recommended prerequisites	University English, C Language Programming, Principles of Operating Systems, Principles of Computer Networks		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	By integrating content on engineering ethics and professional ethics education, students are guided to not only master computer-specific English but also recognize engineers' social responsibilities. Through case studies and practical project exercises, their service-oriented mindset is cultivated, enabling them to apply professional knowledge to socioeconomic development.	R6

	CLO2	By reading and analyzing English scientific and technical literature, students develop engineering thinking skills and product research capabilities. Additionally, through practical activities based on real-world engineering project cases, their engineering practice abilities are enhanced.	R1
	CLO3	Through professional English instruction, we enhance students' communication skills in English, enabling them to communicate effectively with peers both domestically and internationally. Additionally, by staying abreast of global software technology trends and incorporating the latest English-language resources, we help students rapidly stay informed about cutting-edge technological advancements worldwide.	R9
	CLO4	Through team projects and case studies, students develop teamwork skills and software project management capabilities across multiple disciplines. By simulating real-world work environments, students can assume roles as core team members or team leaders.	R8
	CLO5	We cultivate students 'autonomous learning and lifelong learning capabilities by incorporating the latest technological advancements and encouraging participation in international exchanges, enabling them to keep pace with developments in software engineering. Additionally, innovative practical projects are designed to foster students' creativity and hands-on skills.	R11

Content	"Computer Professional English" integrates computer science expertise with English language skills. This course serves both as a fundamental professional course mandated by computer science curricula and as a crucial supplement to undergraduate students 'general English education. Its content primarily covers computer-related specialized knowledge and provides practical guidance for real-world applications. The prerequisite for this course is "College English," while subsequent requirements include reading English literature for graduation projects and acquiring foundational professional knowledge, thereby laying the groundwork for students' further research and development of specialized competencies. Through guided study of selected texts, the course enables students to familiarize themselves with and master computer-specific English terminology, enhance their ability to read technical English materials, and develop proficiency in interpreting professional English publications as well as performing translation tasks related to their field. Knowledge Module 1: Computer Hardware (Weight: 2/32; Level: Memory + Understanding) Knowledge Module 2: Computer Software and Software Development (Weight: 4/32; Level: Memory + Understanding + Application) Knowledge Module 3: Program Design, Programming Languages, and Computer Programming (Weight: 8/32; Level: Memory + Understanding + Application) Knowledge Module 4: Operating Systems (Weight: 2/32; Level: Memory + Application) Knowledge Module 5: Networking Devices and Topologies (Weight: 4/32; Level: Understanding + Application) Knowledge Module 6: Cloud Storage and Big Data (Weight: 4/32; Level: Memorization + Application) Knowledge Module 7: Data Mining (Weight: 2/32; Level: Understanding + Application) Knowledge Module 8: E-commerce (Weight: 2/32; Level: Memory + Application) Knowledge Module 9: AI (Weight: 4/32; Level: Memory + Understanding + Application)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is in the form of a course report, accounting for 60% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 40% of the overall grade and includes: (1) class participation 12%; (2) regular assignments 12%; and (3) interim tests 16%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.

Reading list	1. Computer Professional English [M], Zhang Qianghua, Si Aixia, China Machine Press, 2024. 2. Computer Professional English [M], by Jin Zhiquan and Zhang Xing'er, Electronic Industry Press, 2020. 3. Downey, Allen B. Think Python: How to Think Like a Computer Scientist[M]. O'Reilly Media, 2024. 4. Howe P N .ChatGPT has a language problem - but science can fix it.[J].Nature,2024. 5. Chi G .The Spoken English Practice System Based on Computer English Speech Recognition Technology[J].Mobile Information Systems,2022. 6. Xie L ,Shang Z,Wang X.Research on the Teaching of Computer English under the Network Environment[J].Higher Education and Practice,2024. 7. Wenxia L ,Weiming S .The English Teaching Mode under the Environment of Computer Technology[J].Mobile Information Systems,2022.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Fundamentals of Information Security Technology

Module name	Fundamentals of Information Security Technology		
Code	3120105258		
Semester	Fall, 7th Semester		
Contact person	Chai yan, Zheng Haiyan, Qian Yuying		
Language	Chinese		
Relation to curriculum	Elective, Professional Expansion Course		
Type of teaching, contact hours	lecture method, Case Analysis Method，Laboratory experimental method Teaching hours: 2 hours per week for 16 weeks, with a total of 32 hours.		
Work load	Total Estimated Workload: 56 credit hours Teaching Hours: 32 hours Self-study Hours: 24 hours, including after-class assignments, exam preparation and other related learning activities.		
Credit points	2		
Recommended prerequisites	Computer Networks, Principles of Operating Systems, Principles of Database Systems		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Course Objective 1: This course aims to enable students to master the basic principles and technical methods of information security. Students will be able to apply fundamental knowledge of mathematics, computer science and software engineering to identify security risks and complex engineering problems in software systems. They can analyze security vulnerabilities, attack mechanisms and protection schemes through literature research, standardize problem expression and draw valid conclusions, and develop preliminary abilities to analyze and solve complex information security problems in the field of software engineering.	**R2**
	CLO2	Course Objective 2: This course enables students to master the design principles of information security, design software security solutions and security algorithm processes, and integrate innovative thinking into the design process. Meanwhile, students shall comprehensively take into account security, legal, social, health and other factors to meet the design requirements of complex engineering problems in software engineering.	**R3**

	CLO3	Course Objective 3: This course cultivates students' ability to apply software engineering theories and scientific methods to research information security issues. Students can reasonably design experimental schemes, analyze experimental data, and draw valid conclusions by integrating relevant information, so as to acquire the basic competence for conducting research and experimental verification on security-related engineering problems.	**R4**
	CLO4	Course Objective 4: This course helps students cultivate humanistic literacy and social responsibility, understand engineering ethics in the field of information security, master relevant laws and regulations related to network security. It enables students to abide by professional ethics and norms in software engineering practice, consciously fulfill safety responsibilities, and carry out development and operation maintenance work in a legal, compliant and responsible manner.	**R7**
Content	This course focuses on how to protect data from unauthorized access, tampering, destruction and leakage. With the rapid development of digitalization and internet technologies, information security issues have become increasingly prominent, and enterprises and organizations have a growing demand for information security professionals.The fundamental course of information security technology is designed to cultivate professionals equipped with information security knowledge and practical skills. The curriculum covers a wide range of contents, including basic knowledge of information security, cryptography, digital signature and identity authentication, key management, access control, system security, application security, as well as laws and regulations on information security.Through the study of this course, students can understand the basic concepts, principles and technologies of information security, and master practical operational skills such as data encryption, data backup and recovery, and emergency response to data leakage. Module 1: Introduction (Weight: 1/32, Level: Comprehension) Module 2: Overview of Cryptography (Weight: 4/32, Level: Memorization & Comprehension) Module 3: Digital Signature (Weight: 6/32, Level: Memorization-Comprehension-Analysis) Module 4: Identity Authentication Technology (Weight: 6/32, Level: Memorization-Comprehension-Analysis) Module 5: Message Authentication Technology (Weight: 6/32, Level: Memorization-Comprehension-Analysis) Module 6: Key Management (Weight: 6/32, Level: Memorization-Comprehension-Analysis) Module 7: Network Security (Weight: 1/32, Level: Comprehension-Application) Module 8: Data Security (Weight: 2/32, Level: Comprehension-Application)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is in the form of a course report, accounting for 50% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 15%; (2) regular assignments 15%; and (3) interim tests 20%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.		

Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1]Chen Y, Yang K W, Hu X X. Data Security Theory and Technology. Science Press, 2023. [2] Chen Z, Zou H, Zhang X Q, Zhang J F, Huang Y J, Liu H B. Data Security and Governance. Tsinghua University Press, 2022. [3] Yue S. Analysis on multi-scenario adaptive application of data encryption technology in computer network information security[J]. China Computer & Communication, 2026, 38(07): 124-126. [4] Xia Q. Research on enhanced data information security technology in smart substation[D]. Beijing University of Posts and Telecommunications, 2025. doi:10.26969/d.cnki.gbydu.2025.001103. [5] Du F. China Telecom innovates to build a new line of defense for information security[N]. Communication Information News, 2024-12-04(004). Special Topic. doi:10.28808/n.cnki.ntxxx.2024.000500.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Internet of Things Applications

Module name	Internet of Things Applications		
Code	3080903335		
Semester	Fall, 7th Semester		
Contact person	Wang Xinjiao, Qiao Qun, Zheng Haiyan, Xu Shaojun		
Language	Chinese		
Relation to curriculum	Elective, Professional Expansion Course		
Type of teaching, contact hours	Lecturing method；Case Analysis Method；Simulation Experiment Method Teaching hours: 2 hours per week for 16 weeks, with a total of 32 hours.		
Work load	Total Estimated Workload: 56 credit hours Teaching Hours: 32 hours Self-study Hours: 24 hours, including homework, examination preparation and other related learning activities.		
Credit points	2		
Recommended prerequisites	《C Language Programming》		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1**: Students can apply the principles of mathematics, natural sciences and software engineering to identify and standardly express complex engineering problems in IoT applications. They shall conduct analysis through literature research, sort out key technologies and application pain points, and draw scientific and reasonable conclusions independently.	**R2**
	CLO2	**Course Objective 2**: Guide students to study complex engineering problems of the Internet of Things based on software engineering theories and scientific methods. Students will rationally plan research directions, design relevant verification experiments, analyze experimental data, integrate multi-source information, summarize results, and form objective, reasonable and scientific research conclusions.	**R4**

	CLO3	Course Objective 3: Enable students to combine the background knowledge of Internet of Things and software engineering, analyze and evaluate the impacts of IoT engineering practice on security, environment, law, economy and social development, identify potential risks, establish the concept of sustainable development, clarify engineering responsibilities, and carry out IoT application development practice rationally.	**R6**
	CLO4	Course Objective 4: Master the basic methods of IoT software project management and understand the economic and management elements in project implementation. Combined with multidisciplinary integration scenarios, students will learn to coordinate the development process of IoT projects, allocate resources rationally, and apply management concepts to IoT software engineering practice.	**R10**
Content	This course is a required course for Network Engineering. Its main contents include automatic identification technology and RFID, wireless communication systems, wireless single-chip microcomputers, sensor technology, positioning systems, short-range wireless communication technologies, embedded intelligent sensors, communication and telecommunication, and intelligent information processing technology. Through this course, students can gain a comprehensive understanding of the Internet of Things, and master its architecture, basic concepts and related technologies. By learning automatic identification and RFID, sensing technology and positioning systems, students will acquire basic knowledge of the perception and identification layer. The study of wireless broadband networks and mobile communication networks helps them grasp the fundamentals of the network construction layer. Learning big data and mass information storage, database systems, information security and privacy enables students to understand the management and service layer. In addition, by exploring intelligent transportation, logistics, buildings and other systems, students will learn the multi-field applications of IoT technology and improve their adaptability to the evolving Internet of Things. Knowledge Module 1: Overview of Internet of Things (Weight: 1/48, Level: Comprehension) **Knowledge Module 2:** RFID Technology (Weight: 2/48, Level: Memory + Comprehension) **Knowledge Module 3:** Wireless Communication Systems (Weight: 1/48, Level: Comprehension – Application – Analysis) **Knowledge Module 4:** Wireless Single-chip Microcomputer Technology (Weight: 4/48, Level: Comprehension – Analysis) **Knowledge Module 5:** Sensor Technology (Weight: 4/48, Level: Memory + Comprehension + Analysis) **Knowledge Module 6:** Wireless Sensor Network Technology (Weight: 4/48, Level: Memory + Comprehension) **Knowledge Module 7:** Short-range Wireless Communication Technology (Weight: 3/48, Level: Memory + Comprehension) **Knowledge Module 8:** Telecommunication Technology (Weight: 3/48, Level: Memory + Comprehension) **Knowledge Module 9:** Intelligent Information Processing Technology (Weight: 6/48, Level: Memory + Comprehension) **Knowledge Module 10:** Applications of Internet of Things Technology (Weight: 6/48, Level: Memory + Comprehension)		

Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is in the form of a course report, accounting for 50% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 15%; (2) regular assignments 15%; and (3) interim tests 20%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Xu Yinqin, Xiong Weili. Internet of Things Technology and Application (2nd Edition)[M]. Beijing: China Machine Press, 2020. [2] Liu Yunhao. Introduction to the Internet of Things (3rd Edition)[M]. Beijing: Science Press, 2017. [3] Li Zhicheng. Research on Development and Performance Optimization of IoT Application System Based on Cloud Platform[J]. China Broadband, 2026, 22(05): 164-166. DOI:10.20167/j.cnki.ISSN1673-7911.2026.05.55. [4] Brief Introduction to Internet of Things Technology[J]. Internet of Things Technology, 2026, 16(08): 164. [5] Sun Xiaoqi. Reform of Practical Courses for Higher Vocational IoT Application Technology Major Based on Virtual Simulation System—Taking Wireless Transmission Technology Course as an Example[J]. Journal of Liaoning Teachers College (Natural Science Edition), 2026, 28(01): 54-58+63.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Innovative Practices in AIGC Applications

Module name	Innovative Practices in AIGC Applications		
Code	3080902307		
Semester	Fall, 5th Semester		
Contact person	Yuan Qianmin, Shao Xiaofang		
Language	Chinese		
Relation to curriculum	Compulsory , Entrepreneurship and Innovation Education Course		
Type of teaching, contact hours	Lecture Method, Demonstration Method, Group Discussion Method, Project-Based Learning Method Teaching Hours: 2 hours per week, 16 weeks in total, 32 hours in total		
Work load	Total Workload (Estimated): 56 credit hours Teaching Hours: 32 hours Self-study Hours: 24 hours, including: Homework, Exam Preparation Time, etc. Each teaching hour is calculated as 1 hour here		
Credit points	2		
Recommended prerequisites	Python Programming		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes	Description	Supported PLOs
	CLO1	Course Objective 1: Master the core technical principles of AIGC, be familiar with the functions and application scenarios of mainstream AIGC tools, understand the application boundaries and ethical norms of AIGC in different industries, and establish a systematic cognition from technology to application to innovation.	**R2**
	CLO2	Course Objective 2: Possess the practical ability to independently operate mainstream AIGC tools to complete basic creation and scenario adaptation, be able to decompose tasks according to specific needs, use AIGC technology to solve practical problems, and improve the practical ability of AIGC tool optimization and secondary application.	**R5**
	CLO3	Course Objective 3: Cultivate innovative thinking in AIGC applications, be able to explore innovative application scenarios of AIGC in specific fields combined with industry pain points and their own professional characteristics, complete the design of simple AIGC application innovation schemes, and improve the ability of innovative practice and scheme implementation.	**R3**

Content	This course focuses on the practical application and innovative practice of AIGC (Generative Artificial Intelligence), based on the core orientation of "technology implementation, scenario innovation, and ability improvement", connecting with preliminary basic courses such as Computer Science, Python, and Deep Learning, and balancing theoretical cognition and practical implementation. The course will systematically explain the core technical logic of AIGC and the application of mainstream tools, guide students to explore the innovative application path of AIGC combined with industry scenarios, cultivate students' ability to use AIGC technology to solve practical problems and design innovative schemes, and help students master the necessary practical skills and innovative thinking in the AIGC era, adapting to the AIGC application needs in multiple fields such as new media, design, education, and technology. Knowledge Module 1: Core Cognition and Industry Application of AIGC 4/48 Memory + Understanding Knowledge Module 2: Basic Operation of Mainstream AIGC Tools 8/48 Understanding + Application Knowledge Module 3: Practical Application of AIGC in Scenarios 12/48 Understanding + Application + Analysis Knowledge Module 4: Innovative Thinking and Scheme Design of AIGC Applications 16/48 Application + Analysis + CreationKnowledge Module 5: Implementation and Achievement Display of AIGC Innovation Schemes 8/48 Application + Analysis + Evaluation
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is in the form of a course report, accounting for 50% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 15%; (2) regular assignments 15%; and (3) interim tests 20%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.

Reading list	[1] General Knowledge of Artificial Intelligence and AIGC Applications, Edited by Wang Lei, He Zongyao, Tsinghua University Press [2] Practical Application of Generative Artificial Intelligence (MOOC Edition), compiled by He Jing, Posts & Telecommunications Press. [3] General Textbook on AIGC Technology and Applications (Microcourse Edition), compiled by Song Yanfeng, Xu Tingyu, Tsinghua University Press. [4] AIGC: Make Generative AI Your External Brain, written by Cheng Shenghui, Tsinghua University Press. [5] Fundamentals and Applications of Generative Artificial Intelligence, edited by Li Li, Yin Gen, Ma Qingxiang, Higher Education PressRefer to the textbooks and main teaching reference materials in the course syllabus here.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Internet+ Innovative Application

Module name	**Internet+ Innovative Application**		
Code	3080903332		
Semester	Spring, 6th Semester		
Contact person	Wang Xiaoqing, Su Nannan, Su Wanli, Xu Shaojun		
Language	Chinese		
Relation to curriculum	Compulsory, Entrepreneurship and Innovation Education Course		
Type of teaching, contact hours	Lecture, demonstration, group discussion, project-based learning. Teaching hours: 2 hours per week, 16 weeks, total 32 hours.		
Work load	Total workload (estimated): 56 hours Teaching hours: 32 hours Self-study hours: 24 hours (including assignments, exam preparation, etc.)		
Credit points	2		
Recommended prerequisites	Principles of Computer Composition, Computer Networks		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes	Description	Supported PLOs
	CLO1	Through course learning and practice, cultivate students' innovative thinking, problem-solving abilities, teamwork and communication skills. Enable students to apply knowledge to propose innovative solutions for practical problems.	R2
	CLO2	Through lab sessions and project practice, enable students to master Internet-based innovation application skills such as data analysis, user research, product design, and development. Cultivate practical abilities and innovation/entrepreneurship competence.	R3
	CLO3	Enable students to deeply understand the application and development trends of "Internet Plus" across different industries, such as e-commerce, Internet finance, smart logistics, online education, etc. Enhance sensitivity and insight into industry evolution, and prepare for future career development.	R8

Content	This course involves multiple disciplines including computer science, business management, sociology, etc., offering a comprehensive and integrated perspective on innovative applications in the "Internet Plus" era. It focuses not only on technical aspects but also on how technology deeply integrates with society, economy, culture and other fields to generate new value and innovation. The core of this course lies in innovation. It encourages students to challenge conventional thinking, experiment with new methods and technologies to address various challenges and opportunities brought by the "Internet Plus" era. Through the course, students will master innovative thinking and methodologies, enabling them to independently discover and solve problems. The course emphasizes practical application. It is not merely theoretical knowledge acquisition but also practical skill development. Through case studies, hands-on exercises, and projects, students experience the actual process of "Internet Plus" innovation applications, enhancing their practical and problem-solving capabilities. Since "Internet Plus" is a rapidly evolving field, this course maintains forward-looking awareness, tracking the latest technological trends and market dynamics. It not only teaches current knowledge and skills but also cultivates students' ability to adapt to future changes, empowering them to remain competitive in a constantly changing environment.
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination is in the form of a course report, accounting for 50% of the overall grade. Students are required to submit a course report. The continuous assessment component accounts for 50% of the overall grade and includes: (1) class participation 15%; (2) regular assignments 15%; and (3) interim tests 20%. Class participation is evaluated comprehensively based on students' contributions to class discussions, the quality of their questions and answers, their participation in group discussions, and their classroom discipline.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the course.

Reading list	Textbooks / reference books: [1] Yu Laiwen, Lin Xiaowei, Internet Thinking 2.0: IoT, Cloud Computing, Big Data (Internet+ and Business Model Application Series), Economic Management Press, 2017. [2] Gao Zehan, Hui Gangxing, Fundamentals and Applications of "Internet Plus", Xidian University Press, 2018. [3] Yan He, Li Guihua, Internet+ Era: Mobile Internet Transformation for Traditional Enterprises, Tsinghua University Press / Publishing House of Electronics Industry, 2017. [4] Si Xiao et al., Internet + Manufacturing: Toward Made in China 2025. [5] Chen Wang, Implementation Path and Application Strategy Research on "Internet Plus" Business Model Innovation of Traditional Decoration Enterprises [D]. Southeast University, 2019. DOI:10.27014/d.cnki.gdnau.2019.004329. [6] Gao Sipeng, Research on the Ecosystem of "Internet Plus Government Services" from the Perspective of Holistic Government [D]. Party School of the Central Committee of CPC, 2020. DOI:10.27479/d.cnki.gzgcd.2020.000159. Online resources: In addition to textbooks and references, the course provides online platforms (Chinese University MOOC, Zhihuishu, Rain Classroom, etc.) for students to expand knowledge and perspectives based on individual needs. • Chinese University MOOC: https://www.icourse163.org • Zhihuishu: https://www.zhihuishu.com
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Basic Training in Programming

Module name	Basic Training in Programming		
Code	3080902504		
Semester	Spring, 2nd Semester		
Contact person	Yuan Qianmin, Wang Xin		
Language	Chinese		
Relation to curriculum	Compulsory, Integrated Practical Course		
Type of teaching, contact hours	Lecture Method, Demonstration Method, Group Discussion Method, Laboratory Experiment, Online Learning Method Teaching hours: 4 hours per week for 8 weeks, totaling 32 hours		
Work load	Total Workload (Estimated): 84 hours Teaching hours: 32 hours Self-study hours: 52 hours, including after-class assignments, exam preparation, etc.		
Credit points	3		
Recommended prerequisites	C Language Programming		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**CLO1：** Master core C syntax and three basic program structures, independently write and debug basic programs, follow code formatting standards, and acquire basic programming and error-fixing skills.	**R1**
	CLO2	**CLO2：** Master modular programming with arrays, functions, pointers, and structures; design and implement medium-complexity programs; split modules, design interfaces, and optimize programs.	**R2**
	CLO3	**CLO3：** Solve practical problems with basic algorithms, develop small integrated projects, and conduct program testing, verification, and simple optimization.	**R3**
	CLO4	**CLO4：** Able to select and properly use modern programming, development and debugging tools for coding, debugging, testing and implementation; understand the scope and limitations of tools; maintain standard programming practices and engineering competence, and apply modern tools to solve practical programming problems.	**R5**

Content	This course is a compulsory concentrated practice course for Software Engineering majors, serving as the practical extension and reinforcement of C Language Programming. It focuses on core C syntax, program debugging, modular programming, and basic algorithm implementation. Through structured practical projects, students are trained in coding, debugging, logical analysis, and problem-solving. Emphasizing hands-on practice, the course helps students master sequential, selective, and loop structures, as well as arrays, functions, pointers, and structures, while standardizing programming habits and improving code readability and robustness. It bridges the gap from syntax understanding to code implementation, laying a solid practical foundation for subsequent courses such as Data Structures and Algorithm Design. Knowledge Module 1: Introduction and Development Environment (Weight 2/32, Level: Understanding + Application) Knowledge Module 2: Basic C Structures (Weight 4/32, Level: Memory + Understanding + Application) Knowledge Module 3: Arrays and Strings (Weight 4/32, Level: Memory + Understanding + Application) Knowledge Module 4: Modular Programming with Functions (Weight 6/32, Level: Application + Analysis + Evaluation) Knowledge Module 5: Pointers and Structures (Weight 6/32, Level: Understanding + Application + Analysis) Knowledge Module 6: Comprehensive Project Practice (Weight 4/32, Level: Application + Analysis + Evaluation)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Tan, H. Q. C Programming (5th ed.). Tsinghua University Press, 2017. [2] Feng, X. Z. C Language Programming Study Guide and Experimental Tutorial (5th ed.). Tsinghua University Press, 2021. [3] Reek, K. A. Pointers on C. Posts & Telecom Press, 2020. [4] Liu, J., Zhao, Y. Q., & Liu, J. G. Teaching Reform and Exploration of "C Programming" Course Based on OBE Concept. Educational Theory and Practice, 2022, 42(03): 61–63. [5] Wang, R., & Huang, J. Q. Exploration of Ideological and Political Teaching in C Language Programming Course. Computer Education, 2022(01): 74–77+83.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Basic Training in Data Structures and Algorithms

Module name	Basic Training in Data Structures and Algorithms		
Code	3080902502		
Semester	Spring, 2nd Semester		
Contact person	Liu Yanfang, Diao Tianru		
Language	Chinese		
Relation to curriculum	Compulsory, Integrated Practical Course		
Type of teaching, contact hours	Case Analysis Method； Situational Teaching Method； Project-Based Learning (PBL) Teaching hours: 2 hours per week, 16 weeks in total, 32 hours total		
Work load	Total estimated workload: 84 class hours Teaching hours: 32 hours (total hours in the teaching schedule) Self-study hours: 52 hours, including after-class assignments, exam preparation, etc.		
Credit points	3		
Recommended prerequisites	C Language Programming, Data Structures and Algorithms		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Course Objective 1: This course aims to enable students to master the core knowledge of data structures and algorithms, and to use mathematical modeling and professional methods to analyze, design and implement efficient algorithms for solving complex engineering problems in software engineering. It cultivates students' abilities in abstract modeling, complexity analysis and program optimization, supports the achievement of Graduation Requirement R1 (Engineering Knowledge), and lays a foundation for subsequent professional study and engineering practice.	**R1**
	CLO2	Course Objective 2: This course cultivates students' ability to design algorithm and data structure solutions for complex software problems, enabling them to rationally design algorithmic processes and software modules with innovative ideas in implementation. It trains students to take into account safety, efficiency and standards, and consider social, safety, environmental and other constraints in design, so as to form complete and feasible engineering solutions, supporting the achievement of design/development solution indicators in graduation requirements.	**R3**

	CLO3	Course Objective 3: This course enables students to master common algorithms and data structure tools, and to select and apply appropriate programming and debugging tools to complete algorithm design, implementation and testing. It trains students to analyze complex engineering problems with the help of modern information technology tools, understand the applicable scenarios and limitations of tools, and carry out software modeling, coding and verification in a standardized manner, so as to support the achievement of the graduation requirement "Using Modern Tools".	**R5**
	CLO4	Course Objective 4: Through group practice and project training, this course cultivates students' awareness of teamwork and communication skills, enabling them to clarify responsibilities and cooperate efficiently in multidisciplinary teams. It guides students to complete tasks independently and play their roles competently in teamwork, with a strong sense of responsibility and collaboration ability, so as to support the achievement of the "Individual and Team" indicator in graduation requirements.	**R8**
Content	《Basic Training in Data Structures and Algorithms》 is an important practical supporting course for the Software Engineering major. It aims to strengthen students' abilities in data organization, algorithm design and engineering implementation. The course provides practical training focusing on linear structures, tree structures, graph structures, as well as common search and sorting algorithms. Combining mathematical modeling and software engineering methods, it guides students to abstract, analyze and model complex engineering problems. Through computer programming, module implementation and comprehensive project practice, students can master the skills of algorithm design, process planning, component implementation and program debugging, and learn to use professional tools to complete software development and optimization. The course focuses on cultivating engineering thinking, innovation awareness and teamwork ability, enabling students to achieve comprehensive literacy in identifying key problems, designing solutions and implementing software functions, so as to lay a solid foundation for follow-up professional courses and software engineering practice. Knowledge Module 1: Linear List (Weight: 2/32, Levels: Understanding, Application, Analysis) Knowledge Module 2: Stack and Queue (Weight: 2/32, Levels: Understanding, Application, Analysis) Knowledge Module 3: String (Weight: 2/32, Levels: Understanding, Application, Analysis) Knowledge Module 4: Array and Generalized List (Weight: 2/32, Levels: Understanding, Application, Analysis) Knowledge Module 5: Trees and Binary Trees (Weight: 6/32, Levels: Understanding, Application, Analysis) Knowledge Module 6: Graphs (Weight: 6/32, Levels: Understanding, Application, Analysis) Knowledge Module 7: Searching (Weight: 6/32, Levels: Understanding, Application, Analysis) Knowledge Module 8: Internal Sorting (Weight: 6/32, Levels: Understanding, Application, Analysis)		

Forms of examination	This course adopts a combination of process assessment and final assessment to conduct a comprehensive evaluation of students' completion of practical projects, application of theoretical knowledge involved in the projects, and mastery of practical operation skills. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Yan Weimin, Li Dongmei, Wu Weimin. Data Structures (C Language Edition). Beijing: Posts & Telecom Press. [2] Yan Wujun, Li Jianchang, Wang Jiahui. The "Trinity + Guided" Hybrid Teaching Model of Data Structure and Algorithm Experimental Course[J]. Computer Education, 2025(02): 128-133. DOI:10.16512/j.cnki.jsjjy.2025.02.038. [3] Yang Fan, Bao Yongqiang. Design Method of Project-Based Teaching Model for Data Structure Course[J]. The Theory and Practice of Innovation and Entrepreneurship, 2024, 7(01): 135-137. [4] Lanqiao Cup Competition Organizing Committee, Luo Yongjun, Yang Peilin. Special Challenge Tutorial for Programming Contests[M]. Posts & Telecom Press, 2023(01): 377. [5] Data Structures (C Language, Micro-Lecture Edition): From Concepts to Algorithms. Beijing: Posts & Telecom Press, 2023.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Basic Training in Software Front-end Development

Module name	Basic Training in Software Front-end Development		
Code	3080902503		
Semester	Spring , 2nd Semester		
Contact person	Sun Yingying, Cui Zhenzhen, Zhu Xuesheng, Jiang Zhijun		
Language	Chinese		
Relation to curriculum	Compulsory，Integrated Practical Course		
Type of teaching, contact hours	Teacher-centered method (lecture), practical method (case analysis, project practice). Contact hours: 2 hours per week for 16 weeks , totaling 32hours.		
Work load	Total estimated workload: 84hours Teaching hours: 32 hours Self-study hours:52 hours, including homework, exam preparation, etc		
Credit points	3		
Recommended prerequisites	C Language Programming		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Students can identify basic requirements and technical problems in frontend page development, and sort out common issues in web layout, style adaptation and basic interaction. They can analyze causes with basic frontend knowledge, evaluate solutions to simple development problems, and develop independent problem-analyzing ability in basic frontend development.	**R2**
	CLO2	Students can design and develop static and simple dynamic pages using HTML, CSS and JavaScript according to web development requirements. Following frontend development specifications, they can implement page layout, styling and basic interaction, and acquire the ability to design and develop simple frontend pages.	**R3**
	CLO3	Students can proficiently use mainstream frontend development tools such as VS Code and browser debugging tools to troubleshoot style and code errors. They can master basic code management tools and efficiently complete frontend coding, debugging and optimization with modern development tools.	**R5**
	CLO4	Students can accurately clarify development task requirements, communicate requirements and report difficulties clearly. They can effectively cooperate in team projects, coordinate progress, and output standardized development documents, thus obtaining professional communication and teamwork competence in frontend development.	**R9**

Content	this course is a core practical training course for beginners in frontend development. It focuses on basic frontend technologies and provides practical teaching on web layout, style design and basic interaction. It integrates tool usage, problem troubleshooting and teamwork training to lay a solid foundation for frontend development and cultivate students' practical abilities in basic page development, problem analysis and teamwork. Knowledge Module 1: Introduction to Frontend Development and Development Environment Setup (weight 2/32, level: Remember + Understand) Knowledge Module 2: HTML Web Structure Construction (weight 4/32, level: Remember + Understand + Apply) Knowledge Module 3: Basic CSS Style Setting (weight 4/32, level: Remember + Understand + Apply) Knowledge Module 4: Web Layout and Responsive Design (weight 8/32, level: Understand + Apply) Knowledge Module 5: Basic JavaScript Syntax and Variable Data (weight 2/32, level: Remember + Understand) Knowledge Module 6: JS Control Flow and Function Encapsulation (weight 4/32, level: Understand + Apply) Knowledge Module 7: Basic Frontend DOM Interaction Development (weight 4/32, level: Understand + Apply) Knowledge Module 8: Comprehensive Frontend Project Training, Debugging and Optimization (weight 4/32, level: Remember + Understand + Apply)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Web Design and Web Frontend Development Case Tutorial, edited by Mo Xiaomei, Tsinghua University Press, 2025, 2nd Edition. [2] Web Frontend Technologies, edited by Wang Qi, Wang Chunhong, Xidian University Press, 2024. [3] Professional JavaScript for Web Developers, by Nicholas C. Zakas, People's Posts and Telecommunications Press, 2024, 5th Edition. [4] Mobile Web Frontend Design and Development Based on HTML5 Technology, by Luo Haixia, Science & Technology and Innovation, 2024. [5] Web Frontend Development Technology and Optimization Based on Website Production, by Han Yinghong, Software, 2022. [6] HTML5 Web Frontend Design, edited by Zhou Wenjie, Tsinghua University Press, 2025, 3rd Edition.

Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Basic Training in Database Application

Module name	Basic Training in Database Application		
Code	3080902505		
Semester	Fall, 3rd Semester		
Contact person	Wan Ming'ao，Zhu Xiaoxia,Wang Fengqin, Fan deming		
Language	Chinese		
Relation to curriculum	Compulsory, Integrated Practical Course		
Type of teaching, contact hours	Lecture Method、Demonstration Method、Case Analysis Method、Simulation Experiment Method Class hours: 4 hours per week, 12 weeks in total, totaling 48 hours		
Work load	Estimated Total Workload: 112 class hours Teaching Hours: 48 hours Independent Study Hours: 64 hours, including homework, exam preparation and other self-study tasks.		
Credit points	4		
Recommended prerequisites	C Language Programming, Data Structures and Algorithms		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Apply the relevant principles of mathematics, natural sciences and software engineering, identify and express complex engineering problems in the field of databases in combination with business scenarios, systematically analyze the problems and clarify constraints through literature research and other methods, and finally form logically rigorous analysis conclusions to provide support for subsequent database design and development.	**R2**
	CLO2	For the complex engineering problems related to databases identified in the training, students need to combine business requirements to complete the design of database conceptual, logical, physical structures and core algorithm processes, and integrate innovative awareness into the design to optimize the scheme; at the same time, they should comprehensively consider various factors such as society, law and environment to ensure that	**R3**

		the scheme is compliant, practical and scalable, and balance data security and rational use of resources.	
	CLO3	Based on the relevant theories of software engineering and databases, adopt scientific methods to conduct research on complex problems in the field of databases, independently select research routes, design experimental schemes, collect and analyze experimental data, and finally form reasonable and effective research conclusions through information synthesis to provide support for improving the performance of database systems.	**R4**
	CLO4	For complex engineering problems related to databases, select and use appropriate modern tools to complete the entire database workflow, while understanding the limitations of various tools, flexibly choosing tools to avoid problems, and improving work efficiency and the rationality of solutions.	**R5**
Content	Based on the basic theories, technologies and methods of database systems, this course focuses on practical training, focusing on cultivating students' practical database operation capabilities and engineering application capabilities. It requires students to proficiently master the practical skills of relational data theory, database protection, database design and database management systems. Through the training of this course, students will proficiently master the practical skills in database basic knowledge, database design, database management and maintenance, and realize the transformation of theoretical knowledge into practical capabilities. **Knowledge Modules** **Module 1:** Overview of Database SystemsWeight: 4/48Competence Level: Memorization & Comprehension **Module 2:** Relational DatabasesWeight: 4/48Competence Level: Memorization, Comprehension & Application **Module 3:** Fundamentals of SQL Server DatabaseWeight: 4/48Competence Level: Memorization & Comprehension **Module 4:** Standard Database Language SQLWeight: 24/48Competence Level: Memorization, Comprehension, Application & Analysis **Module 5:** Relational Database Design TheoryWeight: 4/48Competence Level: Memorization, Comprehension & Application **Module 6:** Database DesignWeight: 4/48Competence Level: Memorization, Comprehension, Application & Analysis **Module 7:** Database ManagementWeight: 4/48Competence Level: Memorization, Comprehension & Application		

Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	1.Wang Fengling. Principles and Applications of Database Systems [M]. Xidian University Press, 2022. 2.Wang Shan,Du Xiaoyong,Chen Hong. Introduction to Database Systems (6th Edition) [M]. Higher Education Press, 2023. 3.Li Guoliang, Feng Jianhua, Chai Chengliang. Database System Theory: From Fundamental Principles to System Construction[M]. Higher Education Press, 2024. 4.Wang Xuhui. Database Technology and Its Application in the Big Data Environment [J]. Information Recording Materials, 2023, 24(03). 5.Wu Dan. The Application Value and Methods of Distributed Database Technology [J]. Digital Communication World, 2023(04).
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Linux System Programming Training

Module name	Linux System Programming Training		
Code	3080902513		
Semester	Fall, 3rd Semester		
Contact person	Liu Qianqian,Zhu Xuesheng,Liu Xing,You Haidong		
Language	Chinese		
Relation to curriculum	Compulsory, Integrated Practical Course		
Type of teaching, contact hours	Lecture Method, Case Analysis Method, Laboratory Experiment Method, Project-Based Learning Method Teaching hours: 8 hours per week, 4 weeks in total, 32 hours		
Work load	Total estimated workload: 84 hours Teaching hours: 32 hours Self-study hours: 52 hours (including after-class assignments, exam preparation, etc.)		
Credit points	3		
Recommended prerequisites	C Language Programming, Principles of Operating Systems, Data Structures and Algorithms		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Target1** : Master basic Linux commands, development environment, compilation and debugging tools, and be able to build a Linux programming environment.	**R1**
	CLO2	**Target2**：Master basic Linux file operations and be able to write simple file processing programs.	**R2**
	CLO3	**Target3**：Understand basic concepts of threads, be able to write simple multi-threaded programs and implement basic synchronization.	**R3**
	CLO4	**Target4** ： Master basic Linux network programming and implement simple client/server communication programs.	**R4**

Content	This course mainly introduces the core knowledge of Linux system programming, file operations, thread programming and network programming. It enables students to master the use of Linux development environment, basic program writing, thread creation and synchronization, and Socket network communication implementation. Students will learn to develop and debug simple programs on the Linux platform, cultivate practical and problem-solving abilities, and lay a solid foundation for subsequent embedded and server-side development. Knowledge Module 1: Linux Basics and Development Environment (Weight 2/32, Level: Memorization + Understanding) Knowledge Module 2: Linux Common Commands and Editor Usage (Weight 2/32, Level: Understanding + Application) Knowledge Module 3: Basics of Linux Compilation and Debugging Tools (Weight 2/32, Level: Understanding + Application) Knowledge Module 4: Basic Linux File I/O Programming (Weight 4/32, Level: Application + Analysis) Knowledge Module 5: Basic Concepts, Creation and Exit of Threads (Weight 4/32, Level: Application + Analysis) Knowledge Module 6: Basic Thread Synchronization Mechanisms (Weight 4/32, Level: Application + Analysis) Knowledge Module 7: Basic Linux Socket Network Programming (Weight 8/32, Level: Application + Analysis) Knowledge Module 8: Comprehensive Project Practice of Thread + Network (Weight 4/32, Level: Application + Analysis)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] The Linux Programming Interface, Michael Kerrisk, Posts & Telecom Press, 2023. [2] Linux System Programming, Robert Love, Posts & Telecom Press, 2022. [3] Advanced Programming in the UNIX Environment, W. Richard Stevens, Posts & Telecom Press, 2021. [4] Practical Linux C Programming, Tong Yongqing, Posts & Telecom Press, 2022. [5] Design and Implementation of a Multi-process Concurrent File Server Based on Linux . Computer Engineering and Design, 2024.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Comprehensive Training in Software Requirement Analysis

Module name	Comprehensive Training in Software Requirement Analysis		
Code	3080902518		
Semester	Spring, 4th Semester		
Contact person	Yang Weiqing,Li Yan		
Language	Chinese		
Relation to curriculum	Compulsory, Integrated Practical Course		
Type of teaching, contact hours	Lecture Method, Group Discussion Method,Case Analysis Method,Laboratory Experiment Method. Teaching Hours: 6 hours per week for 8 weeks, totaling 48 hours		
Work load	Total Workload (Estimated): 112 hours Teaching Hours: 48 hours Self-study Hours: 64 hours, including: After-class assignments, Project practice, etc.		
Credit points	4		
Recommended prerequisites	Basic Training in Software Front-end Development, Basic Training in Programming, Basic Training in Database Application		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1:** Proficiency in the basic theories and practical application technologies of each stage of requirements engineering (requirements elicitation, analysis, modeling, verification, management), and understanding the core role of requirements in software development.	**R2**
	CLO2	**Course Objective 2:** Master the knowledge and skills in business modeling, requirements elicitation, requirements definition, requirements analysis, specification, requirements verification and requirements management, and be able to standardly carry out the full-process practical operation of requirements engineering.	**R3**
	CLO3	**Course Objective 3:** Able to use requirements analysis tools and UML modeling tools to carry out requirements modeling, standardly write requirements specification documents, and use requirements analysis methods to improve the requirements control ability related to software development quality.	**R4**
	CLO4	**Course Objective 4:** Master the basic concepts and tasks of software engineering, integrate the relevant technical standards and specifications of the software engineering major, industrial policies, etc., standardly carry out practical operations, and improve team collaboration and communication skills.	**R8,R9**

Content	This course is a supporting core practical link of the "Software Requirements Analysis" course, closely connecting the theoretical knowledge of the course, taking real small-scale software projects as the carrier, focusing on the full-process practical operation of requirements engineering. By completing the complete practical tasks from requirements elicitation, requirements analysis, requirements modeling to requirements document writing, requirements verification and management in groups, it strengthens students' ability to apply the methods and tools of each stage of requirements engineering, improves students' abilities in requirements research, problem analysis, document writing and team collaboration, implements the core objectives of the course, helps students transform the theoretical knowledge of software requirements analysis into engineering practice ability, and lays a solid practical foundation for subsequent software testing, graduation design and related work in the workplace. Knowledge Module 1: Tool Cognition and Practical Training (Weight: 6/48, Level: Understanding + Application) Knowledge Module 2: Project Topic Selection and Requirements Elicitation (Weight: 10/48, Level: Understanding + Application) Knowledge Module 3: Requirements Analysis and Structured Modeling (Weight: 10/48, Level: Application + Analysis + Evaluation) Knowledge Module 4: Business Modeling and Use Case Modeling (Weight: 8/48, Level: Understanding + Application + Analysis) Knowledge Module 5: Requirements Document Writing and Quality Verification (Weight: 8/48, Level: Understanding + Application + Analysis) Knowledge Module 6: Achievement Arrangement, Display and Summary (Weight: 6/48, Level: Application + Analysis + Evaluation)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] 《Software Requirements Analysis》, Li Meirong, He Zhonghai, People's Posts and Telecommunications Press, 2024. [2] 《Practical Software Requirements Analysis》, Yang Changchun, Tsinghua University Press, 2020. [3] 《Methods and Practice of Software Requirements Engineering》, Jin Zhi, Liu Lin, et al., Tsinghua University Press, 2023. [4] 《Software Requirements Engineering《, Liang Zhengping, Wu Guoqing, et al., China Machine Press, 2020. [5]An Interpretable Requirements Analysis Framework for Machine Learning Applications [J]. Pei Zhongyi, Liu Lin, Wang Chen, et al. Journal of Computer Research and Development, 2024, 61(04): 983-1002.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Comprehensive Training in Object-Oriented Programming

Module name	Comprehensive Training in Object-Oriented Programming		
Code	3080902515		
Semester	Spring, 4th Semester		
Contact person	Sun Fang,,Li Yan		
Language	Chinese		
Relation to curriculum	Compulsory , Integrated Practical Course		
Type of teaching, contact hours	Lecture Method, Demonstration Method, Group Discussion Method, Laboratory Experiment and Project-Based Learning Teaching Hours: 4 hours per week for 16 weeks, totaling 64 hours		
Work load	Total estimated workload: 140 hours Contact hours: 64 hours Self-study hours: 76 hours (including project design, coding, testing, debugging, report writing, etc.)		
Credit points	5		
Recommended prerequisites	C Language Programming		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**CLO1：** Be able to use the core ideas of object-oriented programming and Java language knowledge, combined with common class libraries, exception handling, collection framework and other technologies, to solve medium-scale programming problems, and have solid ability in engineering knowledge application.	**R1**
	CLO2	**CLO2：** Be able to conduct object-oriented analysis and modeling for actual business requirements, identify key entities, attributes and behaviors in the system, design reasonable class hierarchy and interfaces, and have the ability to abstract and analyze complex problems.	**R2**
	CLO3	**CLO3：** Be able to independently complete the design and development of medium-scale Java applications according to requirements, implement a modular and extensible program architecture, conduct unit testing and debugging, and have system design and development capabilities.	**R3**
	CLO4	**CLO4：** Be able to skillfully use modern tools such as IDE (e.g. IntelliJ IDEA), debugging tools, version control system (Git) for project development, testing and maintenance to improve development efficiency and code quality.	**R5**

	CLO5	**CLO5：** Be able to cooperate with team members in project development, jointly promote development tasks, undertake the design and implementation of corresponding modules, and have a good sense of teamwork and personal responsibility.	**R8**
	CLO6	**CLO6：** Be able to write project documents and development reports clearly and standardly, and accurately express design ideas and implementation schemes in project defense, with good technical communication and achievement expression abilities.	**R9**
Content	This course is a follow-up comprehensive practical intensive course of Java Object-Oriented Programming. It takes project practice as the core, focuses on improving students' ability of object-oriented analysis, design and coding implementation, and emphasizes the combination of theory and engineering practice. The course focuses on object-oriented modeling, class design principles, interface-oriented programming, exception system, collection framework, I/O operation, comprehensive project development and standard document writing. Through phased intensive training and complete project practice, students can systematically master the whole process of software development, improve code specifications, modular design ability and problem-solving ability, and lay a solid practical foundation for subsequent advanced courses and enterprise project development. Knowledge Module 1: Review of Object-Oriented Core Knowledge and Modeling Training (Weight 8/64, Level: Application, Analysis) Knowledge Module 2: Class Structure Design, Encapsulation and Code Refactoring Training (Weight 12/64, Level: Application) Knowledge Module 3: Comprehensive Practice of Inheritance, Polymorphism and Interface (Weight 12/64, Level: Application, Analysis) Knowledge Module 4: Practical Training of Exception Handling, Collection Framework and I/O (Weight 12/64, Level: Application) Knowledge Module 5: Comprehensive Design of Medium-scale Java Object-Oriented Projects (Weight 12/64, Level: Creation) Knowledge Module 6: Project Testing, Debugging, Code Optimization and Training Report Writing (Weight 8/64, Level: Evaluation, Creation)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.		
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.		
Reading list	[7] Joshua Bloch. Effective Java[M]. 3rd Edition. Boston: Addison-Wesley, 2018. [8] Cay S. Horstmann. Core Java Volume I – Fundamentals[M]. 12th Edition. Beijing: China Machine Press, 2022. [9] Sun Qinqin. Java Object-Oriented Programming[M]. 2nd Edition. Beijing: Publishing House of Electronics Industry, 2019. [10] Kathy Sierra. Head First Java[M]. 3rd Edition. Beijing: China Electric Power Press, 2023. [11] Qin B. Comparative Teaching of C++ and Java in Object-Oriented Programming Courses[J]. Computer Education, 2024(04): 144-148+153. [12] Tao J. Experimental Research on Encapsulation of OOP in Java[J]. Information & Computer, 2024(02): 115-118.		

Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Comprehensive Training in Java Web Application Development

Module name	Comprehensive Training in Java Web Application Development		
Code	3080902516		
Semester	Fall ,5th Semester		
Contact person	Chai Yan, Zhu Xuesheng		
Language	Chinese		
Relation to curriculum	Compulsory, Integrated Practical Course		
Type of teaching, contact hours	Lecture Method, Demonstration Method, Group Discussion Method, Laboratory Experiment and Project-Based Learning Teaching Hours: 4 hours per week, 8 weeks in total, 32 hours in total		
Work load（Including teaching hours and self-study hours）	Total Workload (Estimated): 84 hours Teaching Hours: 32 hours Self-study Hours: 52 hours,(including project design, coding, testing, debugging, report writing, etc.)		
Credit points	3		
Recommended prerequisites	Object-Oriented Programming, Data Structures and Algorithms, Principles of Operating Systems, Software Engineering, Web Front-end Development Technology, Principles of Database Systems		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1**: Understand the basic principles and processes of Java Web application development; master at least two mainstream development frameworks (Spring MVC and MyBatis); have a comprehensive understanding of the Java Web technology stack; independently analyze and solve common problems in Java Web development.	**R2**
	CLO2	**Course Objective 2**: Proficiently use Java Web-related technologies to independently develop small-scale Web applications, including front-end page design, back-end logic processing, and database interaction; complete a full Java Web project.	**R3**
	CLO3	**Course Objective 3:** Skillfully use basic information technology and R&D tools in the Java Web development field, including but not limited to integrated development environments (e.g., IntelliJ IDEA, Eclipse), version control tools (e.g., Git), database management tools (e.g., MySQL Workbench, Navicat); master basic tool operations for coding, debugging, version management, and data interaction to lay a foundation for solving complex engineering problems.	**R5**

	CLO4	**Course Objective 4:**Possess professional ethics and teamwork spirit; demonstrate active communication and division of labor in project cooperation; reflect awareness of social responsibility and technical ethics in projects.	**R8、R9**
Content	This course aims to comprehensively improve students' Java Web development capabilities. Through a series of well-designed practical projects, students will experience the entire development process from requirement analysis, system design to coding implementation. The course covers core frameworks and technologies such as Servlet, JSP, Spring MVC, and MyBatis, emphasizing the integration of theory and practice, focusing on cultivating students' practical operation and problem-solving abilities. The expected outcome is that students can independently develop Java Web applications with reasonable structure and excellent performance, and establish a user-centered design concept, laying a solid foundation for future software development work. Knowledge Module 1: Building Java Web Development Environment (Weight: 2/32, Level: Application) Knowledge Module 2: Web Server Setup (Weight: 2/32, Level: Application) Knowledge Module 3: Basics of HTML & CSS Web Development (Weight: 2/32, Level: Understanding + Application + Analysis) Knowledge Module 4: JavaScript Scripting Language (Weight: 2/32, Level: Understanding + Application + Analysis) Knowledge Module 5: Basic JSP Syntax (Weight: 2/32, Level: Understanding + Application + Analysis) Knowledge Module 6: JSP Implicit Objects (Weight: 2/32, Level: Understanding + Application + Analysis) Knowledge Module 7: JavaBean Components (Weight: 2/32, Level: Understanding + Application + Analysis) Knowledge Module 8: Servlet Technology (Weight: 4/32, Level: Understanding + Application + Analysis) Knowledge Module 9:Database Development in Java Web (Weight: 4/32, Level: Understanding + Application + Analysis) Knowledge Module 10: Application of Ajax Technology (Weight: 2/32, Level: Understanding + Application + Analysis) Knowledge Module 11:Application of Struts 2 Framework (Weight: 4/32, Level: Understanding + Application + Analysis) Knowledge Module 12: Application of Hibernate Framework (Weight: 4/32, Level: Understanding + Application + Analysis)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.		
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.		

Reading list	[1] Qi Yan. Java Web Application Development[M]. Xidian University Press, 2024, 1st Edition. [2]Li Donghai. Lightweight Java EE Web Framework Technologies - Spring MVC + Spring + MyBatis + Spring Boot[M]. Tsinghua University Press, 2022, 1st Edition. [3]Chen Zhenxing. Java Web and Its Framework Technologies[M]. Publishing House of Electronics Industry, 2024. [4] Zhao Mingxuan. Development and Application of Structural Design Platform Based on Web[D]. Shenyang University of Technology, 2025. [5]Guo Wenxue. Design and Implementation of Rapid Development Tools for Web Applications[D]. Shandong University, 2022.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Software Engineering Practice

Module name	Software Engineering Practice		
Code	3080902511		
Semester	Fall, 5th Semester		
Contact person	Cao Xiuping, Wei Liangping		
Language	Chinese		
Relation to curriculum	Compulsory, Integrated Practical Course		
Type of teaching, contact hours	Lecture method, demonstration method, group discussion method, project-based practice Teaching duration: 4 hours per week for 12 weeks, totaling 48 hours		
Work load	Total workload (estimated): 112 credit hours Teaching duration: 48 hours Self-study hours: 64 hours, including homework and exam preparation time.		
Credit points	4		
Recommended prerequisites	C Language Programming, Data Structures and Algorithms, Principles of Operating Systems, Principles of Database Systems		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1：** Through the study of "Software Engineering Practice," students will master the fundamental processes of software engineering, core practical methods (requirement analysis, preliminary design, detailed design, coding, testing, and deployment), and project management fundamentals. By examining the current state of the software engineering industry both domestically and internationally, as well as the opportunities and challenges facing China's independent software industry, the course aims to foster students' professional responsibility and innovative mindset, strengthen their commitment to contributing to the development of China's independent software industry, and solidify their core engineering knowledge in software engineering.	**R1**
	CLO2	**Course Objective 2：** Through this course, students will master the core practical skills and tool usage methods across all stages of software engineering, understand the constraints and risk points in software project development, enable them to properly plan project workflows and design solutions based on actual project requirements, enhance their ability to analyze project issues and diagnose development faults, and cultivate systematic engineering thinking and problem analysis competencies.	**R2**

	CLO3	**Course Objective 3：** By participating in comprehensive software project development practices, students systematically and deeply integrate theoretical knowledge of software engineering with practical skills such as programming, testing, and design. They are capable of designing appropriate software architectures, developing core functional modules, and completing testing and deployment based on actual project requirements, thereby acquiring the ability to design and develop small to medium-sized software projects and laying a solid foundation for their future careers in software development.	**R3**
	CLO4	**Course Objective 4：** During the entire software project development process, students can communicate effectively, collaborate through clear division of labor, leverage each other's strengths, and fully utilize every member's professional expertise to jointly complete comprehensive software products (such as management systems or application mini-programs). This experience enhances their engineering practice skills and teamwork abilities, helps them establish sound software engineering ethics and professional standards, fosters innovative thinking, and develops their capacity to apply software engineering knowledge comprehensively to solve real-world engineering problems, thereby promoting overall personal growth.	**R7、R8**
Content	"Software Engineering Practice" is a core practical course for software engineering majors, designed to transform theoretical knowledge into engineering competencies. It provides students with comprehensive training in mastering the entire software project development lifecycle, core methodologies, and tool applications—including requirements analysis and modeling, software architecture design, detailed design, coding implementation, software testing, project management, and deployment. Through this course, students become proficient in using software engineering tools to complete project development processes systematically, develop the ability to design, develop, test, and manage small-to-medium software projects, and cultivate engineering thinking, collaboration skills, and innovation awareness, thereby laying a solid practical foundation for advanced coursework or careers in software development and project management. Knowledge Module 1: Overview of Software Engineering Practices and Tool Introduction (Weight: 4/48; Level: Memorization + Understanding + Application) Knowledge Module 2: Practice in Requirements Analysis and Modeling (Weight: 6/48; Level: Understanding + Application + Analysis) Knowledge Module 3: Software Design Practice (Overview Design + Detailed Design) (Weight: 10/48; Level: Understanding + Application + Evaluation) Knowledge Module 4: Code Implementation and Specifications (Weight: 12/48; Level: Application + Analysis + Creation) Knowledge Module 5: Software Testing Practice (Weight: 12/48; Level: Understanding + Application + Analysis) Knowledge Module 6: Project Management and Deployment (Weight: 4/48; Level: Memorization + Understanding + Application)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.		

Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1]Zhang Haifan, Mou Yongmin. Introduction to Software Engineering (8th Edition) [M]. Beijing: Tsinghua University Press, 2022 [2] Roger S. Pressman. Software Engineering: A Practitioner's Approach (9th Edition) [M], Beijing: Mechanical Industry Press, 2021 [3] Li Jianzhong, Wang Aiguo. Software Engineering Practice Tutorial (3rd Edition) [M], Beijing: Posts and Telecommunications Press, 2023 [4] Wang Yanqiu, Li Li, Zhang Min. Research on Reform and Innovation in Practical Teaching of Software Engineering[J], Computer Engineering and Science, 2024 [5] Chen Yang, Li Juan. Exploration of a Project-Driven Software Engineering Practice Teaching Model[J]. Computer Education, 2023, (08): 108-112. [6] Liu Min, Zhang Kai, Wang Li. Research and Practice on the Industry-Education Integration Model for Software Engineering Practice Courses[D]. Higher Education Research, 2024
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Comprehensive Training in System Analysis and Design

Module name	Comprehensive Training in System Analysis and Design		
Code	3080902519		
Semester	Spring, 6th Semester		
Contact person	Yang Weiqing, Fan deming		
Language	Chinese		
Relation to curriculum	Compulsory,Integrated Practical Course		
Type of teaching, contact hours	Lecture Method, Group Discussion Method,Case Analysis Method,Laboratory Experiment Method. Teaching Hours: 4 hours per week for 8 weeks, totaling 32 hours		
Work load	Total Workload (Estimated): 84 hours Teaching Hours: 32 hours Self-study Hours: 52 hours, including: After-class assignments, Project practice, etc.		
Credit points	3		
Recommended prerequisites	System Analysis and Design, Comprehensive Training in Object-Oriented Programming, Software Engineering Practice, Basic Training in Data Structures and Algorithms, Basic Training in Database Application		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1:** Proficiency in the basic operations of UML modeling tools such as Rose, able to independently use the tools to draw various UML diagrams, and implement the application requirements of UML modeling tools in the course theory.	**R5**
	CLO2	**Course Objective 2:** Able to conduct systematic requirement analysis on small-scale software systems, identify user needs and system functions, sort out requirements using tools such as UML use case diagrams, and cultivate the ability to analyze and decompose problems.	**R2**
	CLO3	**Course Objective 3:** Able to use UML for modeling static structures (class diagrams, object diagrams) and dynamic behaviors (sequence diagrams, state diagrams, etc.), design software system architectures and functional modules that meet requirements, and optimize the schemes.	**R3**
	CLO4	**Course Objective 4:** Able to standardly write system analysis and design documents, effectively communicate modeling ideas and design schemes in teams, and improve document writing and team collaboration capabilities.	**R9**

Content	This course is a supporting core practical link of the "System Analysis and Design" course, closely connecting the theoretical knowledge of the course, taking real software projects as the carrier, focusing on the application of UML modeling tools and the full-process practical operation of object-oriented system analysis and design. By completing the complete practical tasks from requirement analysis, system modeling to scheme design in groups, it strengthens students' ability to draw and apply various UML diagrams (use case diagrams, class diagrams, sequence diagrams, etc.), improves students' engineering practice ability to analyze complex problems and design feasible solutions, cultivates team collaboration and communication skills, and lays a solid practical foundation for subsequent software project development, graduation thesis design and workplace work. Knowledge Module 1: Training Preparation and Tool Operation (Weight: 4/32, Level: Memory+Understanding) Knowledge Module 2: Requirement Analysis and Use Case Modeling (Weight: 8/32, Level: Understanding+Application) Knowledge Module 3: System Modeling (Weight: 12/32, Level: Application + Analysis + Evaluation) Knowledge Module 4: System Design and Achievement Summary (Weight: 8/32, Level: Application + Analysis + Evaluation)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] 《UML Object-Oriented Analysis, Modeling and Design》, Lü Yunxiang, Zhao Tianyu, Tsinghua University Press, 2021. [2]《UML Software Modeling Task-Driven Tutorial》, Chen Chenghuan, People's Posts and Telecommunications Press, 2022. [3] 《UML Basics, Modeling and Application》, Cao Hanhua, Yi Yang, Guan Chunxi, et al., China Machine Press, 2022. [4] 《UML Modeling Example Tutorial (3rd Edition)》, Feng Xiangke, Liu Zhicheng, Electronic Industry Press, 2024. [5] A UML-Based Modeling Analysis and Design of Common Business for College Students [J]. Sun Xingfu, Yan Mengru, Cao Jiachi, et al., China Science and Technology Information, 2024, (15): 65-68.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Comprehensive Training in Software Testing

Module name	Comprehensive Training in Software Testing		
Code	3080902520		
Semester	Spring, 6th Semester		
Contact person	Zong Chengjie、Zhou Ying		
Language	Chinese		
Relation to curriculum	Compulsory, Integrated Practical Course		
Type of teaching, contact hours	Lecture Method、Demonstration Method、Group Discussion Method、Case Analysis Method Teaching Hours: 4 hours per week for 8 weeks, totaling 32 hours.		
Work load	Total Workload (Estimated): 84 credit hours Teaching Hours: 32 hours Self-study Hours: 52 hours, including after-class assignments, exam preparation and other self-directed learning activities.		
Credit points	3		
Recommended prerequisites	《Data Structures and Algorithms》《Software Engineering》《Principles of Database Systems》		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	Course Objective 1:Understand the complete software testing process, learn to implement and execute software testing according to software testing documents, improve software testing capabilities, cultivate teamwork spirit, gradually accumulate software testing experience, lay a solid foundation for students to engage in software testing, and achieve seamless connection with the software tester position.	**R3**
	CLO2	Course Objective 2:Understand and master the basic ideas and scientific system of software product quality assurance, software testing processes and strategies, as well as software testing methods, techniques and tool usage, so as to lay a solid foundation for fully mastering software technologies and software project management.	**R5**
	CLO3	Course Objective 3:Cultivate practical abilities in software testing, adapt to the working environment and industry standards of software enterprises, and keep pace with advanced international concepts of software development and cutting-edge testing technologies.	**R6**

	CLO4	Course Objective 4:Possess sound physical and psychological qualities, understand relevant laws and regulations, grasp the professional ethics and norms of software engineering including honesty, impartiality and integrity, and consciously abide by them in software engineering practice.	**R8**
Content	This course is a compulsory comprehensive training course for undergraduate students. It mainly covers fundamentals of software testing, black-box testing methods, white-box testing methods, interface testing, performance testing, security testing, automated testing, project practice, etc. The course adopts a teaching method that combines theory with practice. Through practical links such as building a testing environment, writing test cases, executing test cases, and submitting defect reports, it helps students quickly master skills. Through systematic study and practice, it aims to help students fully master the core skills and knowledge of software testing. Practical Project Module 1: Fundamentals of Software Testing (Weight: 2/32, Level: Comprehension) Practical Project Module 2: Software Testing Models, Process and Test Management (Weight: 4/32, Level: Comprehension + Memorization) Practical Project Module 3: Black-box Testing and White-box Testing (Weight: 4/32, Level: Comprehension + Application + Analysis) Practical Project Module 4: Unit Testing, Integration Testing and System Testing (Weight: 6/32, Level: Application + Analysis + Creation) Practical Project Module 5: Object-oriented Software Testing and Automated Testing (Weight: 8/32, Level: Application + Analysis + Creation) Practical Project Module 6: Web System Testing Cases (Weight: 8/32, Level: Application + Analysis + Creation)		
Forms of examination	he overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.		
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.T		
Reading list	[1] WEI N D, LI W B. Training of Software Testing Technology and Case Design[J]. Tsinghua University Press, 2022. [2] SONG X J, LIU S Y, SU M Y, et al. Research and practice of FPGA black-box testing method based on code review[J]. Application of Electronic Technology, 2026. [3] RU B S, CHEN L, ZHU S M. Authoritative Guide to Modern Software Testing Technology[M]. Beijing: Publishing House of Electronics Industry, 2025: 856. [4] ZHU Y J, FU Y Z, FANG X D. Project-based Tutorial of Software Testing[M]. Beijing: China Railway Publishing House, 2024: 197. [5] WANG H G, XIE B X, WANG Z F. Software Engineering Theory and Practice[M]. Xi'an: Northwest University Press, 2024: 262.		
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.		

Comprehensive Training in Enterprise-level Java Application Development

Module name	Comprehensive Training in Enterprise-level Java Application Development		
Code	3080902517		
Semester	Spring, 6th semester		
Contact person	Wang Xin, Qiao Qun		
Language	Chinese		
Relation to curriculum	Compulsory, Integrated Practical Course		
Type of teaching, contact hours	Lecture-based teaching、Group discussion method、Laboratory experiment method **Instructional hours:** 4 hours per week for 12 weeks, totaling 48 hours.		
Work load	**Total workload (estimated):** 112 hours **Instructional hours:** 48 hours **Self-study hours:** 64 hours, including after-class assignments, exam preparation time, etc.		
Credit points	4		
Recommended prerequisites	Object-Oriented Programming, Data Structures and Algorithms, Computer Networks, Software Engineering, Software Architecture, Principles of Database Systems		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1:** Design enterprise-level applications with high cohesion and low coupling based on mainstream frameworks such as Spring Boot. Follow layered architecture and design patterns to implement reusable, scalable, secure, and stable core business functions.	**R3**
	CLO2	**Course Objective 2:** Use IntelliJ IDEA as the core tool to proficiently integrate Maven, Git, Docker, and CI/CD tools. Leverage functions such as code generation, debugging and analysis, and containerized deployment to improve development and delivery efficiency.	**R5**
	CLO3	**Course Objective 3:** Clarify personal responsibilities in team collaboration and proactively undertake development tasks. Foster a sense of accountability and team spirit through code reviews and collaborative communication to jointly achieve project goals.	**R8**
	CLO4	**Course Objective 4:** Apply agile processes to manage project iterations, reasonably break down tasks and control progress. Standardize deliverable documentation to ensure project quality.	**R10**

Content	This course aims to build a bridge for students to transition into enterprise-level Java development. It adheres to the principles of "constructing a systematic Java knowledge framework, deeply analyzing the principles of SSM and Spring Boot frameworks, and strengthening enterprise-level project practice," thereby laying a solid foundation and providing clear direction for students to cultivate deep expertise in the field of Java development. The course systematically teaches core knowledge including Java Servlet, Spring, Spring MVC, MyBatis, Spring Boot core concepts, Spring Boot Web development, Spring Boot data access, Spring Boot security control, asynchronous messaging, deployment and testing, application monitoring, as well as Vue.js 3 fundamentals and advanced topics. Through the phased completion of three comprehensive case studies — based on SSM+JSP, Spring Boot+Thymeleaf+MyBatis, and Spring Boot+Vue.js 3+MyBatis — students will enhance their ability to flexibly build development frameworks and develop systems for different industry scenarios. **Knowledge Module 1:** Servlet and Spring Core (Weight: 8/48, Level: Comprehension + Application) **Knowledge Module 2:** SSM Framework Integration (Weight: 8/48, Level: Comprehension + Application + Analysis) **Knowledge Module 3:** Spring Boot Core (Weight: 8/48, Level: Memory + Comprehension + Application) **Knowledge Module 4:** Spring Boot Advanced (Weight: 8/48, Level: Memory + Comprehension + Application + Creation) **Knowledge Module 5:** Vue.js 3 Frontend (Weight: 8/48, Level: Comprehension + Application) **Knowledge Module 6:** Comprehensive Practical Case Studies (Weight: 8/48, Level: Memory + Comprehension + Creation)
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1]LI Donghai, JIN Zongxin. Lightweight Java EE Web Framework Technologies[M]. Beijing: Tsinghua University Press, 2022. [2] Ishu Anand Jaiswal. (2025). Integrating ai into enterprise java applications for secure high performance and scalable systems. *International Journal of Computational and Experimental Science and Engineering*, 11(4).
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Software Engineering Project Management Practice

Module name	Software Engineering Project Management Practice		
Code	3080902522		
Semester	Fall, 7th Semester		
Contact person	Sun Yingying, Liu Xing,Sun Ge		
Language	Chinese		
Relation to curriculum	Compulsory, Integrated Practical Course		
Type of teaching, contact hours	Lecture Method、Demonstration Method、Group Discussion Method、Case Analysis Method Teaching hours: 2 hours per week, for 16 weeks, totaling 32 hours .		
Work load	Total workload (estimated): 84 hours Teaching hours: 32 hours Self-study hours: 52 hours, including homework, exam preparation, etc.		
Credit points	3		
Recommended prerequisites	Software Engineering, Software Architecture, System Analysis and Design		
Learning outcomes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Through this course, students will master the knowledge of software project lifecycle management and the principles of agile development. They will acquire abilities in project planning, task decomposition, progress control, risk management, and documentation writing, and will be proficient in using project management tools for practical application. The course aims to cultivate standardized engineering awareness and teamwork competencies, equipping students with the fundamental skills required for implementing, managing, and delivering software projects.	Supported PLOs
	CLO1	**Course Objective 1:** Be able to apply mathematics, natural sciences, and software engineering expertise to solve complex engineering problems in the field of software engineering.	**R1**
	CLO2	**Course Objective 2:** Be able to design solutions for complex engineering problems in the field of software engineering, and develop software systems, modules, or algorithmic processes that meet specific requirements. Students should also demonstrate innovative thinking during the design phase, while considering social, health, safety, legal, cultural, and environmental factors.	**R3**
	CLO3	**Course Objective 3:** Possess a spirit of teamwork, and be able to fulfill the roles of an individual, team member, or leader within a multidisciplinary team to accomplish assigned tasks.	**R8**

	CLO4	**Course Objective 4:** Master software engineering project management methods, understand the important economic and managerial factors involved in software engineering activities, and be able to apply them in a multidisciplinary environment.	**R10**
Content	"Software Engineering Project Management Practice" is a core professional practical course in the software engineering program. Guided by the principle of "theory as the foundation, practice as the core, and tools for application," it systematically covers the entire lifecycle management process of software projects. The course integrates classical management theories with industry-proven methods, aiming to cultivate students' comprehensive capabilities in planning, executing, monitoring, and delivering software projects. It serves as a bridge between academic learning and real-world engineering practice in the industry. Knowledge Module 1: Overview of Software Project Management (Weight: 2/32, Level: Remember + Understand) Knowledge Module 2: Project Lifecycle and Management Processes (Weight: 4/32, Level: Understand + Apply) Knowledge Module 3: Project Initiation and Scope Management (Weight: 6/32, Level: Understand - Apply) Knowledge Module 4: Project Schedule and Cost Management (Weight: 6/32, Level: Apply - Analyze) Knowledge Module 5: Project Quality Management and Risk Control (Weight: 6/32, Level: Apply - Evaluate) Knowledge Module 6: Project Communication, Documentation, and Stakeholder Management (Weight: 4/32, Level: Understand + Apply) Knowledge Module 7: Agile Project Management Practice (Weight: 4/32, Level: Apply - Create)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.		
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.		
Reading list	[1] Application Exploration of the "Short Video Resources + BOPPPS" Blended Teaching Model in the "Software Project Management" Course Under the OBE Concept, Ren Chunhua, Wang Changyuan, Zeng Anping, Western Quality Education, 2025. [2] Evaluating Success Factors of Software Project Management in Global Software Development, IEEE Access, 2024. [3] Software Project Management and Practice, edited by Song Yingying, Tsinghua University Press, 2025. [4] Software Project Management (3rd Edition, Micro-Course Version), edited by Zhu Shaomin, Posts and Telecommunications Press, 2024. [5] Case Studies in Software Project Management (5th Edition), edited by Han Wanjiang, Mechanical Industry Press, 2024.		

Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Comprehensive Training in System Development

Module name	Comprehensive Training in System Development		
Code	3080902510		
Semester	Fall, 7th Semester		
Contact person	Chen Linong,Zhao Xia		
Language	Chinese		
Relation to curriculum	Compulsory，Integrated Practical Course		
Type of teaching, contact hours	This course adopts an integrated implementation of four categories of teaching methods: teacher-centered methods, interactive methods, individualized methods, and practical methods. Teacher-centered methods include: lectures, case teaching, questioning; Interactive methods include: group discussions, case seminars, phase reviews, code reviews, daily stand-up meetings, defense evaluations, project retrospectives; Individualized methods include: independent design/document writing, self-study tutoring, personalized assignments; Practical methods include: exercises/training, field research, pair programming, laboratory learning, role-playing, achievement presentation/demonstration. Teaching Hours: 3 weeks of concentrated practice, 48 hours in total.		
Work load	Total Workload (estimated): 112 hours. Teaching Hours: 48 hours. Self-Study Hours: 64 hours, including: technical research and self-directed learning, project document writing and refinement, presentation material preparation, etc.		
Credit points	4		
Recommended prerequisites	Data Structures and Algorithms, Principles of Database Systems, Software Engineering, System Analysis and Design, Object-Oriented Programming, Java Web Application Development, Enterprise-level Java Development		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes	Description	Supported PLOs
	CLO1	Be able to apply the basic methods of software engineering and object-oriented techniques to elicit, analyze, and produce standardized descriptions of the requirements for a given software project, and accurately depict the system's functional requirements, static structure, and dynamic behavior using UML models.	**R1**
	CLO2	Be able to identify and determine key issues and technical difficulties in the system development process for specific business scenarios, and to justify and determine reasonable technical routes and solutions through technical research, literature review, and other means.	**R2**
	CLO3	Be able to complete the system architecture design, module partitioning, database design, and detailed design based on the results of requirements analysis, produce standardized design documents, and conduct scheme comparison and optimization by comprehensively considering non-functional requirements such as technical feasibility and scalability.	**R3**

	CLO4	Be able to organize a team for coding implementation and testing based on the design scheme, analyze and evaluate the development process and product quality, and write standardized project technical documentation and summary reports.	**R4**
	CLO5	Be able to proficiently use modern software engineering tools such as UML modeling tools, IDE development tools, database design tools, version control tools (Git), and project management tools, understand the applicable scenarios and limitations of these tools, and communicate effectively, divide work reasonably, and collaboratively complete development tasks within a team.	**R5**
Content	System Development Comprehensive Training is an extended training course within the concentrated practice segment of the Software Engineering major. It adopts a project-driven teaching model and is conducted intensively over a continuous period of 3 weeks. The course simulates the entire process of enterprise software project development through small-group collaboration. Students are required to comprehensively apply knowledge acquired in prerequisite courses—such as data structures and algorithms, database design and optimization, software engineering methodology, system analysis and modeling, object-oriented programming, Java Web development, and enterprise-level framework technologies—to complete the entire process of requirements elicitation, system analysis, system design, coding implementation, testing and deployment, and achievement presentation for a small-scale software system under a given topic. Through the concentrated training of this course, students will further consolidate their theoretical knowledge of each stage of the software engineering life cycle, master the practical application of UML modeling tools, IDE development environments, database design tools, and version control and project management tools, effectively enhance their teamwork, communication and expression, self-directed learning, and ability to solve complex engineering problems, and lay a solid foundation for their graduation project and future career development. This course includes the following knowledge modules: Knowledge Module 1: Project Initiation and Team Formation (Weight 4/48, Level: Remembering + Understanding) Knowledge Module 2: Requirements Elicitation and Analysis (Weight 8/48, Level: Understanding + Applying + Analyzing) Knowledge Module 3: System Analysis and Modeling (Weight 10/48, Level: Applying + Analyzing + Creating) Knowledge Module 4: System Design (Weight 10/48, Level: Applying + Analyzing + Evaluating + Creating) Knowledge Module 5: Coding Implementation and Unit Testing (Weight 10/48, Level: Applying + Creating) Knowledge Module 6: System Testing and Integration (Weight 8/48, Level: Applying + Evaluating) Knowledge Module 7: Achievement Presentation and Defense (Weight 8/48, Level: Synthesizing + Evaluating)		
Forms of examination	The overall grade for this course consists of continuous assessment and the final examination grade. The grade is evaluated on a percentage scale, with 60 being the passing score for the course. The final examination takes the form of a comprehensive project design and oral defense, accounting for 50% of the overall grade. Students are required to submit their project design and present it in a defense session. The continuous assessment component accounts for the remaining 50% of the overall grade and is evaluated comprehensively based on students' task completion and the quality of their work.		
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.		

Reading list	Recommended Textbooks: [1] Practical Software Engineering (4th Edition), Lu Huien et al., China Machine Press, 2022. [2] Software Engineering: A Practitioner's Approach (9th Edition), Roger S. Pressman, translated by Zheng Renjie et al., China Machine Press, 2021. [3] UML 2 Foundations, Modeling and Design Practice, Li Bo et al., Tsinghua University Press, 2020. [4] Systems Analysis and Design (9th Edition), Kenneth E. Kendall, Julie E. Kendall, translated by Shi Ping'an et al., China Machine Press, 2020. [5] Agile Software Development: Principles, Patterns, and Practices, Robert C. Martin, translated by Deng Hui, Tsinghua University Press, 2003. [6] Refactoring: Improving the Design of Existing Code (2nd Edition), Martin Fowler, translated by Xiong Jie et al., Posts and Telecom Press, 2019. Recommended Reading Papers: [7] Guiding Principles for Assessing Software Engineering Teams[C]. Proceedings of the 2024 IEEE Frontiers in Education Conference (FIE), 2024. [8] Attribution of Work in Programming Teams with Git Reporter[C]. Proceedings of the 55th ACM Technical Symposium on Computer Science Education (SIGCSE TS 2024), 2024.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Graduation Internship

Module name	Graduation Internship		
Code	3080903522		
Semester	Spring ,8th Semester		
Contact person	Wang Fengqin,Li Yan		
Language	Chinese		
Relation to curriculum	Compulsory，Integrated Practical Course		
Type of teaching, contact hours	Teaching method：Field Trip Method、Project-Based Learning Method、Online Learning Method Teaching hours: totaling 32 hours .		
Work load	Total workload (estimated): 180 hours Teaching hours: 32 hours Self-study hours: 148 hours, including: technical research and self-directed learning, project document writing and refinement, presentation material preparation, etc.		
Credit points	6		
Recommended prerequisites	Labor Practice、Basic Training in Programming、Basic Training in Data Structures and Algorithms、Basic Training in Software Front-end Development、Basic Training in Database Application、Linux System Programming Training、Comprehensive Training in Software Requirement Analysis 、Comprehensive Training in Object-Oriented Programming 、Comprehensive Training in Java Web Application Development 、Software Engineering Practice、Comprehensive Training in System Analysis and Design、Comprehensive Training in Software Testing、Comprehensive Training in Enterprise-level Java Application Development、Software Engineering Project Management Practice、Comprehensive Training in System Development		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	In the graduation internship course, professional theoretical knowledge was deeply integrated with job practice, systematically mastering the core business processes and operational standards of the industry.	Supported PLOs
	CLO1	Course Objective 1: This graduation internship course aims to enable students to leverage software engineering knowledge in complex project practices, analyze and evaluate the impacts of development, operations, and maintenance phases on health and safety, ecological environment, compliance, and socio-economic sustainable development, foster a sense of responsibility, and grow into software engineering professionals equipped with both technical competence and social accountability.	**R6**

	CLO2	Course Objective 2: This graduation internship course aims to strengthen students' humanities and social sciences literacy and sense of social responsibility. Through practical experiences in real-world software engineering scenarios, students are guided to understand and uphold engineering ethical principles, proficiently master industry professional ethics standards and relevant laws and regulations, establish compliance awareness, and fulfill professional responsibilities throughout the entire project development and collaboration process, growing into software engineering professionals equipped with both technical competence and moral integrity.	**R7**
	CLO3	Course Objective 3: This graduation internship course aims to cultivate students' teamwork spirit, enabling them to practice switching between different roles, clearly understand the responsibility boundaries of individual contributors, core team members, and project leaders, and master communication and collaboration methods to ensure efficient task completion, laying a solid foundation for future workplace collaboration.	**R8**
	CLO4	Course Objective 4: This graduation internship course aims to enable students to master the skills of writing technical reports, designing solutions, and delivering oral presentations for complex software engineering problems in real-world project settings. Students will learn to precisely address the needs of industry peers and respond to public inquiries. Through cross-cultural collaboration tasks, they will broaden their international perspective and enhance their adaptability in cross-cultural communication, laying a solid foundation for diverse workplace interactions.	**R9**
Content	This course is a vital component of talent development and a key step in deepening classroom learning, serving as an important pathway for students to understand society, engage with real-world production practices, and acquire on-site professional knowledge. It plays a significant role in cultivating practical abilities, fostering innovation, and shaping a strong sense of responsibility and career commitment. Knowledge Module 1: Consolidation of Professional Theoretical Knowledge (Weight: 8/32, Level: Understanding) Knowledge Module 2: On-the-Job Practical Skill Development (Weight: 16/32, Level: Application) Knowledge Module 3: Professional Competence and Comprehensive Abilities (Weight: 2/32, Level: Understanding) Knowledge Module 4: Industry Awareness and Career Development (Weight: 2/32, Level: Understanding) Knowledge Module 5: Problem Analysis and Reflective Improvement (Weight: 4/32, Level: Evaluation)		
Forms of examination	The total score for the graduation internship consists of formative assessment and summative assessment, both evaluated on a percentage scale. The formative assessment accounts for 50% of the total score, comprehensively evaluating the student's task completion, work quality, internship logs, weekly summaries, monthly summaries, attendance and discipline during the internship, as well as the combined evaluations from the enterprise mentor and the on-campus supervisor. The summative assessment accounts for the remaining 50%, taking the form of a comprehensive internship report and an on-site oral defense, which comprehensively examines the student's mastery of job-specific skills, ability to apply theoretical knowledge to solve practical problems, and level of professional competence. The final internship grade is classified into four tiers: Excellent (90–100), Good (80–89), Pass (60–79), and Fail (0–59). Students who receive a Fail must retake the internship with the following cohort before they can proceed to the graduation process.		

Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Practical Software Engineering Tutorial, authored by Sang Haitao, Wang Xiaoye, and Hou Rui, 1st edition, Tsinghua University Press, 2021. [2] Practical Software Project Management, edited by Wang Qiang, 2nd edition, Tsinghua University Press, 2024. [3] Software Engineering Ethics and Professional Literacy, edited by Liu Jing, 1st edition, Higher Education Press, 2023. [4] Chang Ying. Diversified Teaching Method Practice for Applied Undergraduate Software Engineering Courses [J]. Heilongjiang Science, 2019, 10(09): 48-49. [5] Lan Gang, Gan Tao, Guan Qing, Fu Chong. Exploration and Reform of Process Management in Comprehensive Design Course for Software Engineering Specialty [J]. Education and Teaching Forum, 2020(18): 196-199.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.

Graduation Thesis (Design)

Module name	Graduation Thesis (Design)		
Code	3080903601		
Semester	Spring, 8th Semester		
Contact person	Wang Fengqin, Guo Ning		
Language	Chinese		
Relation to curriculum	Compulsory, Integrated Practical Course		
Type of teaching, contact hours	Lecture Method, Case Analysis Method, Project-Based Learning Method Teaching Hours: 4 hours per week for 16 weeks, totaling 64 hours		
Work load	Total Workload (estimated): 360 hours. Teaching Hours: 64 hours. Self-study Hours: 296 hours, including: technical research and self-directed learning, project document writing and refinement, presentation material preparation, etc.		
Credit points	12		
Recommended prerequisites	Introduction to Software Engineering Program, C Language Programming, Data Structures and Algorithms, Principles of Database Systems, Software Engineering, Web Front-end Development Technology, Software Architecture, System Analysis and Design, Software Testing, Software Project Management, Enterprise-level Java Development, Android Mobile Application Development		
Learning out-comes and their corresponding PLOs	Course Learning Outcomes (CLOs)	Description	Supported PLOs
	CLO1	**Course Objective 1:** Cultivate students' ability to use software engineering related principles to accurately identify key links of complex engineering problems in graduation thesis (design), skillfully use flowcharts, use case diagrams and other software engineering tools to describe problems, sort out problem roots through literature research, requirement analysis and other methods, compare the rationality of multiple solutions, form scientific and effective analysis conclusions, and implement the core requirements of problem identification and analysis in graduation project.	**R2**
	CLO2	**Course Objective 2:** Guide students to design solutions for complex software engineering problems that meet actual needs based on specific requirements of graduation project, complete system architecture, algorithm, component and database design, integrate innovative consciousness, fully consider constraints such as society, health, safety, law, etc., and have the ability to transform design schemes into implementable software systems or modules, meeting the functional and non-functional design requirements of graduation project.	**R3**

CLO3	**Course Objective 3:** Guide students to determine research routes for graduation project through literature research based on software engineering theory, design feasible experimental schemes, use software modeling, algorithm analysis and other methods to build related software units, standardize collection and organization of experimental data, scientifically analyze and interpret data and synthesize information, form reasonable and effective research conclusions, improve scientific research practice ability, and achieve the research-oriented training purpose of graduation project.	**R4**
CLO4	**Course Objective 4:** Meet the practical needs of graduation project, enable students to skillfully use literature retrieval tools to complete graduation project data collection, reasonably select and use mainstream programming languages, database management systems, software design and testing tools, apply them to analysis, design, implementation and verification of graduation project, understand application scenarios and limitations of various tools, improve comprehensive tool application ability, and ensure smooth implementation of graduation project results.	**R5**
CLO5	**Course Objective 5:** Combined with the engineering practice attributes of graduation project, guide students to become familiar with technical standards, intellectual property rights, laws and regulations related to software engineering during graduation project, analyze the impact of graduation project engineering practice on society, health, safety and environment, evaluate the sustainability of solutions, establish environmental protection and sustainable development concepts, clarify their responsibilities in engineering practice, and meet the requirements of engineering ethics and sustainable development for graduation project.	**R6**
CLO6	**Course Objective 6:** Improve students' communication ability, enable them to clearly express views on complex engineering problems in graduation project through graduation project reports, design documents, charts and other forms, effectively communicate with supervisors, team members and peers, have the ability to write standardized graduation project theses and engineering technical documents, cultivate international vision, and improve basic cross-cultural communication ability.	**R9**
CLO7	**Course Objective 7:** Cultivate students' awareness of autonomous learning and lifelong learning, guide students to pay attention to development trends and research hotspots in the field of software engineering during graduation project, learn to track frontier knowledge of the discipline, improve the ability to identify the latest research directions, cultivate the ability of rational analysis and problem posing, lay the foundation for continuous adaptation to industry development after graduation, and achieve the long-term training purpose of graduation project.	**R11**

Content	Graduation Thesis (Design) is an advanced development course for the undergraduate stage of Software Engineering major, and a comprehensive examination of students' professional knowledge, skills and literacy learned over four years. This course focuses on cultivating students' ability to solve complex engineering problems in the field of software engineering. Combined with professional graduation requirements, it guides students to implement the cultivation of research ability, practical ability, teamwork and professional literacy through the whole process training of literature research, problem analysis, scheme design, practice development, thesis writing, etc., helps students meet professional graduation requirements, and lays a solid foundation for entering the workplace or continuing further study. Knowledge Module 1: Overview of Graduation Thesis (Design) (Weight 4/64, Level: Remember, Understand) Knowledge Module 2: Literature Retrieval and Research Methods for Graduation Project (Weight 6/64, Level: Remember, Apply, Analyze) Knowledge Module 3: Identification and Analysis of Complex Software Engineering Problems (Weight 12/64, Level: Understand - Analyze) Knowledge Module 4: Graduation Project Scheme Design and Development (Weight 12/64, Level: Apply - Create) Knowledge Module 5: Application of Modern Software Engineering Tools (Weight 12/64, Level: Apply - Evaluate) Knowledge Module 6: Graduation Project Thesis Writing and Standards (Weight 10/64, Level: Remember, Apply, Analyze) Knowledge Module 7: Graduation Project Management and Teamwork (Weight 4/64, Level: Understand, Apply, Analyze) Knowledge Module 8: Graduation Project Ethics Standards and Sustainable Development (Weight 4/64, Level: Remember, Understand, Evaluate)
Forms of examination	The graduation thesis (design) assessment implements full-process quality monitoring, covering stages such as proposal, mid-term inspection, final acceptance, plagiarism check, and defense review. The final grade is composed of three parts: the supervisor (30%), the reviewer (10%), and the defense committee (60%). The final grade is evaluated on a five-tier scale: Excellent (90–100), Good (80–89), Medium (70–79), Pass (60–69), and Fail (below 60). The supervisor assesses the student's task completion, work quality, independence, work attitude, and adherence to deadlines throughout the thesis writing process. The reviewer evaluates the academic rigor, structural completeness, and clarity of expression of the final written thesis. Only those with a similarity ratio below 30% are eligible to proceed to the defense. During the defense, students are required to submit their thesis or design project and respond to questions posed by the defense committee. The committee makes a comprehensive assessment of the defense outcome based on the quality of the written work, the student's presentation, and their ability to answer questions.
Study and examination requirements	Following the examination regulations, students must have selected and registered for the courses.
Reading list	[1] Software Engineering Case Tutorial (Micro-video Version), edited by Hu Xiaopeng, Chen Guiyou, Tsinghua University Press, 2024. [2] Introduction to Software Engineering, edited by Zhang Haifan, Tsinghua University Press, 2022, 6th Edition. [3] Practical Software Engineering, edited by Zheng Renjie, Tsinghua University Press, 2021, 5th Edition. [4] Software Engineering: A Practitioner's Approach, edited by Ian Sommerville, Mechanical Industry Press, 2019, 10th Edition. [5] A Large-Scale Survey on the Usability of AI Programming Assistants: Successes and Challenges，ICSE '24，2024.
Revision Date	V2024.1, September 2024. V2024.2, December 2025, Update: Credits and workload are calculated based on ECTS.