# Content

**Note: The undergraduate program of Computer Science and Technology (English Taught) requires 140 credits in 8 semesters for a student to obtain a bachelor diploma. The ratio of ECTS with JNU credits is approximately 240/140 = 1.7 (ECTS : JNU credits = 1.7)**

| Course Title | Advanced Mathematics I | | | | |
|---|---|---|---|---|---|
| Credits | 3 | Hrs./Week | 3 | Total Hrs. | 54 |
| Course Code | 60080001 | | Course Type | Compulsory | |
| Prerequisite Course(s) | N/A | | Course Evaluation | Closed-book Exam | |

Teaching Objectives & Requirements

Advanced Mathematics is the most important basis subject for the majors of science and technology at university. The subject is concentrated on the basic concepts, basic theories and basic methods by the limiting method. By the study of the subject, the student must master the following contents: be skilled in the function, be skilled in finding limits, be skilled in judging the continuity of a function, be skilled in finding derivatives and finding derivatives of implicit functions, understand the relation between the monotone and the first derivative of a function, understand the relation between the convexity and the second derivative of a function, be able to sketch the graph of a function, be able to do approximate calculation, be able to distinguish the difference and connection between definite integral and indefinite integral, be able to evaluate integral, be able to evaluate the area bounded by curves. The ability of students to analyze problem, to solve problem will be enhanced. Advance mathematics is necessary for the following courses.

Teaching Contents:

The main contents of Advanced Mathematics I are limits and continuity, derivatives, differentiation rules and applications.

Chapter 1: Preliminary (2 weeks)

Chapter 2: Limits and Continuity (4 weeks)

Chapter 3: Differentiation (4 weeks)

Chapter 4: Applications of Derivatives (3 weeks)

| Text Book(s) and Reference Materials | Calculus, James Stewart. Brooks/Cole Pub Co, 5th edition |
|---|---|

| Course Title | Introduction to Computer Science | | | | |
|---|---|---|---|---|---|
| Credits | 3 | Hrs./Week | 3 | Total Hrs. | 54 |
| Course Code | 60080002 | | Course Type | Compulsory | |
| Prerequisite Course(s) | N/A | | Course Evaluation | Closed-book Exam | |

Teaching Objectives:

This course aims to provide a breadth-first coverage of the discipline of computer science, and to give student essential information about computer science so that they can pursue the subject further. This course covers computing history and future, number systems and digital data representation. It introduces curriculum, education and career opportunities of computer science majors. It explains computing hardware, software, operating systems and networks, with essential concepts and technologies. It also explains preliminary programming and algorithms, with a series of coding practices. It offers a gentle introduction to subareas in computer science such as software engineering, multimedia, database, artificial intelligence, and so on. It also discusses security technologies and social issues of computing.

Teaching Requirements:

Students will be able to：

- Understand how the courses in computer science are related.
- Intelligently understand the history of computer technology.
- Master how to convert between binary and decimal numbers, and master digital data representation.
- Describe the sequence of computer hardware switching technologies and discuss their advantages over previous hardware along with any inherent weaknesses.
- Master the usage of popular application software, describe the functionalities of operation systems, and differentiate popular operation systems from each other.
- Master basic concepts and technologies of LANs and WANs.
- Describe the history, basic concepts and technologies of the Internet
- Describe the history, basic concepts and technologies of the Web.
- Understand the relationship among algorithm, problem solving and programming.
- Master the basic ideas of programming abstraction, logic and bugs
- Describe the computer language translation processes and list the sequence of computer language generations and analyze their differences.
- Master the basic concepts of software engineering and list the software development life cycle.
- Master the basic concepts of multimedia.
- Master how structured data works and the basic concepts of database
- Describe common security attacks, the best way to defend against them, and how viruses, worms and other malware attack computer systems, and discuss other security issues.
- Understand the history and basic concepts of artificial intelligence.
- Discuss social issues related to computing technology

Teaching Contents:

Part I. Groundwork

1. Layers of a Computing System
2. Computer history
   History of Hardware, History of Software
3. Computer science as a discipline
   Subareas of computer science, Curriculum of CST majors，Course orientation，
4. Computer science as a Career
   Jobs and salaries, Education and certification

Part II. Digital data representation

1. Number system
2. Digital Data representation

Part III. Computer System

1. Hardware
   Gates and circuits (Optional), Stored-program concepts, von Neumann Architecture
   Computer components: CPU, Memory, Storage devices, peripheral devices
2. Software
   Software basics, popular applications, security software
3. Operating System
   Operating system basics, today's OS, File basics

Part IV. Network

1. LAN
   Network components, wired networks, wireless networks, LAN, encryption
2. The Internet
   Internet technology, Internet access, Internet services, Internet security
3. The Web and Email
   Web technology, search engine, email, security

Part V. Multimedia

1. Digital images
2. Digital sounds and video

Part VI. Programming, Algorithm and Software Engineering

1. Programming
   Programming language, abstraction, logic, loops, bugs
2. Algorithm
   Problem solving and algorithm design

| | |
|---|---|
| 3. Software Engineering<br>Software development life cycle, planning, coding, testing and documentation<br><br>Part VII. Database<br>1. How structured data works<br>2. Database concepts, DBMS<br><br>Part VIII. Artificial Intelligence -Optional<br>Thinking machines, expert systems, neural networks | |
| Text Book(s) and Reference Materials | Textbook:<br>1. New Perspectives on Computer Concepts， Comprehensive (China Student Edition), J.J. Parsons, Dan Oja, 15th Edition, 2013. China Machine Press, ISBN: 9787111428039<br>2. Computer Science Illuminated， Nell Dale, John Lewis，3rd Edition, 2008. China Machine Press, ISBN：9787111235163<br><br>References:<br>1. Eloquent JavaScript: A Modern Introduction to Programming, Marijn Haverbeke, No Starch Press, 2 edition, 2014, ISBN-13: 978-1593275846<br>2. Computer Science: An Overview，J.Glenn Brookshear， 11th Edition, 2012. People's Post Press，ISBN：9787115277947<br>3. Computers Are Your Future, Complete, 11th Edition. Laberta, Catherine. Prentice Hall, Inc., 2010. |

| Course Title | Problem Solving and Programming | | | | |
|---|---|---|---|---|---|
| Credits | 3 | Hrs./Week | 3 | Total Hrs. | 54 |
| Course Code | 60080003 | | Course Type | Compulsory | |
| Prerequisite Course(s) | N/A | | Course Evaluation | Closed-book Exam | |

Teaching Objectives:

"Problem Solving & Programming" aims to equip the students with basic concepts and techniques of problem solving and programming using a high-level programming language; and to develop practical skills in solving problems with computer programs.

Teaching Requirements:

After successfully completing this course, you will understand how to write computer programs using ANSI C language. And you will be able to

a) use common program development tools
b) analyze problems and apply basic problem-solving skills/techniques to construct programs with good structures
c) practice good programming discipline
d) apply advanced programming techniques in programming

Teaching Contents:

**Chapter 1 Introduction to Computer Programming**

- history and hardware
- programming languages
- algorithms
- the software development process
- case study: design and development
- common programming errors

**Chapter 2 Getting Started in C Programming**

- introduction to c programming
- programming style
- data types
- arithmetic operations
- variables and declarations
- case study: temperature conversion

**Chapter 3 Processing and Interactive Input**

- assignment
- mathematical library functions
- interactive input

- formatted output
- symbolic constants
- case study: interactive input

**Chapter 4 Selection**

- relational expressions
- the if and if-else statements
- the if-else chain
- the switch statement
- case study: data validation

**Chapter 5 Repetition**

- basic loop structures
- the while statement
- computing sums and averages using a while loop
- the for statement
- case studies: loop programming techniques
- nested loops
- the do-while statement

**Chapter 6 Modularity Using Functions: Part I**

- function and parameter declarations
- returning a value
- case study: calculating age norms
- standard library functions

**Chapter 7 Modularity Using Functions: Part II**

- variable scope
- variable storage class
- pass by reference
- case study: swapping values
- recursion 4

**Chapter 8 Arrays**

- one-dimensional arrays
- array initialization
- arrays as function arguments
- case study: computing averages and standard deviations
- two-dimensional arrays

**Chapter 9 Character Strings**

- string fundamentals
- library functions
- input data validation
- formatting strings
- case study: character and word counting

**Chapter 10 Data Files**

- declaring, opening, and closing file streams
- reading from and writing to text files
- random file access
- passing and returning filenames
- case study: creating and using a table of constants
- writing and reading binary files

**Chapter 11 Arrays, Addresses, and Pointers**

- array names as pointers
- manipulating pointers
- passing and using array addresses
- processing strings using pointers
- creating strings using pointers

**Chapter 12 Structures**

- single structures
- arrays of structures
- passing and returning structures
- unions

**Chapter 13 Dynamic Data Structures**

- introduction to linked lists
- dynamic memory allocation
- stacks
- queues
- dynamically linked lists

**Chapter 14 Additional Capabilities**

- additional features
- bit operations
- macros
- command-line arguments

| | |
|---|---|
| Text Book(s) and Reference Materials | A First Book of ANSI C,4th ed., by Gary J. Bronson, Thomson Course Technology, 2007. |

| Course Title | Problem Solving & Programming Lab | | | | | |
|---|---|---|---|---|---|---|
| Credits | 1 | Hrs./Week | 2 | Total Hrs. 36 | | |
| Course Code | 60080004 | | Course Type | | Compulsory | |
| Prerequisite Course(s) | N/A | | Course Evaluation | | Open-book | |

Teaching Objectives:

"Problem Solving & Programming Experiment" aims to equip the students with basic concepts and techniques of problem solving and programming using a high-level programming language via practical labs in class; and to develop practical skills in solving problems with computer programs.

Teaching Requirements:

After successfully completing this course, you will understand how to write computer programs using ANSI C language. And you will be able to:

a) use common program development tools
b) analyze problems and apply basic problem-solving skills/techniques to construct programs with good structures
c) practice good programming discipline
d) apply advanced programming techniques in programming

Teaching Contents:

Lab#1: C Programming Using Visual C++
Lab#2: Data types and Variables
Lab#3: Interactive Input and Formatted Output
Lab#4: Selection
Lab#5: Loops
Lab#6: Functions
Lab#7: Pointers
Lab#8: Arrays
Lab#9: Recursion
Lab#10: Strings
Lab#12: Arrays & Pointers
Lab#13: Data Files
Lab#14: Structures

| Text Book(s) and Reference Materials | A First Book of ANSI C,4th ed., by Gary J. Bronson, Thomson Course Technology, 2007. |
|---|---|

<table>
<tr><td>Course Title</td><td colspan="5">Advanced Mathematics II</td></tr>
<tr><td>Credits</td><td>3</td><td>Hrs./Week</td><td>3</td><td>Total Hrs.</td><td>54</td></tr>
<tr><td>Course Code</td><td colspan="2">60080005</td><td colspan="2">Course Type</td><td>Compulsory</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">Advanced Mathematics I</td><td colspan="2">Course Evaluation</td><td>Closed-book Exam</td></tr>
<tr><td colspan="6">Teaching Objectives & Requirements:<br>Advanced Mathematics is the most important basis subject for the majors of science and technology at university. The subject is concentrated on the basic concepts, basic theories and basic methods by the limiting method. By the study of the subject, the student must master the following contents: be skilled in the function, be skilled in finding limits, be skilled in judging the continuity of a function, be skilled in finding derivatives and finding derivatives of implicit functions, understand the relation between the monotone and the first derivative of a function, understand the relation between the convexity and the second derivative of a function, be able to sketch the graph of a function, be able to do approximate calculation, be able to distinguish the difference and connection between definite integral and indefinite integral, be able to evaluate integral, be able to evaluate the area bounded by curves. The ability of students to analyze problem, to solve problem will be enhanced. Advance mathematics is necessary for the following courses.<br><br>Teaching Contents:<br>The main contents of Advanced Mathematics II are integration, infinite series and multivariable calculus.<br>Chapter 5: Integrals<br>Chapter 6: Applications of Integration<br>Chapter 7: Techniques of Integration<br>Chapter 9: Differential Equations<br>Chapter 10: Parametric Equations and Polar Coordinates<br>Chapter 11: Infinite Sequences and Series<br>Chapter 14: Partial Derivatives<br>Chapter 15: Multiple Integrals</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">Calculus, James Stewart. Brooks/Cole Pub Co, 5th edition</td></tr>
</table>

<table>
<tr><td>Course Title</td><td colspan="5">Linear Algebra</td></tr>
<tr><td>Credits</td><td>3</td><td>Hrs./Week</td><td>3</td><td>Total Hrs.</td><td>54</td></tr>
<tr><td>Course Code</td><td colspan="2">60080006</td><td>Course Type</td><td colspan="2">Compulsory</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">N/A</td><td>Course Evaluation</td><td colspan="2">Closed-book Exam</td></tr>
<tr><td colspan="6">Teaching Objectives & Requirements:<br>Linear Algebra is a basic compulsory course of computer science and technique, its major content consists of matrix, determinants, vector space, Eigenvalues and Orthogonality.<br>The first goal of the course is to teach students how to use linear algebra as a powerful tool for computation. The second goal is to give a gentle introduction to the theory of matrix, determinant, abstract vector spaces, Eigenvalues and Orthogonality.<br><br>Teaching Contents:<br>The main contents of Linear Algebra are:<br>* Linear Equations and their solutions;<br>* The Algebra of Matrices;<br>* Determinants;<br>* Vector spaces;<br>* Eigenvalues and Eigenvectors of real matrices;<br>* Orthogonality.</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">Linear algebra with applications, 8th edition，Steven j. Leon</td></tr>
</table>

| Course Title | Discrete Mathematics I | | | | |
|---|---|---|---|---|---|
| Credits | 3 | Hrs./Week | 3 | Total Hrs. | 54 |
| Course Code | 60080007 | | Course Type | Compulsory | |
| Prerequisite Course(s) | N/A | | Course Evaluation | Closed-book Exam | |

Teaching Objectives:

"Discrete mathematics" is the theoretical foundation of computer science and technology. The teaching objectives focus on two aspects:

1) It lays the necessary mathematical foundations for the subsequent courses, such as data structure, compiler principle, operating system, database theory, and artificial intelligence;

2) This course helps students to develop and improve skills of abstract thinking and logical reasoning and improve the ability to analyze and solve problems independently.

Teaching Requirements:

After successfully completing this course, students will be able to:

1. Understand fundamental concepts, basic terminology, and basic theorems of logic, sets, algorithms, number theory and cryptography, induction and recursion, and counting.

2. Develop and gain practical skills of abstract thinking and logical reasoning and use the knowledge to analyze and solve practical problems.

Teaching Contents:

**Chapter 1 The foundations: logic and proofs**

- propositional logic
- applications of propositional logic
- propositional equivalences
- predicates and quantifiers
- nested quantifiers
- rules of inference
- introduction to proofs
- proof methods and strategy

**Chapter 2 Basic structures: sets, functions, sequences, sums, and matrices**

- sets
- set operations
- functions
- sequences and summations
- cardinality of sets
- matrices

**Chapter 3 Algorithms**

- algorithms
- the growth of functions
- complexity of algorithms

**Chapter 4 Number theory and cryptography**

- divisibility and modular arithmetic
- integer representations and algorithms
- primes and greatest common divisors
- solving congruences
- applications of congruences
- cryptography

**Chapter 5 Induction and recursion**

- mathematical induction
- strong induction and well-ordering
- recursive definitions and structural induction
- recursive algorithms
- program correctness

**Chapter 6 Counting**

- the basics of counting
- the pigeonhole principle
- permutations and combinations
- binomial coefficients and identities
- generalized permutations and combinations
- generating permutations and combinations

| | |
|---|---|
| Text Book(s) and Reference Materials | Discrete Mathematics and Its Applications Seventh Edition, Kenneth H. Rosen, McGraw-Hill Science/Engineering/Math, ISBN:9787111385509 |

| Course Title | Discrete Mathematics II | | | | |
|---|---|---|---|---|---|
| Credits | 3 | Hrs./Week | 3 | Total Hrs. | 54 |
| Course Code | 60080010 | | Course Type | Compulsory | |
| Prerequisite Course(s) | N/A | | Course Evaluation | Closed-book Exam | |

Teaching Objectives:

"Discrete mathematics" is the theoretical foundation of computer science and technology. The teaching objectives focus on two aspects:

1) It lays the necessary mathematical foundations for the subsequent courses, such as data structure, compiler principle, operating system, database theory, and artificial intelligence;

2) This course helps students to develop and improve skills of abstract thinking and logical reasoning and improve the ability to analyze and solve problems independently.

Teaching Requirements:

After successfully completing this course, students will be able to:

1. Understand fundamental concepts, basic terminology, and basic theorems of logic, sets, algorithms, number theory and cryptography, induction and recursion, and counting.

2. Develop and gain practical skills of abstract thinking and logical reasoning and use the knowledge to analyze and solve practical problems.

Teaching Contents:

**Chapter 8 Advanced Counting Techniques**

- Applications of Recurrence Relations
- Solving Linear Recurrence Relations
- Divide-and-Conquer Algorithms and Recurrence Relations
- Generating Functions
- Inclusion–Exclusion
- Applications of Inclusion–Exclusion

**Chapter 9 Relations**

- Relations and Their Properties
- n-ary Relations and Their Applications
- Representing Relations
- Closures of Relations
- Equivalence Relations
- Partial Orderings

**Chapter 10 Graphs**

- Graphs and Graph Models
- Graph Terminology and Special Types of Graphs
- Representing Graphs and Graph Isomorphism
- Connectivity
- Euler and Hamilton Paths
- Shortest-Path Problems
- Planar Graphs
- Graph Coloring

**Chapter 11 Trees**

- Introduction to Trees
- Applications of Trees
- Tree Traversal
- Spanning Trees
- Minimum Spanning Trees

**Chapter 12 Boolean Algebra**

- Boolean Functions
- Representing Boolean Functions
- Logic Gates
- Minimization of Circuits

**Chapter 13 Modeling Computation**

- Languages and Grammars
- Finite-State Machines with Output
- Finite-State Machines with No Output
- Language Recognition
- Turing Machines

| | |
|---|---|
| Text Book(s) and Reference Materials | Discrete Mathematics and Its Applications Seventh Edition, Kenneth H. Rosen, McGraw-Hill Science/Engineering/Math, ISBN:9787111385509 |

| Course Title | Data Structures | | | | |
|---|---|---|---|---|---|
| Credits | 3.5 | Hrs./Week | 4 | Total Hrs. | 64 |
| Course Code | 60080011 | | Course Type | Compulsory | |
| Prerequisite Course(s) | C Programming<br>Discrete Mathematics | | Course Evaluation | Closed-book Exam | |

Teaching Objectives:

This course covers some of the general-purpose data structures and algorithms. It is aimed at providing skills on how data may be structured, and instructions sequenced in algorithms and programs as well as the relationship between appropriate data and control structures. Topics covered consist of the following three parts: Part One introduces the concept of abstract data type (ADT) and the basic methods for time-space complexity analysis. In Part Two, fundamental data structures for stack, queue, list, tree and graph are discussed, together with their implementations. Part Three talks about implementations and analysis of sorting and searching. Lab exercises reinforce the lectures.

Teaching Requirements:

At the completion of the course, students are required to be able to

1. Describe the properties, interfaces and behavior of general used data structures
2. Analyze the correctness and computational complexity of computer algorithms
3. Design and implement general-purpose, reusable data structures

Teaching Contents:

**1. Introduction & Recursion (3 hours)**

**2. Algorithm Analysis (3 hours)**

**3. ADT, List, Stack, Queue (8 hours)**

• Linear List Concept
• Sequential Lists
• Linked Lists
• Linked Lists operations
• Stacks: LIFO structure, create, POP, PUSH, delete stack
• Applications of stack
• Queues: FIFO structure Priority Queues, Circular Queues, operations on Queues

**4. Introduction to Trees (12 hours)**

• Preliminaries
• Binary Trees
• Threaded binary trees

• Search Trees
• AVL Trees
• Multi-way search trees
• B Trees

**5. HEAPS (Priority Queues) (4 hours)**
• Binary Heap
• Applications of Priority Queues
• *d*-Heaps
• Leftist Heaps*
• Skew Heaps*

**6. GRAPH algorithms (12 hours)**
• Operations (Add vertex, Delete Vertex, Add Edge, Delete Edge, Find Vertex)
• Traverse Graph (Depth-First, Breadth-First)
• Graph Storage Structures (Adjacency Matrix, Adjacency List)
• Topological sort
• Networks

**7. SORTING (8 hours)**
Bubble, Selection, Insertion, Shell sorts, Heap sort, Merge sort, Quick sort, Radix sort, External sorting.

**8. Hashing Techniques (4 hours)**
•Hash function
•Address calculation techniques, Common hashing functions
•Collision resolution
•Linear probing, Quadratic
•Double hashing
•Bucket hashing
•Deletion and rehashing

| | |
|---|---|
| Text Book(s) and Reference Materials | Data structures and Algorithm Analysis in C, Mark Allen Weiss, Posts & Telecom Press |

| Course Title | Data Structures Lab | | | | |
|---|---|---|---|---|---|
| Credits | 0.5 | Hrs./Week | 1 | Total Hrs. | 18 |
| Course Code | 60080012 | | Course Type | Compulsory | |
| Prerequisite Course(s) | C Programming<br>Discrete Mathematics | | Course Evaluation | Experiment-based | |

Teaching Objectives:

A collection of values that share a common set of operations is called a data type. Structured or composite data types are collections of individual data items of the same or different data types. Data structures are collections of variables, possibly of different data types, connected in various ways. An abstract data type is a programming language facility for organizing programs into modules using criteria that are based on the data structures of the program. Also, an abstract data type can be defines as a set of values and a set of procedures that manipulate those values. The specification of the module should provide all information required for using the type, including the allowable values of the data and the effects of the operations. However, details about the implementation, such as data representations and algorithms for implementing the operations, are hidden within the module. This separation of specification from implementation is a key idea of abstract data types. This laboratory course is intended to facilitate understanding of the widely used data structures such as lists, trees, graphs, hash tables and the operations associated with them as well as some sorting algorithms. The language used for implementations is C.

Teaching Requirements:

After this course, a student should be able to:

1. familiar with the basic methods of computer processing data.
2. know how to evaluate the performance of an algorithm.
3. choose appropriate logical structure, storage structure and algorithm for a specific application
4. improve the programming skill by realize the algorithms introduced in the theory class

Teaching Contents:

**1. Lists (3 hours)**

Objectives: The purpose of this lab session is to practice and to define the abstract data type of both sequential and linked lists. And, show how to implement operations on them.

**2. Calculator (3 hours)**

Objectives: In this comprehensive experiment students will practice the usage of stack.

**3. Binary Tree (3 hours)**

Objectives: In this lab session, students will work on basic operations for binary trees.

**4. Graph (3 hours)**

<table>
<tr><td colspan="2">Objectives: In this lab session students will implement some graph representation methods and graph traversals.<br><br>5. Graph Processing Algorithms (3 hours)<br>Objectives: In this lab session we will implement some of the graph processing algorithms, i.e. Dijkstra’s and Floyd’s for minimum cost paths, and Kruskal’s and Prim’s for minimum spanning trees.<br><br>6. Fundamental Sorting Algorithms (3 hours).<br>Objectives: In this lab session we will experiment with a number of sorting algorithms. Specifically: insertion sort (direct insertion and shell sort) and quicksort.</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td>Data structures and Algorithm Analysis in C, Mark Allen Weiss, Posts & Telecom Press</td></tr>
</table>

<table>
<tr><td>Course Title</td><td colspan="5">Computer Organization</td></tr>
<tr><td>Credits</td><td>3</td><td>Hrs./Week</td><td>3</td><td>Total Hrs.</td><td>54</td></tr>
<tr><td>Course Code</td><td colspan="2">60080013</td><td>Course Type</td><td colspan="2">Compulsory</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">C Programming Language</td><td>Course Evaluation</td><td colspan="2">Closed-book Exam</td></tr>
<tr><td colspan="6">Teaching Objectives:<br>Course's aim is to help students become a better programmer by teaching students the basic concepts underlying all computer systems. This course wants students to learn what really happens when their programs run, so that when things go wrong (as they always do) students will have the intellectual tools to solve the problem.<br><br>Teaching Requirements:<br>By the end of the course, students will be prepared to take any of the upper-level systems classes. Even more important, students will have learned skills and knowledge that will help students throughout their career.<br><br>Teaching Contents:<br>Chapter 1 A Tour of Computer Systems (3 hours)<br>Chapter 2 Representing and Manipulating Information (3 hours)<br>Chapter 3 Machine-Level Representation of Programs (3 hours)<br>Chapter 4 Processor Architecture (6 hours)<br>Chapter 5 Optimizing Program Performance (6 hours)<br>Chapter 6: The Memory Hierarchy (6 hours)<br>Chapter 7 Linking (6 hours)<br>Chapter 8 Exceptional Control Flow (3 hours)<br>Chapter 9 Virtual Memory (6 hours)<br>Chapter 10 System-Level I/O (3 hours)<br>Chapter 11 Network Programming (3 hours)<br>Chapter 12 Concurrent Programming (6 hours)</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">Computer Systems: A Programmer's Perspective, Randal E. Bryant and David R. O'Hallaron, Third Edition, Prentice Hall, 2017.</td></tr>
</table>

| Course Title | Computer Organization Lab | | | | |
|---|---|---|---|---|---|
| Credits | 1 | Hrs./Week | 2 | Total Hrs. | 36 |
| Course Code | 60080014 | | Course Type | Compulsory | |
| Prerequisite Course(s) | C Programming Language | | Course Evaluation | Experiment-based | |

Teaching Objectives:

Experiment of Computer Organization is a 1-credit laboratory course for the Computer Organization. Course's aim is to help students become a better programmer by teaching students the basic concepts underlying all computer systems. This course wants students to learn what really happens when their programs run, so that when things go wrong (as they always do) students will have the intellectual tools to solve the problem.

Teaching Requirements:

By the end of the course, students will be prepared to take any of the upper-level systems classes. Even more important, students will have learned skills and knowledge that will help students throughout their career.

Teaching Contents:

**Lab 1 Data Lab**

This lab requires students to implement simple logical and arithmetic functions, but using a highly restricted subset of C. For example, they must compute the absolute value of a number using only bit-level operations. This lab helps students understand the bit-level representations of C data types and the bit-level behavior of the operations on data.

**Lab 2 Architecture Lab**

Several of the homework problems of Chapter 4 can be combined into a lab assignment, where students modify the HCL description of a processor to add new instructions, change the branch prediction policy, or add or remove bypassing paths and register ports. The resulting processors can be simulated and run through automated tests that will detect most of the possible bugs. This lab lets students experience the exciting parts of processor design without requiring a complete background in logic design and hardware description languages.

**Lab 3 Digital Logic Gates**

All the integrated circuits (ICs) we will use are members of the Transistor-Transistor Logic (TTL) family and its improved variants. The circuits are a mixture of chips from the 74LS series (“Low-power Schottky,” introduced 1976) and 74HC series (“High-speed CMOS, TTL compatible,” introduced 1982), and are in the form of dual-inline packages (DIPs). This lab lets students experience the TTL digital logic circuits.

**Lab 4 Conversion between the Basic Logic Gates**

The purpose of this assignment is to study the right wiring way in TTL digital logic circuits and the conversion between the basic logic gates.

1. Use one NOR chip (74LS02) to construct circuits that perform NAND function.
2. Use one NAND chip (74LS00) to construct circuits that perform XOR function.

| | |
|---|---|
| Text Book(s) and Reference Materials | Computer Systems: A Programmer's Perspective, Randal E. Bryant and David R. O'Hallaron, Third Edition, Prentice Hall, 2017. |

<table>
<tr><td>Course Title</td><td colspan="5">Machine Learning</td></tr>
<tr><td>Credits</td><td>2</td><td>Hrs./Week</td><td>2</td><td>Total Hrs.</td><td>36</td></tr>
<tr><td>Course Code</td><td colspan="2">60080073</td><td>Course Type</td><td colspan="2">Specialized Optional</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">N/A</td><td>Course Evaluation</td><td colspan="2">Closed-book Exam</td></tr>
<tr><td colspan="6">Teaching Objectives:<br>This course provides a broad introduction to machine learning and statistical pattern recognition. Topics include: supervised learning; unsupervised learning; learning theory; reinforcement learning and adaptive control. The course will also discuss recent applications of machine learning, such as to robotic control, data mining, autonomous navigation, bioinformatics, speech recognition, and text and web data processing.<br><br>Teaching Requirements:<br>Students are expected to have the following background:<br>1. Knowledge of basic computer science principles and skills, at a level sufficient to write a reasonably non-trivial computer program.<br>2. Familiarity with the probability theory.<br><br>Teaching Contents:<br>• Introduction: basic concepts.<br>• Supervised learning: naive Bayes, support vector machines, feature selection, ensemble methods<br>• Learning theory: VC dimension, Bias<br>• Unsupervised learning: clustering, K-means, EM, PCA (Principal components analysis)<br>• Reinforcement learning and control: MDPs, POMDPs, Bellman equations, value iteration and policy iteration, Q-learning, value function approximation</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Trevor Hastie , World Book Inc, ISBN：9787510084508</td></tr>
</table>

| Course Title | Java Programming | | | | |
|---|---|---|---|---|---|
| Credits | 2 | Hrs./Week | 2 | Total Hrs. | 36 |
| Course Code | 60080008 | | Course Type | Compulsory | |
| Prerequisite Course(s) | Problem Solving & Programming | | Course Evaluation | Closed-book Exam | |

Teaching Objectives:

- To introduce concepts and principles of problem solving by computer, and the construction of appropriate algorithms for the solution of problems.
- To demonstrate the principles underlying the design of high-level programming languages.
- To give students experience and confidence in the use of a high-level programming language to implement algorithms.

Teaching Requirements:

General requirements:

- An understanding of the principles and practice of object-oriented analysis and design in the construction of robust, maintainable programs which satisfy their requirements;
- A competence to design, write, compile, test and execute straightforward programs using a high-level language;
- An appreciation of the principles of object-oriented programming;
- An awareness of the need for a professional approach to design and the importance of good documentation to the finished programs.

Specific requirements:

- Students will be able to create a Java program using a simple text editor. Students can compile programs through a command prompt window creating Java bytecode using Sun's JDK. Programs can be debugged using errors displayed in the command prompt window.
- Students will be able to effectively utilize variables, constants, if statements, switch statements, conditional expressions, loops and nested loops, logical operators, relational operators, arithmetic operators, arrays, and methods. Students will be able to validate input and format output.
- Students will be able to utilize predefined classes to create objects as necessary. Students will be able to define classes with properties, constructors, and methods and use those classes to instantiate objects.
- Students will be able to develop an algorithm to solve a given problem. Students can translate the algorithm into a working Java program using recommended programming style and techniques.

Teaching Contents:

*Part I: Fundamentals of Programming*

Chapter 1 Introduction to Computers, Programs, and Java 1

Chapter 2 Elementary Programming 23

Chapter 3 Selections 71

| | |
|---|---|
| Chapter 4 Loops 115<br>Chapter 6 Single-Dimensional Arrays 197<br>Chapter 7 Multidimensional Arrays 235<br><br>*Part II: Object-Oriented Programming*<br>Chapter 5 Methods 155<br>Chapter 8 Objects and Classes 263<br>Chapter 9 Strings and Text I/O 301<br>Chapter 10 Thinking in Objects 343<br>Chapter 11 Inheritance and Polymorphism 373<br>Chapter 13 Exception Handling 431<br>Chapter 18 Applets and Multimedia (Optional)<br>Chapter 29 Multithreading (Optional)<br><br>*Part III: GUI Programming*<br>Chapter 12 GUI Basics 405 | |
| Text Book(s) and Reference Materials | Introduction to Java Programming, Comprehensive 8/E, Liang, ISBN: 9780132130806 |

| Course Title | Java Programming Lab | | | | |
|---|---|---|---|---|---|
| Credits | 0.5 | Hrs./Week | 1 | Total Hrs. | 18 |
| Course Code | 60080009 | Course Type | Compulsory | | |
| Prerequisite Course(s) | Problem Solving & Programming | Course Evaluation | Lab Exercises | | |

Teaching Objectives & Requirements:

Upon successful completion of this course, students should be able to:

1. Construct a basic Java program using Sun's Java Development Toolkit (JDK).

Students will be able to create a Java program using a simple text editor. Students can compile programs through a command prompt window creating Java bytecode using Sun's JDK. Programs can be debugged using errors displayed in the command prompt window.

2. Use basic programming fundamentals such as variables, constants, selection statements, loops, methods and arrays.

Students will be able to effectively utilize variables, constants, if statements, switch statements, conditional expressions, loops and nested loops, logical operators, relational operators, arithmetic operators, arrays, and methods. Students will be able to validate input and format output.

3. Use basic object-oriented programming concepts.

Students will be able to utilize predefined classes to create objects as necessary. Students will be able to define classes with properties, constructors, and methods and use those classes to instantiate objects.

4. Develop an algorithm to solve a given problem and translate it into a working Java program.

Students will be able to develop an algorithm to solve a given problem. Students can translate the algorithm into a working Java program using recommended programming style and techniques.

Teaching Contents:

**Programming exercises** of the following chapters:

- Chapter 1 Introduction to Computers, Programs, and Java 1
- Chapter 2 Elementary Programming 23
- Chapter 3 Selections 71
- Chapter 4 Loops 115
- Chapter 5 Methods 155
- Chapter 6 Single-Dimensional Arrays 197
- Chapter 7 Multidimensional Arrays 235
- Chapter 8 Objects and Classes 263
- Chapter 9 Strings and Text I/O 301
- Chapter 10 Thinking in Objects 343
- Chapter 11 Inheritance and Polymorphism 373

| | |
|---|---|
| • Chapter 12 GUI Basics 405<br>• Chapter 13 Exception Handling 431 (Optional) | |
| Text Book(s) and Reference Materials | Introduction to Java Programming, Comprehensive 8/E, Liang, ISBN: 9780132130806 |

<table>
<tr><td>Course Title</td><td colspan="5">Operating Systems</td></tr>
<tr><td>Credits</td><td>3.5</td><td>Hrs./Week</td><td>3.5</td><td>Total Hrs.</td><td>72</td></tr>
<tr><td>Course Code</td><td colspan="2">60080015</td><td colspan="2">Course Type</td><td>Compulsory</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">Data structure, algorithm, computer organization, assembly programming, introduction to computer science, programming in C</td><td colspan="2">Course Evaluation</td><td>Closed-book Exam</td></tr>
<tr><td colspan="6">Teaching Objectives & Requirements:<br>This course aims to provide the students a comprehensive understanding of operating system and its role within a computing system.<br><br>Teaching Contents:<br>• Chapter 1 INTRODUCTION<br>• Chapter 2 PROCESSES AND THREADS<br>• Chapter 3 MEMORY MANAGEMENT<br>• Chapter 4 FILE SYSTEMS<br>• Chapter 5 INPUT/OUTPUT<br>• Chapter 6 DEADLOCKS<br>• Chapter 7 VIRTUALIZATION AND THE CLOUD<br>• Chapter 8 MULTIPLE PROCESSOR SYSTEMS<br>• Chapter 9 SECURITY<br>• Chapter 10 CASE STUDY 1: UNIX, LINUX, AND ANDROID<br>• Chapter 11 CASE STUDY 2: WINDOWS 8<br>• Chapter 12 OPERATING SYSTEM DESIGN</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">Textbook:<br>Andrew Tanenbaum and Herbert Bos, Modern Operating Systems, 4th ed,<br>References:<br>Abraham Silberschatz, Peter Galvin, Greg Gagne, Operating System Concepts, 9th or 10th ed.<br>Andrew Tanenbaum, Albert Woodhull, Operating Systems Design and Implementation, 3rd ed.<br>Nick Blundell, Writing a simple Operating System – from scratch.</td></tr>
</table>

<table>
<tr><td>Course Title</td><td colspan="5">Operating Systems Lab</td></tr>
<tr><td>Credits</td><td>0.5</td><td>Hrs./Week</td><td>1</td><td>Total Hrs.</td><td>18</td></tr>
<tr><td>Course Code</td><td colspan="2">60080016</td><td>Course Type</td><td colspan="2">Compulsory</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">Data structure, algorithm, computer organization, assembly programming, introduction to computer science, programming in C</td><td>Course Evaluation</td><td colspan="2">Lab work report and presentation</td></tr>
<tr><td colspan="6">Teaching Objectives & Requirements:<br>This course aims to provide the students a comprehensive understanding of how an operating system is implemented by reading the corresponding source codes which have been open-sourced.<br><br>Teaching Contents:<br>Three lab works are provided to help the students better understand how an operating system works, mainly by reading parts of the operating system codes which have been open-sourced. More specifically,<br>• Lab work A - give the students a taste of virtual machine, host/guest operating systems setup, and programming in such an environment.<br>• Lab work B - give the students a better understanding of how an operating system is designed and organized.<br>• Lab work C - give the students a better understanding of how a virtual machine is designed and organized.</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">Textbook:<br>Andrew Tanenbaum and Herbert Bos, Modern Operating Systems, 4th ed,<br>References:<br>Abraham Silberschatz, Peter Galvin, Greg Gagne, Operating System Concepts, 9th or 10th ed.<br>Andrew Tanenbaum, Albert Woodhull, Operating Systems Design and Implementation, 3rd ed.<br>Nick Blundell, Writing a simple Operating System – from scratch.</td></tr>
</table>

<table>
<tr><td>Course Title</td><td colspan="5">Software Engineering</td></tr>
<tr><td>Credits</td><td>3</td><td>Hrs./Week</td><td>3</td><td>Total Hrs.</td><td>54</td></tr>
<tr><td>Course Code</td><td colspan="2">60080017</td><td>Course Type</td><td colspan="2">Compulsory</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">OO Programming<br>Data Structures</td><td>Course Evaluation</td><td colspan="2">Closed-book Exam</td></tr>
<tr><td colspan="6">Teaching Objectives & Requirements:<br>• The key objective of this course is to learn modular design of software and documenting the design using symbolic representations, i.e., UML diagrams. The course will cover software life-cycle models and different phases of the software development process.<br>• Object-oriented techniques are key to the course. Since the ultimate result of software engineering is a working software package, the course will put a great emphasis on developing a demonstrable software package. However, this is not a programming course.<br>• The key characteristic is having three stages of project works: individual work, pair work, teams of three to four students work on developing complex software systems over a course of one semester.<br><br>Teaching Contents:<br>• Topic 1 (3hrs): Software Lifecycle<br>• Topic 2 (3hrs): Requirements Engineering and Use Cases<br>• Topic 3 (2hrs): Software Architecture<br>• Topic 4 (4hrs): Object-Oriented Analysis<br>• Topic 5 (4hrs): Object-Oriented Design<br>• Topic 6 (2hrs): Test-Driven Implementation<br>• Topic 7 (3hrs): System Specification<br>• Topic 8 (6hrs): Problem Frames & Software Measurement<br>• Topic 9 (6hrs): Design Patterns</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">Software Engineering, Ivan Marsic, http://eceweb1.rutgers.edu/~marsic/books/SE/</td></tr>
</table>

<table>
<tr><td>Course Title</td><td colspan="5">Design and Analysis of Algorithms</td></tr>
<tr><td>Credits</td><td>3</td><td>Hrs./Week</td><td>3</td><td>Total Hrs.</td><td>54</td></tr>
<tr><td>Course Code</td><td colspan="2">60080018</td><td>Course Type</td><td colspan="2">Compulsory</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">Computer programming</td><td>Course Evaluation</td><td colspan="2">Closed-book Exam</td></tr>
<tr><td colspan="6">Teaching Objectives:<br>To make students understand the importance of algorithms in the field of computers and master basic ideas and methods of the design and analysis of algorithms, so that they can better understand and write computer programs and enhance the ability of solving problems.<br><br>Teaching Requirements:<br>The course study should be organized in the way of lecturing as well as discussion by groups. And students need to attend class and finish homework on time.<br><br>Teaching Contents:<br>1. Prologue: introduce the performance and the asymptotic analysis of algorithms.<br>2. Algorithms with numbers: introduce the basic arithmetic algorithms and modular arithmetic and hashing.<br>3. Divide-and-conquer algorithms: introduce the recurrence relations and the applications of divide-and-conquer algorithms in merge sort and medians.<br>4. Graph algorithms: introduce the basic search algorithms on graphs and the application of graph algorithms in shortest path problems.<br>5. Greedy algorithms: introduce the ideas of greedy algorithms and their applications in the problems of minimum spanning trees and set cover.<br>6. Dynamic programming: introduce the ideas of dynamic programming and its application in knapsack and shortest path problems.<br>7. Linear programming and reductions: introduce to linear program and its application in flows in networks and bipartite matching.<br>8. NP-complete problems: introduce the features of NP-complete problems and their solving methods.</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">Textbook:<br>Algorithm, Sanjoy Dasgupta et al., China Machine Press, ISBN:9787111253617<br>References:<br>Introduction to Algorithms, Thomas H.Cormen et al., China Machine Press, ISBN：9787111407010</td></tr>
</table>

| Course Title | Probability Statistics | | | | |
|---|---|---|---|---|---|
| Credits | 3 | Hrs./Week | 3 | Total Hrs. | 54 |
| Course Code | 60080019 | | Course Type | Compulsory | |
| Prerequisite Course(s) | Calculus, Linear Algebra | | Course Evaluation | Closed-book Exam | |

Teaching Objectives & Requirements:

Probability Statistics is a mathematical discipline which studies stochastic phenomena. Now it is widely used in industrial and agricultural production, science and technologies. This course is one of the important basic courses for computer science majors in comprehensive universities, through which students shall know the general conceptions and methods about probability and statistics, master the basic definitions, theories and corresponding methods, master the methods to deal with random phenomenon by means of establishing the basic statistical models and master the necessary ability in English listening, speaking and so on. We stress theory and practice combined, in order to help students promote their ability of applying statistical methods in their daily life and scientific research.

Teaching Contents:

**Chapter 1 Overview and Descriptive Statistics**

1.1 Populations, Samples, and Processes

1.2 Pictorial and Tabular Methods in Descriptive Statistics

1.3 Measures of Location

1.4 Measures of Variability

**Chapter 2 Probability**

2.1 Sample Spaces and Events

2.2 Axioms, Interpretations and Properties of Probability

2.3 Counting Techniques2.4 Conditional probability

2.5 Independence

**Chapter 3 Discrete Random Variables and Probability Distributions**

3.1 Random Variables

3.2 Probability Distributions for Discrete Random Variables

3.3 Expected Values of Discrete Random Variables

3.4 The Binomial Probability Distribution

3.6 The Poisson Probability Distribution

**Chapter 4 Continuous Random Variables and Probability Distributions**

4.1 Continuous random variables and probability density functions

4.2 Cumulative Distribution Functions and Expected Values

4.3 The Normal Distribution

**Chapter 5 Joint Probability Distributions and Random Samples**

5.1 Jointly Distributed Random Variables

5.2 Expected Values, Covariance, and Correlation

5.3 Statistics and their Distributions

5.4 The Distribution of the Sample Mean

5.5 The Distribution of a Linear Combination

**Chapter 6 Point Estimation**

6.1 Some General Concepts of Point Estimation

6.2 methods of point estimation

**Chapter 7 Statistical Intervals Based on a Singles Sample**

7.1 Basic properties of confidence intervals

7.2 Large-sample confidence intervals for a population mean and proportion

7.3 Intervals based on a normal population distribution

7.4 Confidence intervals for the variance and standard deviation of a normal population

**Chapter 8 Testing of Hypotheses based on a single sample**

8.1 Hypotheses and test procedures

8.2 Tests about a population mean

8.3 Tests concerning a population proportion

8.4 P-values

8.5 Some comments on selecting a test procedure

| | |
|---|---|
| Text Book(s) and Reference Materials | Jay L.Devore, Probability and statistics for engineering and the sciences (sixth edition),China Machine Press. |

| Course Title | Database Systems | | | | |
|---|---|---|---|---|---|
| Credits | 3 | Hrs./Week | 3 | Total Hrs. | 54 |
| Course Code | 60080020 | | Course Type | Compulsory | |
| Prerequisite Course(s) | N/A | | Course Evaluation | Closed-book Exam | |

Teaching Objectives & Requirements:

The goal is to teach students the fundamental concepts underlying database system design, including not only the design of applications using databases, but also covering the fundamental implementation techniques used in database systems. The amount of material that needs to be covered will make such a course a rather intensive one to cover in one semester, and students must be prepared for the course load.

The course would be supplemented by assignments and a project. The assignments can involve the design of a schema for a realistic application, and coding and executing SQL queries on a relational database system. They are to get students familiar with using databases, running queries and using application development tools. The project would involve the design of a schema for a realistic application, and the implementation of the entire application using SQL and any language/APIs including but not limited to Java.

Teaching Contents:

**Chapter 1:** Introduction to Database Systems

**Chapter 2:** Introduction to the Relational Model

**Chapter 3:** Basics of SQL

**Chapter 4:** Intermediate SQL

**Chapter 5:** Advanced SQL

**Chapter 6:** Other Relational Languages (Relational Algebra, Tuple Relational Calculus, Domain Relational Calculus)

**Chapter 7:** Entity-Relationship Model

**Chapter 8:** Relational Database Design

**Chapter 9:** Application Design and Development

**Chapter 10:** Storage and File Structure

**Chapter 11:** Indexing and Hashing

**Chapter 12:** Query Processing

**Chapter 13:** Query Optimization

| | |
|---|---|
| **Chapter 14:** Transactions<br>**Chapter 17:** Database-System Architectures<br>**Chapter 20:** Data Warehousing and Mining<br>**Chapter 21:** Information Retrieval | |
| Text Book(s) and Reference Materials | Textbook:<br>Database System Concepts, 6ed, Abraham Silberschatz *et al.*, McGraw-Hill, ISBN: 9780073523323<br>Fundamentals of Database Systems, 7ed, Ramez Elmasri, Shamkant B. Navathe, Pearson 2015<br>References:<br>Data Model Patterns: Conversion of Thoughts, David C. Hay, Dorset House, 1996 |

| Course Title | Computer Networks | | | | |
|---|---|---|---|---|---|
| Credits | 3 | Hrs./Week | 3 | Total Hrs. | 54 |
| Course Code | 60080021 | | Course Type | Compulsory | |
| Prerequisite Course(s) | Operating System, Data Structure | | Course Evaluation | Closed-book Exam | |

Teaching Objectives:

This course introduces the basics of networking, ranging from sending bits over wires to networked applications such as the Web. The outcome of this course for students should be an appreciation of the fundamental principles and design challenges of computer networking, including the network layer models and related networking technologies.

Teaching Requirements:

This course will be organized by the OSI and TCP/IP layer model to offer students comprehensive knowledge on computer networks. Topics required for students include: 1) The fundamental architecture and design principles of computer networks; 2) Basic data communication, error detection and correction and packet switching, etc.; 3) Networking technologies in LAN; 4) Networking technologies in wireless networks; 5) Operations on network layer including addressing and forwarding (IP), routing, etc.; 6) Reliable transport in networks, congestion control (TCP); 7) Several typical Internet application protocols e.g., DNS, HTTP, WWW and EMAIL; and 8) Overview of secure network protocols.

Teaching Contents:

**Chapter 1 Introduction:**

As the introduction of computer networks, this section will explain the basic concepts and components of computer networks, including related hardware, software and the reference models (OSI and TCP/IP). The Internet, mobile phone networks, 802.11, and RFID and sensor networks will also be discussed as examples.

**Chapter 2 The Physical Layer**

The physical layer will discuss how to transmit data over physical medium, including: digital modulation, guided transmission media (e.g., twisted pairs, coaxial cable and power lines) and wireless transmission (e.g., radio, microwave and light), the public switched telephone networks and mobile telephone system (e.g., 3G networks based on CDMA).

**Chapter 3 The Data Link Layer**

Starting with the key design issues present in data link layer, this section will study the nature of errors in during transmission, how they can be detected and corrected, i.e., error control and flow control; and several protocols for solving these problems.

**Chapter 4 Medium Access Control**

This section discusses LAN technologies, channel allocation problems and multiple access protocols, especially CSMA/CD (Carrier Sense Multiple Access/Collision Detection). Example networks in this section include Ethernet, 802.11, 802.16, Bluetooth, and RFID. Also, the LAN switching, including VLANs, will be discussed.

**Chapter 5 The Network Layer**

This section covers the design issues of network layer, routing algorithms, congestion control, quality of service and internetworking problems. The network layer in Internet is discussed as an example, including IP addressing and forwarding, Internet control and IP routing protocols, as well as multicasting and mobile IP.

**Chapter 6 The Transport Layer**

This section will introduce the basic principles of transport services, e.g., connectionless and connection-oriented services; the implementation of Internet transport layer, especially the UDP and TCP protocols. The DTN (delay-tolerant network) will also be described.

**Chapter 7 The Application Layer**

This section will discuss several classical Internet applications, including: DNS (Domain Name Systems), Email services, WWW (World Wide Web) and streaming audio/video. Topics of content distribution, including CDNs and peer-to-peer networks, will also be discussed.

**Advanced Networking Technologies**

This section will discuss several recent advances for the Internet technologies. For example, cloud networking, Software Defined Networking (SDN), Data Center Networking and so on

| | |
|---|---|
| Text Book(s) and Reference Materials | Textbook:<br>Computer Networks, 5th edition, Andrew S. Tanenbaum and David J. Wetherall, Prentice Hall, ISBN-10: 0132126958, ISBN-13: 978-0132126953<br>References:<br>Computer Networks, Fifth Edition: A Systems Approach, Larry L. Peterson, Bruce S. Davie, Publisher: Morgan Kaufmann, ISBN-10: 0123850592, ISBN-13: 978-0123850591 |

| Course Title | Computer Networks Lab | | | | |
|---|---|---|---|---|---|
| Credits | 1 | Hrs./Week | 2 | Total Hrs. | 36 |
| Course Code | 60080022 | | Course Type | Compulsory | |
| Prerequisite Course(s) | Operating System, Data Structure | | Course Evaluation | Experiment-based | |

Teaching Objectives:

One's understanding of network protocols can often be greatly deepened by "seeing protocols in action" and by "playing around with protocols" - observing the sequence of messages exchanges between two protocol entities, delving down into the details of protocol operation, and causing protocols to perform certain actions and then observing these actions and their consequences. This course of Computer Networks Labs is to let students have on-hand experience of networking, understanding the principle of protocols through on-going network packets and writing simple networking programs.

Teaching Requirements:

This course will be organized by nine lab experiments, including both verification and comprehensive, to offer students comprehensive understanding on Internet protocols and networking technologies. Experiments required for students include: 1) Ethernet Frames format; 2) IP protocol, IP addressing and IP packet format etc.; 3) ARP (Address Resolution Protocol); 4) ICMP(Internet Control Messages Protocol); 5) UDP (User Datagram Protocol); 6) TCP (Transmission Control Protocol); 7)DNS (Domain Name Service) Protocol; 8) SMTP and POP3 protocols; and 9) HTTP (Hyper Text Transport Protocol).

Teaching Contents:

**Experiment 1 Ethernet Frames Format (4 class hours):**

This experiment is a verification experiment on the Ethernet frames, including the following objective: 1) Study the format of Ethernet frames and definitions of each fields; 2) Study the function of MAC address, including their different types of address; 3) Some useful commands for networking, e.g., ipconfig.

**Experiment 2 IP Protocol, IP Addressing and IP Packet Format (4 class hours)**

This experiment is a comprehensive experiment on IP protocols, including the following objective: 1) The function of IP layer and the IP addressing issue; 2) The classification of subnet and function of subnet mask code; 3) The format of IP packet and the function of network layer; 4) A simple programming to read the IP address of host.

**Experiment 3 ARP (Address Resolution Protocol) (4 class hours)**

This experiment is a verification experiment on the ARP, including the following objective: 1) Study the function and format of ARP protocol; 2) Understand the relationship between IP address and MAC address; 3) Study several arp command.

| | |
|---|---|
| **Experiment 4 ICMP (Internet Control Messages Protocol) (4 class hours)**<br>This experiment is a comprehensive experiment on ICMP protocols, including the following objective: 1) Study the format of ICMP; 2) Study the responses of ICMP protocol and different types of networking problems.<br><br>**Experiment 5 UDP (User Datagram Protocol) (4 class hours)**<br>This experiment is a comprehensive experiment on UDP protocols, including the following objective: 1) Study the principle and process of UDP protocol; 2) Study the format of UDP packet; 3) Understanding the assignment of UDP ports;<br><br>**Experiment 6 TCP (Transmission Control Protocol (4 class hours)**<br>This experiment is a verification experiment on TCP protocols, including the following objective: 1) Study the basic principle of TCP protocol; 2) Study the communication process of TCP protocols.<br><br>**Experiment 7 DNS (Domain Name Service) Protocol (4 class hours)**<br>This experiment is a comprehensive experiment on DNS protocols, including the following objective: 1) Study the principle of DNS implementation; 2) Study the format of DNS messages.<br><br>**Experiment 8 SMTP and POP3 protocols (4 class hours)**<br>This experiment is a verification experiment on EMAIL protocols, including the following objective: 1) Study the basic principle of SMTP and POP3; 2) Study the basic command format of SMTP and POP3; 3) Understand the relationship between Application layer and Transport layer.<br><br>**Experiment 9 HTTP (Hyper Text Transport Protocol) (4 class hours)**<br>This experiment is a verification experiment on HTTP protocols, including the following objective: 1) Study the function of HTTP protocol; 2) Study the implementation of HTTP protocols. | |
| Text Book(s) and Reference Materials | Textbook:<br>Computer Networks, 5th edition, Andrew S. Tanenbaum and David J. Wetherall, Prentice Hall, ISBN-10: 0132126958, ISBN-13: 978-0132126953<br>References:<br>Computer Networks, Fifth Edition: A Systems Approach, Larry L. Peterson, Bruce S. Davie, Publisher: Morgan Kaufmann, ISBN-10: 0123850592, ISBN-13: 978-0123850591 |

<table>
<tr><td>Course Title</td><td colspan="5">Digital Image Processing</td></tr>
<tr><td>Credits</td><td>2</td><td>Hrs./Week</td><td>2</td><td>Total Hrs.</td><td>36</td></tr>
<tr><td>Course Code</td><td colspan="2">60080023</td><td>Course Type</td><td colspan="2">Compulsory</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">Advance programming language, Advance Mathematic</td><td>Course Evaluation</td><td colspan="2">Closed-book Exam</td></tr>
<tr><td colspan="6">Teaching Objectives & Requirements:<br>Visual information plays an important role in almost all areas of our life. Today, much of this information is represented and processed digitally. Digital image processing is ubiquitous, with applications ranging from television to tomography, from photography to printing, from robotics to remote sensing.<br>This course’s teaching objectives are understanding and mastering the basic concepts and methodologies of digital image processing.<br>This course will make extensive use of MATLAB as an analysis, demonstration, and experiment tool, so students should teach themselves some knowledge and techniques about MATLAB.<br>The course is an undergraduate-level introductory course to the fundamentals of digital image processing. It covers topics such as intensity transformations for image enhancement, two-dimensional discrete Fourier transform, spatial and frequency domain linear image filtering, nonlinear image filtering, noise reduction, image restoration, point operations, color processing and image compression.<br><br>Teaching Contents:<br>• Chapter 1 Introduction (4 hours)<br>• Chapter 2 Fundamentals of Digital Images (4 hours)<br>• Chapter 3 Image Enhancement in the Spatial Domain (6 hours)<br>• Chapter 4 Image Enhancement in the Frequency Domain (6 hours)<br>• Chapter 5 Image Restoration (4 hours)<br>• Chapter 6 Color Image Processing (4 hours)<br>• Chapter 7 Image Compression (2 hours)</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">Textbook:<br>R. C. Gonzalez and R. E. Woods, Digital Image Processing (third edition), Prentice Hall, 2010.<br>References:<br>A. K. Jain, Fundamentals of Digital Image Processing, Prentice Hall, 1989.<br>K. R. Castleman, Digital Image Processing, Prentice Hall, 1996.</td></tr>
</table>

| Course Title | Digital Image Processing Lab | | | | |
|---|---|---|---|---|---|
| Credits | 0.5 | Hrs./Week | 1 | Total Hrs. | 16 |
| Course Code | 60080024 | | Course Type | Compulsory | |
| Prerequisite Course(s) | Advance programming language, Advance Mathematic | | Course Evaluation | Experiment-based | |

Teaching Objectives:

The principal objectives of the laboratory are:

(1) To teach the student how to manipulate images using image processing toolbox in MATLAB.

(2) To help in developing a sense of how image processing solutions are prototyped in software

(3) Help students understand the basic image processing techniques.

Teaching Requirements:

(1) To Mast some basic commands and be able to code simple program in MATLAB.

(2) Be able to use the Digital Image Processing Toolbox in MATLAB to do some basic image processing.

(3) Be able understand how to implement basic image processing algorithms.

Teaching Contents:

- **Section 1** Introduction to MATLAB (3 hours)
- **Section 2** Fundamentals of Digital Images Processing Toolbox in MATLAB (3 hours)
- **Section 3** Image Enhancement in the Spatial Domain (4 hours)
- **Section 4** Image Enhancement in the Frequency Domain (4 hours)
- **Section 5** Image Restoration (2 hours)

| Text Book(s) and Reference Materials | Gonzalez, R. C., Woods, R. E., and Eddins, S. L. [2009]. Digital Image Processing Using MATLAB, 2nd ed., Gatesmark Publishing, Knoxville, TN. |
|---|---|

<table>
<tr><td>Course Title</td><td colspan="5">Internet Application Development</td></tr>
<tr><td>Credits</td><td>2</td><td>Hrs./Week</td><td>2</td><td>Total Hrs.</td><td>36</td></tr>
<tr><td>Course Code</td><td colspan="2">60080049</td><td>Course Type</td><td colspan="2">Specialized Optional</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">OOM Programming</td><td>Course Evaluation</td><td colspan="2">Project-based</td></tr>
<tr><td colspan="6">Teaching Objectives & Requirements:<br>This is an introduction course on full-stack Internet-based Application Development Techniques. The course will teach students about basic usage of HTML5, CSS3, JavaScript, Ajax, and json. It focusses on the topics in Static & Dynamic Pages, Interactive Components, Navigation, Validation, Regular Expressions, Persistence, Loggers, Security, Scheduling. Some Java-based frameworks will be introduced to students to finish 4 course projects, including jQuery, VUE, SparkJava, FreeMarker, Sql2o, MyBatis, Vaadin and SpringBoot.<br><br>Teaching Contents:<br>• How the Web Works<br>• HTTP & HTML<br>• CSS Intro<br>• HTML Tables and Forms<br>• Advanced CSS: Layout<br>• JavaScript: Client-Side Scripting<br>• Web Media<br>• Ajax, jQuery and Json<br>• Microservice with SparkJava<br>• Progressive Development with VUE<br>• Component-based Development with Vaadin<br>• Advanced topics on enterprise-level application</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">Textbook:<br>Fundamentals of Web Development, Global Edition, Randy Connolly, Ricardo Hoar, Pearson, 2015<br>References:<br>Web Development Recipes, Brian P. Hogan, Chris Warren, Mike Weber, Chris Johnson, Rebecca Gulick.<br>HTML5 and CSS3 Web Design, Craig Grannell, Victor Sumner, Dionysios Synodinos.<br>Documents of SparkJava, VUE and Vaadin frameworks.</td></tr>
</table>

| Course Title | Human-computer Interaction | | | | |
|---|---|---|---|---|---|
| Credits | 1.5 | Hrs./Week | 1.5 | Total Hrs. | 27 |
| Course Code | 60080074 | | Course Type | Specialized Optional | |
| Prerequisite Course(s) | N/A | | Course Evaluation | Project-based | |

Teaching Objectives & Requirements:

This course covers the principles of human-computer interaction and the design and evaluation of user interfaces. Topics include an overview of human information processing subsystems (perception, memory, attention, and problem solving); how the properties of these systems affect the design of user interfaces; the principles, guidelines, and specification languages for designing good user interfaces, with emphasis on tool kits and libraries of standard graphical user interface objects; and a variety of interface evaluation methodologies that can be used to measure the usability of software. Other topics may include World Wide Web design principles and tools, computer-supported cooperative work, multimodal and "next generation" interfaces, speech and natural language interfaces, and virtual reality interfaces. Course work includes both the creation and implementation of original user interface designs, and the evaluation of user interfaces created by others.

Teaching Contents:

There are two major components to the course, treated in parallel during the semester. The central focus of the course is a semester-long team project, in which students will design, implement and evaluate a user interface. Teams will be incrementally led through the phases of ethnographic study and requirements analysis, scenario-based design, paper prototyping, computer prototyping, and several methods of usability analysis and evaluation. The second component of the course involves exposure to current research in HCI, in order to provide students with an understanding of the range of issues addressed in the field, to provide them with practice reading, presenting and critiquing HCI research, and to provide ideas for team projects. This component of the course will be implemented in a seminar style, with students presenting and critiquing short HCI research papers throughout the course.

| Text Book(s) and Reference Materials | Human-computer Interaction, 3rd Edition, Alan Dix, Janet Finley<br>The Design of Everyday Things, by Donald Norman. Currency/Doubleday, 1990. |
|---|---|

<table>
<tr><td>Course Title</td><td colspan="5">Artificial Intelligence</td></tr>
<tr><td>Credits</td><td>2</td><td>Hrs./Week</td><td>2</td><td>Total Hrs.</td><td>36</td></tr>
<tr><td>Course Code</td><td colspan="2">60080031</td><td>Course Type</td><td colspan="2">Specialized Optional</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">Introduction to Computer Science</td><td>Course Evaluation</td><td colspan="2">Closed-book Exam</td></tr>
<tr><td colspan="6">Teaching Objectives & Requirements:<br>The goal of Artificial Intelligence (AI) is the design of agents that can behave rationally in the real world by sensing their environment, planning their goals, and acting to maximally achieve these goals. This course provides an introductory survey to the techniques and applications of modern AI. The course will cover a broad range of conceptual approaches, from logic to probabilistic reasoning, and a broad range of applications. Lectures will stress not only the technical concepts themselves, but also the history of ideas behind them.<br><br>Teaching Contents:<br>Part I Artificial Intelligence<br>1 Introduction<br>2 Intelligent Agents<br>Part II Problem Solving<br>3 Solving Problems by Searching<br>4 Beyond Classical Search<br>5 Adversarial Search<br>6 Constraint Satisfaction Problems<br>Part III Knowledge and Reasoning<br>7 Logical Agents<br>8 First-Order Logic<br>9 Inference in First-Order Logic<br>10 Classical Planning<br>11 Planning and Acting in the Real World<br>12 Knowledge Representation</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">Textbook:<br>Artificial Intelligence: A Modern Approach, Stuart J. Russell and Peter Norvig. 3rd ed. Prentice Hall<br>References:<br>Artificial Intelligence: Foundations of Computational Agents<br>by Poole and Mackworth, Cambridge Univ. Press, 2010</td></tr>
</table>

<table>
<tr><td>Course Title</td><td colspan="5">Formal Methods in Software Engineering</td></tr>
<tr><td>Credits</td><td>2</td><td>Hrs./Week</td><td>2</td><td>Total Hrs.</td><td>36</td></tr>
<tr><td>Course Code</td><td colspan="2">60080032</td><td>Course Type</td><td colspan="2">Specialized Optional</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">Discrete Mathematics</td><td>Course Evaluation</td><td colspan="2">Research Essay</td></tr>
<tr><td colspan="6">Teaching Objectives & Requirements:<br>This course is an introduction to the theory and applications of formal methods, a field of computer science and engineering concerned with the rigorous mathematical specification, design, and verification of systems. At its core, formal methods is about proof: formulating specifications that form proof obligations, designing systems to meet those obligations, and verifying, via algorithmic proof search, that the systems indeed meet their specifications.<br><br>Teaching Contents:<br>The course will cover topics such as model checking and Boolean satisfiability (SAT) solving. The course material has applications to several areas including computer security, software engineering, embedded/cyber-physical systems, control systems, robotics, networking and distributed systems, education, systems and synthetic biology, and CAD for integrated circuits. We will therefore focus on fundamental theory and techniques that apply broadly to many systems. A tentative list of topics to be covered:<br>SAT & BDDs: Complexity, modern DPLL SAT solvers, Binary Decision Diagrams, BDD representation and manipulation algorithms, applications.<br>Model Checking: Modeling with finite automata, Kripke structures and extended finite automata. Temporal logic. Explicit-state model checking, partial-order reduction. Basic fixpoint theory, symbolic model checking, abstraction, bounded model checking.</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">1. Michael Huth and Mark Ryan Title: Logic in Computer Science: Modelling and Reasoning about Systems Publisher: Cambridge Univ. Press Year: June 2004 (2nd edition)<br>2. Edmund M. Clarke, Jr., Orna Grumberg, and Doron A. Peled Title: Model Checking Publisher: MIT Press Year: January 2000</td></tr>
</table>

| Course Title | Computer Vision | | | | |
|---|---|---|---|---|---|
| Credits | 2 | Hrs./Week | 2 | Total Hrs. | 36 |
| Course Code | 60080034 | | Course Type | Specialized Optional | |
| Prerequisite Course(s) | Linear algebra, Digital Image Processing | | Course Evaluation | Project-based | |

Teaching Objectives & Requirements:

A basic problem of computer vision is to understand the structure of a real-world scene given several images of it. Underpinning of the solutions to this problem is the intricate geometric relations that exist between the images of real-world scenes. The goal of this course is to provide the students a complete theoretical understanding of the geometry between multiple uncalibrated views and allow them to use these concepts to calculate the properties of scene and camera from real world images. Recent major developments in the theory and practice of scene reconstruction as well as the mathematics that is necessary for understanding the underlying geometric concepts will be introduced.

The objectives of the course are:

- To understand the relation between multiple views of the scene,
- To be able to retrieve camera parameters and 3d space points from image correspondences,
- To understand the general principles of parameter estimation,
- To obtain basic knowledge on projective geometry.

Teaching Contents:

- Week 1: Introduction
- Week 2: Projective 2D Geometry: homogeneous coordinate, points, lines conics, dual conics, line at infinity
- Week 3: Euclidean & Similarity Transformation: translation, rotation, scaling
- Week 4: Affine & Projective Transformation: affine rectify, similarity rectify, vanishing points, circular points, dual conic of circular points
- Week 5: Parameter Estimation: DLT algorithm, cost functions, Maximum likelihood estimation
- Week 6: Projective 3D Geometry: plane, lines, quadrics, twisted cubic, the plane at infinity, the absolute conic
- Week 7: Camera Model & Calibration: finite cameras, the projective camera, restricted camera estimation, radial distortion
- Week 8: Single View Geometry: action of a projective camera on planes, lines, conics and quadrics, image of the absolute conic, camera center
- Week 9: Epipolar Geometry: epipolar geometry, fundamental matrix, essential matrix
- Week 10: 3D Reconstruction: reconstruction ambiguity, stratified reconstruction
- Week 11: Fundamental Matrix

| | |
|---|---|
| • Week 12: Structure Computation<br>• Week 13: Planes & Homographs: the infinite homograph<br>• Week 14-15: Paper Presentation<br>• Week 16-18: Review | |
| Text Book(s) and Reference Materials | Multiple View Geometry in Computer Vision, R. Hartley and A. Zisserman, second edition, Cambridge University Press, 2003. |

| Course Title | C++ Programming | | | | |
|---|---|---|---|---|---|
| Credits | 2 | Hrs./Week | 2 | Total Hrs. | 36 |
| Course Code | 60080027 | | Course Type | Specialized Optional | |
| Prerequisite Course(s) | N/A | | Course Evaluation | Closed-book Exam | |

Teaching Objectives & Requirements:

This course teaches the students how to write high quality, internally documented and well-structured C++ programs. Students will learn how C++ supports object-oriented programming principles, such as abstraction, information hiding and modularity. Students will see how C++ incorporates and improves upon ISO C and gain experience with syntax and semantics of the ISO standard C++ language.

After this course a student should be able to:

- Understand the basic concept of object-oriented programming.
- Familiar with the new character of C++ syntax, understand the difference between ISO C and C++.
- Build C++ programs using object-oriented programming.

Teaching Contents:

**1. C++ Basics (4 hours)**

• Introduction to C++
• Variables, Expressions, and Assignment Statements
• Program Style
• Libraires and Namespaces

**2. Function Basics (1 hour)**

• Predefined Functions
• Programmer-Defined Functions
• Reference
• Scope Rules

**3. Parameters and Overloading (1 hour)**

• Parameters
• Overloading and Default Arguments
• Testing and Debugging Functions

**4. Arrays (2 hours)**

• Introduction to Arrays
• Arrays in Functions
• Programming with Arrays
• Multidimensional Arrays

**5. Structures and Classes (2 hours)**

• Structures
• Classes

**6. Constructors and Other Tools (2 hours)**

• Constructors

• More Tools

• Vectors-A Preview of the Standard Template Library

**7. Operator Overloading, Friends, and References (4 hours)**

• Basic Operator Overloading

• Friend Functions and Automatic Type Conversion

• References and More Overloaded Operators

**8. Strings (2 hours)**

• An Array Type for Strings

• Character Manipulation Tools

• The Standard Class string

**9. Pointers and Dynamic Arrays (2 hours)**

• Pointers

• Dynamic Arrays

• Classes, Pointers, and Dynamic Arrays

**10. Streams and File I/O (2 hours)**

• I/O Streams

• Tools for Stream

• Stream Hierarchies: A Preview of Inheritance

**11. Recursion (2 hours)**

• Recursive Void Functions

• Recursive Functions that Return a Value

• Thinking Recursively

**12. Inheritance (2 hours)**

• Inheritance Basics

Programming with Inheritance

**13. Polymorphism and Virtual Functions (2 hours)**

• Virtual Function Basics

• Pointers and Virtual Functions

**14. Templates (2 hours)**

• Function Templates

• Class Templates

• Templates and Inheritance

| | |
|---|---|
| Text Book(s) and Reference Materials | Absolute C++, 4th edition, by Walter Savitch and Kenrick Mock, Addison-Wesley |

| Course Title | ACM Programming | | | | |
|---|---|---|---|---|---|
| Credits | 1 | Hrs./Week | 1 | Total Hrs. | 18 |
| Course Code | 60080063 | | Course Type | Specialized Optional | |
| Prerequisite Course(s) | N/A | | Course Evaluation | Project-based | |

Teaching Objectives & Requirements:

The students will gain basic knowledge on ACM/ICPC and will be familiar with computer related subjects, such as computer English, data structure, discrete mathematics, elementary number theory, numeric calculation, algorithms, artificial intelligence and computation geometry. They will be able to use those skills to design and implement application and to solve the problems during the contests.

The course will be arranged according to requirements of programming contests. Through the lectures of specific topics, combing practice and discussion in classroom, the students will prompt their ability of algorithm designing and coding.

Teaching Contents:

The topics will cover the following:

The application of binary representation of integer numbers
Sorting and the k-th bigger number
Linked list
Dynamic programming
Computation geometry
Graph theory and algorithms
Big number and number theory
High performance computing and code optimization
Data structure and algorithm
Coding and debugging

| Text Book(s) and Reference Materials | Absolute C++, 4th edition, by Walter Savitch and Kenrick Mock, Addison-Wesley |
|---|---|

| Course Title | ACM Programming Lab | | | | |
|---|---|---|---|---|---|
| Credits | 0.5 | Hrs./Week | 1 | Total Hrs. | 18 |
| Course Code | 60080064 | | Course Type | Specialized Optional | |
| Prerequisite Course(s) | N/A | | Course Evaluation | Project-based | |

Teaching Objectives & Requirements:

These experiments are for the "ACM programming" course. The students are required to do 8 experiments on an online judge website, through which they will gain experience on solving problems, designing algorithm and debugging until the solution is accepted. The experiments including bit operation, sorting, linked list, dynamic programming, computation geometry, graph theory, big number and code optimization. There are many more problems available if one try to explore more. The final examination will be conducted on the online judge website.

The students will be familiar with ACM/ICPC doing these experiments. They will advance their ability of problem solving and coding, which leads to a promotion to their thoughts and programming ability.

The students are required to solve problem on the online judge website. Each experiment contains 1 to 3 problems.

Teaching Contents:

The application of binary representation of integer numbers (2 problems)
Sorting and the k-th bigger number (2 problems)
Linked list (2 problems)
Dynamic programming (3 problems)
Computation geometry (2 problems)
Graph theory and algorithms (3 problems)
Big number and number theory (2 problems)
High performance computing and code optimization (1 problem)

| Text Book(s) and Reference Materials | Absolute C++, 4th edition, by Walter Savitch and Kenrick Mock, Addison-Wesley |
|---|---|

| Course Title | Cryptographic Algorithms and Protocols | | | | |
|---|---|---|---|---|---|
| Credits | 2 | Hrs./Week | 2 | Total Hrs. | 36 |
| Course Code | 60080041 | | Course Type | Specialized Optional | |
| Prerequisite Course(s) | Probability Statistics | | Course Evaluation | Closed-book Exam | |

Teaching Objectives:

This course covers the basic concepts in Cryptography, the basic cryptographic algorithms and protocols, including the classical cryptosystems, the block cryptosystems, the public-key cryptosystems, Hash functions, digital signature schemes. This course will provide students with a fundamental support for doing further academic research and engineering practice in Cryptography.

Teaching Requirements:

1. To master the basic concepts and related theorems in Number Theory, Algebra, Probability and Information Theory;
2. To grasp the basics of the cryptosystems and cryptanalysis;
3. To master some classical cryptosystems, such as the Affine Cipher, the Substitution Cipher, the Permutation Cipher and the Stream Cipher;
4. To master two block cryptosystems: the DES and AES;
5. To grasp the basics of Hash functions, the security and Random Oracle Model, the famous SHA algorithm;
6. To master the basics of the public-key cryptosystems, including:
   - The RSA encryption;
   - The ElGamal encryption;
   - The digital (RSA and ElGamal) Signature Schemes.

Teaching Contents:

1) Chapter 1-Classical Cryptography: The Shift Cipher, The Substitution Cipher, The Affine Cipher, The Permutation Cipher, Stream Ciphers;
2) Chapter 2-Shannon’s Theory: Elementary of Probability Theory and Entropy, Perfect Secrecy;
3) Chapter 3-Block Ciphers: Basics of the Substitution-Permutation Network, the Data Encryption Standard (DES), the Advanced Encryption Standard (AES);
4) Chapter 4-Hash Functions: Basics and Security of Hash Functions, Iterated Hash Functions and the Secure Hash Algorithm (SHA-1), the Message Authentication Code (MAC);
5) Chapter 5-The RSA Cryptosystem: Introduction to the Public-key Cryptography, The RSA Cryptosystem, Factoring Integers, Factoring Algorithms and other attacks on RSA, the Rabin

<table>
<tr><td colspan="2">Cryptosystem, Semantic Security of RSA;<br>6) Chapter 6-The ElGamal Cryptosystem: The Discrete Logarithm Problem (DLP) and Algorithms, Finite Field and Elliptic Curves, Security of ElGamal Systems;<br>7) Chapter 7-Signature Schemes: Basics of and Security Requirements for Signature Schemes, the RSA signature scheme, the ElGamal signature scheme, the Digital Signature Standard (DSS);<br>8) Some Research Topics in Cryptography.</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td>1. Textbook: Douglas R. Stinson, “Cryptography: Theory and Practice,” 3rd Edition, published by Chapman & Hall/CRC, 2006.<br>2. Reference: Wenbo Mao, “Modern Cryptography: Theory and Practice,” published by Person Education, 2004.<br>3. Reference: Hans Delfs and Helmut Knebl, “Introduction to Cryptography: Principles and Applications,” published by Springer-Verlag, 2007.<br>Reference: William Stallings, “Cryptography and Network Security: Principles and Practices,” 4th Edition, published by Person Education, 2006.</td></tr>
</table>

<table>
<tr><td>Course Title</td><td colspan="5">Pervasive Computing</td></tr>
<tr><td>Credits</td><td>2</td><td>Hrs./Week</td><td>2</td><td>Total Hrs.</td><td>36</td></tr>
<tr><td>Course Code</td><td colspan="2">60080044</td><td>Course Type</td><td colspan="2">Specialized Optional</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">N/A</td><td>Course Evaluation</td><td colspan="2">Open-book Exam</td></tr>
<tr><td colspan="6">Teaching Objectives:<br>The course of《pervasive computing》 is based on the basic technology of pervasive computing. Starting from the academic achievements and application research in this field at home and abroad, it introduces the technology of pervasive computing in an all-round way, and probes into the related problems involved in computing and computing mode. From the formation of the concept of pervasive computing as the source, combined with the cross technology involved in the Internet of things and the field of pervasive computing, this course systematically introduces the core technology in pervasive computing. As an elective course for computer science major of International College, pervasive computing aims to enable students to preliminarily master the concept, key technology and application in various fields. Through the study of this course, students' professional foundation, scientific literacy and practical ability can be effectively improved.<br><br>Teaching Requirements:<br>This course is mainly offered to the senior students majoring in computer science and technology and network engineering in International College. Students who choose this course are required to have a good theoretical foundation of computer network. As the course “computer network” is the core supporting course for the computer major to carry out scientific research and engineering application, it is suggested that the related courses be taken as the prerequisite courses of this course.<br><br>Teaching Contents:<br>With the development and popularization of computer science, especially the emergence of the concept of Internet of things, “pervasive computing” course has become a computer course that college students of science and engineering must master. The course of pervasive computing is an interdisciplinary course, which is a comprehensive basic course related to the Internet of things contacted by students at the undergraduate stage. The main content of the course includes the overview, the architecture, the hardware technology, the wearable computing technology, the service technology of pervasive computing, etc., and it mainly focuses on the sensor technology, its related protocol design, the mobility localization method, and the seamless handover technology of pervasive computing.</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">English research papers based</td></tr>
</table>

<table>
<tr><td>Course Title</td><td colspan="5">Topics in Computer Security</td></tr>
<tr><td>Credits</td><td>2</td><td>Hrs./Week</td><td>2</td><td>Total Hrs.</td><td>36</td></tr>
<tr><td>Course Code</td><td colspan="2">60080048</td><td>Course Type</td><td colspan="2">Specialized Optional</td></tr>
<tr><td>Prerequisite Course(s)</td><td colspan="2">N/A</td><td>Course Evaluation</td><td colspan="2">Assignment + Essay</td></tr>
<tr><td colspan="6">Teaching Objectives & Requirements:<br>The course aims to provide a general introduction to topic in the theory and practice of computer security. This requires students to attend a weekly course, finish three assignments and report a final project from a selected topic in security.<br><br>Teaching Contents:<br>The contents of the course cover the basic cryptography and cryptanalysis, algorithms and computational complexities, security protocols, authentication and authorization, access control, protection from malicious software, intrusion detection, as well as other advanced topics such as Digi cash, Blockchain and smart contracts.</td></tr>
<tr><td>Text Book(s) and Reference Materials</td><td colspan="5">Computer Security --- Principles and Practice, William Stallings, Lawrie Brown, Pearson 2012 for the second edition</td></tr>
</table>

| Course Title | Object Oriented Methodology | | | | |
|---|---|---|---|---|---|
| Credits | 2 | Hrs./Week | 2 | Total Hrs. | 36 |
| Course Code | 60080033 | | Course Type | Specialized Optional | |
| Prerequisite Course(s) | N/A | | Course Evaluation | Closed-book Exam | |

Teaching Objectives:

This course introduces the principles, methods and techniques of systems analysis and design, with an emphasis on object-oriented methodology. It enables students to apply UML (Unified Modeling Language) and design patterns for object modeling and apply object-oriented analysis and design (OOA/D) principles and techniques to create robust and maintainable systems.

Teaching Requirements:

After successfully completing this course, students will be able to:

1. understand different perspectives about the systems development process;
2. understand the basic principles of object-oriented analysis and design;
3. use UML and UP (Unified Process) for object-oriented analysis and design;
4. apply design patterns including GoF patterns and GRASP patterns for object design;
5. become aware of the emerging ideas relevant to information systems development.

Teaching Contents:

**Module 1: Overview of Software Development Process**

Waterfall life cycle; Iterative and incremental process; agile methods

**Module 2: Overview of UML and Unified Process**

Analysis and design; object-oriented analysis and design; UML overview; Unified Process overview.

**Module 3: Requirements Workflow**

Types of requirements; methods for finding requirements; use cases and use case models.

**Module 4: Object-Oriented Analysis Workflow**

Domain model; Guidelines for finding and describing conceptual classes.

**Module 5: Object-Oriented Design Workflow**

Responsibility-driven object design; UML class diagrams, sequence diagrams, and communication diagrams; GRASP and GoF design patterns.

**Module 6: Other topics for system analysis and design**

Test-driven development and refactoring; architectural analysis; iterative and agile project management.

| | |
|---|---|
| Text Book(s) and Reference Materials | Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process, Craig Larman, ISBN: 9787111178415 |

| Course Title | Computer Architecture | | | | |
|---|---|---|---|---|---|
| Credits | 2 | Hrs./Week | 2 | Total Hrs. | 36 |
| Course Code | 60080035 | | Course Type | Specialized Optional | |
| Prerequisite Course(s) | Computer Organization | | Course Evaluation | Closed-book Exam | |

Teaching Objectives & Requirements:

Course's aim is to help students become a better programmer by teaching students the basic concepts underlying all computer systems. This course wants students to learn what really happens when their programs run, so that when things go wrong (as they always do) students will have the intellectual tools to solve the problem. By the end of the course, students will be prepared to take any of the upper-level systems classes. Even more important, students will have learned skills and knowledge that will help students throughout their career. In addition, this course requires students to study the computer organization.

Teaching Contents:

**Chapter 1** Introduction (1 hours)
**Chapter 2** COMPUTER EVOLUTION AND PERFORMANCE (1 hours)
**Chapter 3** A TOP-LEVEL VIEW OF COMPUTER FUNCTION AND INTERCONNECTION (2 hours)
**Chapter 4** CACHE MEMORY (2 hours)
**Chapter 5** INTERNAL MEMORY (2 hours)
**Chapter 6** EXTERNAL MEMORY (2 hours)
**Chapter 7** INPUT/OUTPUT (2 hours)
**Chapter 8** OPERATING SYSTEM SUPPORT (2 hours)
**Chapter 9** NUMBER SYSTEMS (2 hours)
**Chapter 10** COMPUTER ARITHMETIC (2 hours)
**Chapter 11** DIGITAL LOGIC (2 hours)
**Chapter 12** INSTRUCTION SETS: CHARACTERISTICS AND FUNCTIONS (2 hours)
**Chapter 13** INSTRUCTION SETS: ADDRESSING MODES AND FORMATS (2 hours)
**Chapter 14** PROCESSOR STRUCTURE AND FUNCTION (2 hours)
**Chapter 15** REDUCED INSTRUCTION SET COMPUTERS (2 hours)
**Chapter 16** INSTRUCTION-LEVEL PARALLELISM AND SUPERSCALAR PROCESSORS (2 hours)
**Chapter 17** PARALLEL PROCESSING (3 hours)

| Text Book(s) and Reference Materials | William Stallings. Computer Organization and Architecture: Designing for Performance, Ninth Edition, Pearson Education, 2013. |
|---|---|

| Course Title | Internet Security and E-commerce Protocols | | | | |
|---|---|---|---|---|---|
| Credits | 2 | Hrs./Week | 3 | Total Hrs. | 36 |
| Course Code | 60080042 | | Course Type | Specialized Optional | |
| Prerequisite Course(s) | N/A | | Course Evaluation | Assignment + Essay | |

Teaching Objectives & Requirements:

This course aims to provide an understanding of Internet security. Students are expected to gain a broad understanding of Internet security with the goal of recognizing security problems and discovering the security requirements of current computer systems. The course explores existing security mechanisms and offers students the opportunity to evaluate and design techniques for enforcing Internet security and developing secure E-commerce protocols.

This course aims to discuss technologies to facilitate secure online communications, such as SSL and digital certificates, are presented. Canonical threats to web security, such as input validation, XSS and CSRF attacks, are demonstrated. Mechanisms for secure payments, such as EMV, tokenization and mobile payments protocols, are discussed along with case studies of attacks on deployed systems. The technical architecture of cryptocurrencies, notably Bitcoin, are presented. Throughout the course, economic considerations, notably the incentives of system designers and attackers, are discussed.

Teaching Contents:

| **Part** | **TOPIC** |
|---|---|
| Introduction | Syllabus & Introduction: Security, Privacy, E-commerce on the Internet |
| Cryptography | Symmetric Encryption |
| | Block Cipher |
| | Public Key Encryption |
| | Message Authentication Codes (MACs) |
| Data Privacy | Data Anonymization |
| | Differential Privacy |
| | Location Privacy |
| | Local Differential Privacy |
| Web Security | Buffer Overflow |
| | Command Injection |
| | SQL Injection |
| | Cross Site Scripting (XSS) |
| | Firewalls |
| | Intrusion Detection |
| | Transport Layer Security/ Secure Sockets Layer (SSL/TLS) |
| E-commerce Protocols | Legacy Payment Systems |
| | EMV Operation and Attacks |

| | |
|---|---|
| | Credit Card Data Security |
| | Protecting Sensitive Data: From Passwords to PANs |
| | Bitcoin |
| Text Book(s) and Reference Materials | No Required Textbooks. (Course related materials will be released) |