SYLLABUS

for

M.Sc. Computer Science (Two Year Program)

(2026 Onwards)

Master of Science (CS)

Programme

As Per

New Education Policy (NEP-2020)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF ENGINEERING AND TECHNOLOGY

HEMVATI NANDAN BAHUGUNA GARHWAL UNIVERSITY

A CENTRAL UNIVERSITY

SRINGAR GARHWAL – (246174)

Program Structure

Semester	Course Category	Course Title	Credits		Total Credit
			T	P	
1	**Discipline Specific Core [DSC]**	Advanced Operating System	3	2	5
		Computer Organization and Architecture	3	2	5
		Machine Learning	3	2	5
		Theory of Computation	5	-	5
	Discipline Specific Elective [DSE]	Oops using C++/ Python	3	1	4
	Or				
	Multi-Disciplinary Elective [MDE]	Internet of Things	3	1	4
	Total				**24**

Semester	Course Category	Course Title	Credits		Total Credit
			T	P	
2	**Discipline Specific Core [DSC]**	Data Structures Using C	3	2	5
		Software Engineering	5	-	5
		Network Security and Cryptography	5	-	5
		Database Management System	3	2	5
	Discipline Specific Elective [DSE]	Graphic and Animation/ Compiler Design	3	1	4
	Or				
	Multi-Disciplinary Elective [MDE]	Quantum Computation	3	1	4
	Total				**24**

Semester	Course Category	Course Title	Credits		Total Credit
			T	P	
3	**Discipline Specific Core [DSC]**	Design and Analysis of Algorithm	3	2	5
		Combinatorics and Graph Theory	5	-	5
		Java Programming	3	2	5
		Natural Language Processing	3	2	5
	Discipline Specific Elective [DSE]	Computer Vision/Fuzzy Logic & Neural Networks/Android Application Dev/MOOC	3	1	4
	Or				
	Multi-Disciplinary Elective [MDE]	Genetic Algorithm and Application	3	1	4
		Total			24

Semester	Course Category	Course Title	Credits		Total Credit
			T	P	
4	**Discipline Specific Core [DSC]**	Project Work	-	10	10
		Data Warehousing and Data Mining	5	-	5
		E-Commerce	5	-	5
	Discipline Specific Elective [DSE]	Artificial Intelligence/ Image Processing /MOOC	3	1	4
	Or				
	Multi-Disciplinary Elective [MDE]	Bioinformatics	3	1	4
		Total			24

SEMESTER 1

Course: Advanced Operating System (SET/CSE/MCS/DSC101)

Course Objectives

1. Understand the basic concepts and functions of operating systems.
2. Learn process, memory, and CPU scheduling techniques.
3. Understand process synchronization and deadlock management.
4. Study device, storage, and file system management in UNIX/Linux.

Course Outcomes

CO1: Explain the fundamentals and types of operating systems.

CO2: Apply memory management and CPU scheduling techniques.

CO3: Analyze process synchronization and deadlock handling methods.

CO4: Describe device management, storage management, and file system concepts in UNIX/Linux.

Course Content/Syllabus:

Unit I: Introduction

Introduction, what is an Operating System, Simple Batch Systems, Multiprogrammed Batch Systems, Time-Sharing Systems, Personal Computer Systems, Parallel Systems, Distributed Systems, Real-Time Systems.

Unit II: Memory Management and Virtual Memory

Memory Management: Background, Logical versus Physical Address Space, Swapping, Contiguous Allocation, Paging, Segmentation, Segmentation with Paging. Virtual Memory: Demand Paging, Page Replacement, Page Replacement Algorithms, Performance of Demand Paging, Allocation of Frames, Thrashing, Other Considerations, Demand Segmentation.

Unit III: Processes and CPU Scheduling

Processes: Process Concept, Process Scheduling, Operations on Processes, Cooperating Processes, Interprocess Communication.CPU Scheduling: Basic Concepts, Scheduling Criteria, Scheduling Algorithms, Multiple-Processor Scheduling, Real-Time Scheduling, Algorithm Evaluation.

Unit IV: Process Synchronization and Deadlocks

Process Synchronization: Background, The Critical-Section Problem, Synchronization Hardware, Semaphores, Classical Problems of Synchronization, Critical Regions, Monitors, Synchronization in Solaris 2, Atomic Transactions. Deadlocks: System Model, Deadlock Characterization, Methods for Handling Deadlocks, Deadlock Prevention, Deadlock Avoidance, Deadlock Detection, Recovery from Deadlock, Combined Approach to Deadlock Handling.

Unit V: Device, Storage and File Management

Device Management: Techniques for Device Management, Dedicated Devices, Shared Devices, Virtual Devices, Device Characteristics, Hardware Consideration, Input and Output Devices, Storage Devices, Channels and Control Units, Independent Device Operation, Buffering, Multiple Paths, Block Multiplexing, Device Allocation Consideration.

Secondary-Storage Structure: Disk Structure, Disk Scheduling, Disk Management, Swap-Space Management, Disk Reliability, Stable-Storage Implementation. Information Management: Introduction, A Simple File System, General Model of a File System, Symbolic File System, Basic File System, Access Control Verification, Logical File System, Physical File System.

File-System Interface: File Concept, Access Methods, Directory Structure, Protection, Consistency Semantics. File-System Implementation: File-System Structure, Allocation Methods, Free-Space Management, Directory Implementation, Efficiency and Performance, Recovery.

References:

1. Abraham Silberschatz and Peter B. Galvin, Operating System Concepts.
2. Milan Milanković, Operating Systems: Concepts and Design, McGraw-Hill.
3. R. C. Joshi, Operating Systems, Wiley Dreamtech India Pvt. Ltd.
4. Harvey M. Deitel, Operating Systems, Addison-Wesley.

SEMESTER 1

Course: Computer Organization and Architecture (SET/CSE/MCS/DSC102)

Course Objectives

1. Understand the basic concepts of computer organization and architecture.
2. Learn data representation and digital logic design.
3. Study processor organization, memory hierarchy, and I/O systems.
4. Understand instruction execution and addressing techniques.
5. Explore computer hardware components and their interaction.

Course Outcomes

CO1: Explain data representation, logic circuits, and digital components.

CO2: Describe the organization and operation of processor, memory, and buses.

CO3: Apply addressing modes and instruction execution concepts.

CO4: Analyze memory hierarchy, cache, and virtual memory techniques.

CO5: Explain I/O organization and the role of computer peripherals.

Course Content/Syllabus:

Unit I: Representation of Information and Digital Logic

Number System: Binary, Octal, Hexadecimal and their conversion, Character Codes: BCD, ASCII, EBCDIC, Digital Codes: Gray Code, XS-3 Code. Logic Circuits: Basic Logic Functions, Synthesis of Logic Functions using AND, OR and NOT Gates, Minimization of Logic Expressions, Synthesis with NAND and NOR Gates, Implementation of Logic Gates, Flip-Flops, Registers and Shift Registers, Counters, Decoders, Multiplexers, Programmable Logic Devices, Sequential Circuits.

Unit II: Basic Structure of Computer Hardware and Software

Functional Units, Basic Operational Concepts, Bus Structures, Software, Performance, Distributed Computing. Addressing Methods: Basic Concepts, Memory Locations, Main Memory Operations, Addressing Modes, Basic I/O Operations, Stacks and Queues, Subroutines.

Unit III: Processing Unit

Some Fundamental Concepts, Execution of a Complete Instruction, Hardwired Control, Performance Considerations, Microprogrammed Control, Signed Addition and Subtraction, Arithmetic and Branching Conditions, Multiplication of Positive Numbers, Signed-Operand Multiplication, Fast Multiplication, Integer Division, Floating-Point Numbers and Operations.

Unit IV: Input-Output Organization and Memory

Input-Output Organization: Accessing I/O Devices, Interrupts, Direct Memory Access (DMA), I/O Hardware, Standard I/O Interfaces. Memory: Semiconductor RAM Memories, Read-Only Memories (ROM), Cache Memories, Performance Considerations, Virtual Memories, Memory Management Requirements.

Unit V: Computer Peripherals

Introduction to Computer Peripherals: I/O Devices, On-Line Storage.

References

1. William Stallings, *Computer Organization and Architecture*, Pearson Education Asia.
2. M. Morris Mano, *Computer System Architecture*, PHI.
3. Zaky and Hamacher, *Computer Organization*, McGraw-Hill.
4. B. Ram, *Computer Fundamentals: Architecture and Organization*, New Age International.
5. Andrew S. Tanenbaum, *Structured Computer Organization*, PHI.
6. J. P. Hayes, *Computer Architecture and Organization*, McGraw-Hill.
7. G. L. Jr., *Computer Design*, Computech Press Langdon.
8. Bywater, *Hardware-Software Design of Digital Systems*.

SEMESTER 1

Course: Machine Learning (SET/CSE/MCS/DSC103)

Course Objectives:

This course introduces the fundamental concepts and techniques of machine learning, including supervised, unsupervised, and reinforcement learning. It enables students to develop predictive models using decision trees, Bayesian learning, neural networks, genetic algorithms, and data mining techniques for solving real-world problems.

Course Outcomes

1. Explain the fundamental concepts, types, and applications of machine learning.
2. Apply decision tree and Bayesian learning algorithms for classification and prediction tasks.
3. Design and implement artificial neural networks for solving learning problems.
4. Analyze optimization techniques using genetic algorithms and evolutionary learning methods.
5. Apply classification and clustering techniques to extract meaningful patterns from data.

Course Content/Syllabus

Unit 1: Introduction: Well-Posed learning problems, Basic concepts, Designing a learning system, Issues in machine learning. Types of machine learning: Learning associations, Supervised learning, Unsupervised learning, Reinforcement learning

Unit 2: Decision Tree Learning: Decision tree representation, appropriate problems for decision tree learning, Univariate Trees (Classification and Regression), Multivariate Trees, Basic Decision Tree Learning algorithms, Hypothesis space search in decision tree learning, Inductive bias in decision tree learning, Issues in decision tree learning.

Unit 3: Bayesian Learning: Bayes theorem and concept learning, Bayes optimal classifier, Gibbs algorithms, Naive Bayes Classifier, Bayesian belief networks, The EM algorithm.

Unit 4: Artificial Neural Network: Neural network representation, Neural Networks as a paradigm for parallel processing, Linear discrimination, pairwise separation, Gradient Descent, Logistic discrimination, Perceptron, Training a perceptron, Multilayer perceptron, Back propagation Algorithm. Recurrent Networks, dynamically modifying network structure.

Unit 5: Genetic Algorithms: Basic concepts, Hypothesis space search, Genetic programming, Models of evolution and learning, Parallelizing Genetic Algorithms.

Unit 6: Data Mining Techniques for Analysis: Classification: Decision tree induction, Bayes classification, Rule-based classification, Support Vector Machines, Classification Using Frequent Patterns, k-Nearest-Neighbor, Fuzzy-set approach Classifier, Clustering: K-Means, k-Medoids, Agglomerative versus Divisive Hierarchical Clustering Distance Measures in Algorithmic Methods, Mean-shift Clustering

References:

1. Mitchell T.M., Machine Learning, McGraw Hill
2. Bishop C., Pattern Recognition and Machine Learning, Springer-Verlag
3. Stephen Marsland, Machine Learning: An Algorithmic Perspective, CRC Press

SEMESTER 1

Course: Theory of Computation (SET/CSE/MCS/DSC104)

Course Objectives

1. Understand the fundamentals of automata theory and formal languages.
2. Learn finite automata, regular languages, and regular expressions.
3. Study context-free grammars and pushdown automata.
4. Understand Turing machines and computability concepts.
5. Develop the ability to analyze formal language models and computational problems.

Course Outcomes

CO1: Explain the concepts of finite automata and regular languages.

CO2: Apply regular expressions and grammars to language recognition problems.

CO3: Analyze context-free grammars and pushdown automata.

CO4: Explain the operation of Turing machines and models of computation.

CO5: Evaluate the computational capability and limitations of different automata.

Course Content/Syllabus

Unit I: Introduction to Theory of Computation and Finite Automata

Introduction to Theory of Computation, Mathematical Preliminaries and Notation, Basic Concepts, Applications, Deterministic Finite Automata (DFA), Nondeterministic Finite Automata (NFA), Equivalence of DFA and NFA, Minimization of Finite Automata.

Unit II: Regular Languages and Regular Grammars

Regular Languages, Regular Expressions, Relationship between Regular Expressions and Regular Languages, Regular Grammars, Closure Properties of Regular Languages, Decision Problems, Identifying Languages.

Unit III: Context-Free Grammars

Context-Free Languages, Context-Free Grammars, Parsing and Ambiguity, Context-Free Grammars in Programming Languages, Grammar Transformations, Simplification of Context-Free Grammars, Chomsky Normal Form (CNF), Greibach Normal Form (GNF).

Unit IV: Pushdown Automata

Nondeterministic Pushdown Automata, Pushdown Automata and Context-Free Languages, Deterministic Pushdown Automata, Deterministic Context-Free Languages, Pumping Lemmas, Closure Properties, Decision Algorithms for Context-Free Languages.

Unit V: Turing Machines

Turing Machines, Standard Turing Machine, Combining Turing Machines for Complex Tasks, Turing Thesis, Variations of Turing Machines, Universal Turing Machine.

References

1. Peter Linz, *An Introduction to Formal Languages and Automata*, Narosa Publishing House, 1997.
2. John C. Martin, *Introduction to Languages and the Theory of Computation*, McGraw-Hill, 1997.
3. John E. Hopcroft and Jeffrey D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, Narosa Publications.

SEMESTER 1

Course: Object Oriented Programming Using C++ (SET/CSE/MCS/DSE1.1)

Course Objectives

1. Understand the principles of object-oriented programming.
2. Learn programming concepts using C++.
3. Develop applications using classes, inheritance, and polymorphism.
4. Apply templates, file handling, and exception handling.
5. Build efficient and reusable object-oriented programs.

Course Outcomes

CO1: Explain the concepts of object-oriented programming and C++ fundamentals.

CO2: Develop programs using classes, objects, constructors, and inheritance.

CO3: Apply operator overloading, polymorphism, and virtual functions in C++.

CO4: Implement file handling, templates, and stream I/O operations.

CO5: Develop robust applications using exception handling and advanced C++ features.

Course Content/Syllabus:

Unit I: Introduction to Object-Oriented Programming and C++

Introduction to OOP, Basic Concepts of OOP, Applications of OOP, Introduction to C++, C++ Stream I/O, Declarations in C++, Creating New Data Types, Function Prototypes, Inline Functions, Reference Parameters, Const Qualifier, Dynamic Memory Allocation, Default Arguments, Unary Scope Resolution Operator, Linkage Specifications.

Unit II: Classes, Constructors and Friend Functions

Introduction to Classes, Comparing Class with Structure, Class Scope, Accessing Members of a Class, Constructors, Destructors, Const Objects, Const Member Functions, Friend Class, Friend Function, This Pointer, Data Abstraction, Information Hiding, Container Classes, Iterators.

Unit III: Operator Overloading and Inheritance

Operator Overloading: Fundamentals, Restrictions, Overloading Stream Insertion and Extraction Operators, Overloading Unary and Binary Operators, Type Conversion, Overloading ++ and -- Operators.

Inheritance: Protected Members, Base and Derived Classes, Casting Base-Class Pointers to Derived-Class Pointers, Overloading Base-Class Members, Public, Protected and Private Inheritance, Direct and Indirect Base Classes, Constructors and Destructors in Derived Classes, Implicit Derived-to-Base Class Conversion.

Unit IV: Virtual Functions and C++ Stream I/O

Virtual Functions, Type Fields and Switch Statements, Abstract Base Classes, Concrete Classes, Polymorphism, Dynamic Binding, Virtual Destructors.

C++ Stream I/O: Streams, Stream Input and Output, Unformatted I/O, Stream Manipulators, Stream Format States, Stream Error States. File Handling: File Operations, File Pointers, Error Handling during File Operations.

Unit V: Templates and Exception Handling

Templates, Function Templates, Class Templates, Overloading Template Functions, Class Templates with Non-Type Parameters, Templates with Multiple Parameters.

Exception Handling: Basics of C++ Exceptions, Catching Exceptions, Re-throwing Exceptions, Exception Specifications.

References

1. H. M. Deitel and P. J. Deitel, *C++ How to Program*, PHI, 2003.
2. Al Stevens, *C++ Programming*, Wiley Dreamtech, 2003.
3. Herbert Schildt, *The Complete Reference C++*.
4. Tony Gaddis, *Starting Out with Object-Oriented Programming in C++*, Wiley Dreamtech India Pvt. Ltd.

SEMESTER 1

Course: Python Programming: (SET/CSE/MCS/DSE1.2)

Course Objectives:

1. To understand the fundamental concepts and importance of software engineering in the development of high-quality software.
2. To familiarize students with software development life cycle models and requirement engineering.
3. To explore design principles, approaches, and coding strategies used in software development.
4. To study different software testing strategies and quality assurance techniques.
5. To understand project management aspects such as cost estimation, scheduling, risk analysis, and configuration management.
6. To introduce the use and architecture of CASE tools and environments.

Course Content

Unit-1 Introduction to Programming

Problem solving strategies; Structure of a Python program; Syntax and semantics; Executing simple programs in Python.

Unit-2 Creating Python Programs

Identifiers and keywords; Literals, numbers, and strings; Operators; Expressions; Input/output statements; Defining functions; Control structures (conditional statements, loop control statements, break, continue and pass, exit function), default arguments.

Unit-3 Built-in data structures

Mutable and immutable objects; Strings, built-in functions for string, string traversal, string operators and operations; Lists creation, traversal, slicing and splitting operations, passing list to a function; Tuples, sets, dictionaries and their operations.

Unit 4 Object Oriented Programming

Introduction to classes, objects and methods; Standard libraries

Unit 5 File and exception handling

File handling through libraries; Errors and exception handling.

References:

1. Python Programming – S. Taneja
2. Introduction to Computing and Problem-Solving using Python – E. Balaguruswamy

SEMESTER 1

Course: Internet of Things (SET/CSE/MCS/MDE1.1)

Course Objectives

1. Understand the fundamentals and architecture of the Internet of Things (IoT).
2. Learn IoT hardware, software, and communication technologies.
3. Study IoT data acquisition, storage, and cloud integration.
4. Understand IoT security and device management.
5. Explore real-world IoT applications and case studies.

Course Outcomes

CO1: Explain the architecture, components, and applications of IoT.

CO2: Develop IoT solutions using sensors, actuators, and embedded platforms.

CO3: Apply IoT communication protocols and cloud-based data management techniques.

CO4: Analyze IoT security, authentication, and device integration mechanisms.

CO5: Evaluate IoT applications in industrial, healthcare, agriculture, transportation, and smart home domains.

Course Content/Syllabus

Unit I: Introduction to Internet of Things

Introduction, Architectural Overview, Design Principles and Required Capabilities, IoT Applications, Sensing, Actuation, Basics of Networking, M2M and IoT Technology Fundamentals, Devices and Gateways, Data Management, Business Processes in IoT, Everything as a Service (XaaS), Role of Cloud in IoT, Security Aspects in IoT.

Unit II: Elements of IoT

Hardware Components: Computing Platforms (Arduino, Raspberry Pi), Communication, Sensing, Actuation, I/O Interfaces.

Software Components: Programming APIs using Python, Node.js, and Arduino.

Unit III: IoT Communication Protocols

Communication Protocols: MQTT, ZigBee, Bluetooth, CoAP, UDP, TCP.

Unit IV: IoT Solution Framework

Solution Framework for IoT Applications, Device Integration, Data Acquisition and Integration, Device Data Storage, Unstructured Data Storage on Cloud and Local Server, Authentication, Authorization of Devices.

Unit V: IoT Case Studies

Industrial Automation, Transportation, Agriculture, Healthcare, Home Automation.

References

1. Honbo Zhou, *The Internet of Things in the Cloud: A Middleware Perspective*, CRC Press.
2. Vijay Madisetti and Arshdeep Bahga, *Internet of Things: A Hands-on Approach*, University Press.
3. Pethuru Raj and Anupama C. Raman, *The Internet of Things: Enabling Technologies, Platforms, and Use Cases*, CRC Press.

SEMESTER 2

Course: Data Structures Using C (SET/CSE/MCS/DSC201)

Course Objectives

1. Understand fundamental data structures and their applications.
2. Learn techniques for organizing and manipulating data efficiently.
3. Study searching, sorting, and tree-based data structures.
4. Understand file organization and indexing methods.

Course Outcomes

CO1: Explain the concepts and operations of linear and non-linear data structures.

CO2: Implement arrays, stacks, queues, linked lists, and trees for problem solving.

CO3: Apply searching, hashing, and sorting techniques for efficient data management.

CO4: Analyze binary search trees, AVL trees, and B-trees for data organization.

CO5: Describe file organization, indexing, and hashing techniques used in file systems.

Course Content/Syllabus

Unit I: Introduction and Arrays

Introduction: Basic Terminology, Elementary Data Organization, Data Structure Operations, Algorithm Complexity, Time-Space Trade-off.

Arrays: Array Definition, Representation and Analysis, Single and Multidimensional Arrays, Address Calculation, Applications of Arrays, Character Strings in C, Character String Operations, Arrays as Parameters, Ordered Lists, Sparse Matrices, Vectors.

Unit II: Stacks, Recursion and Queues

Stacks: Array Representation and Implementation, Push and Pop Operations, Linked Representation, Applications of Stacks, Conversion of Infix to Prefix and Postfix Expressions, Evaluation of Postfix Expressions. Recursion: Recursive Definition and Processes, Recursion in C, Examples of Recursion, Tower of Hanoi Problem.Queues: Array and Linked Representation, Queue Operations (Create, Add, Delete, Full and Empty), Circular Queue, Dequeue, Priority Queue.

Unit III: Linked Lists and Trees

Linked Lists: Singly Linked List, Two-Way Header List, Traversing and Searching, Overflow and Underflow, Insertion and Deletion, Doubly Linked List, Linked List of Arrays, Polynomial Representation and Addition, Generalized Linked List, Garbage Collection and Compaction.

Trees: Basic Terminology, Binary Trees, Binary Tree Representation, Algebraic Expressions, Complete Binary Trees, Extended Binary Trees, Array and Linked Representation, Tree Traversals, Threaded Binary Trees, Traversing Threaded Binary Trees, Huffman Algorithm.

Unit IV: Searching and Sorting

Searching and Hashing: Sequential Search, Comparison and Analysis, Hash Tables, Hash Functions, Collision Resolution Strategies, Hash Table Implementation. Sorting: Insertion Sort, Bubble Sort, Quick Sort, Two-Way Merge Sort, Heap Sort, Sorting on Different Keys, Practical Considerations for Internal Sorting, Binary Search Trees, AVL Trees, B-Trees.

Unit V: File Structures

Physical Storage Media, File Organization, Organization of Records into Blocks, Sequential Files, Indexing and Hashing, Primary Indices, Secondary Indices.

References

1. Ellis Horowitz and Sartaj Sahni, *Fundamentals of Data Structures*, Galgotia Publications.
2. Richard F. Gilberg and Behrouz A. Forouzan (R. Kruse et al.), *Data Structures and Program Design in C*, Pearson Education.
3. A. M. Tenenbaum et al., *Data Structures and Program Design in C*, Pearson Education.
4. Seymour Lipschutz, *Data Structures*, Tata McGraw-Hill.
5. Kyle Loudon, *Mastering Algorithms with C*, Shroff Publishers and Distributors.

SEMESTER 2

Course: Software Engineering (SET/CSE/MCS/DSC202)

Course Objectives

1. Understand the principles and processes of software engineering.
2. Learn software development life cycle and process models.
3. Study software design, coding, testing, and maintenance techniques.
4. Understand software project management and quality assurance.
5. Explore CASE tools and software reliability concepts.

Course Outcomes

CO1: Explain software engineering principles and software process models.

CO2: Analyze software requirements and apply design methodologies.

CO3: Apply coding standards and software testing techniques.

CO4: Understand software project management, maintenance, and quality assurance practices.

CO5: Describe software reliability models and the role of CASE tools in software development.

Course Content/Syllabus:

Unit I: Introduction to Software Engineering

Introduction to Software Engineering, Importance of Software, Evolving Role of Software, Software Characteristics, Software Components, Software Applications, Software Crisis, Software Engineering Problems, Software Development Life Cycle (SDLC), Software Process.

Unit II: Software Requirement Specification

Analysis, Principles, Waterfall Model, Incremental Model, Prototyping, Spiral Model, Role of Management in Software Development, Role of Metrics and Measurement, Problem Analysis, Requirement Specification, Monitoring and Control.

Unit III: Software Design and Coding

Software Design: Design Principles, Problem Partitioning, Abstraction, Top-Down and Bottom-Up Design, Structured and Object-Oriented Approaches, Design Specification and Verification, Cohesion, Coupling, Functional Independence, Software Architecture, Transaction and Transform Mapping, Component-Level Design, Fourth Generation Techniques. Coding: Top-Down and Bottom-Up Programming, Structured Programming, Information Hiding, Programming Style, Internal Documentation.

Unit IV: Software Testing

Testing Principles, Levels of Testing, Functional Testing, Structural Testing, Test Plan, Test Case Specification, Reliability Assessment, Software Testing Strategies, Verification and Validation, Unit Testing, Integration Testing, Alpha and Beta Testing, System Testing, Debugging.

Unit V: Software Project Management, Reliability and CASE

Software Project Management: Management Spectrum (People, Product, Process, Project), Cost Estimation, Project Scheduling, Staffing, Software Configuration Management, Structured and Unstructured Maintenance, Quality Assurance, Project Monitoring, Risk Management. Software Reliability and Quality Assurance: Reliability Issues, Reliability Metrics, Reliability Growth Models, Software Quality, ISO 9000 Certification, SEI Capability Maturity Model (CMM), Comparison of ISO and SEI CMM. CASE (Computer-Aided Software Engineering): CASE and its Scope, CASE Support in Software Life Cycle, Documentation, Project Management, Internal Interface, Reverse Software Engineering, Architecture of CASE Environment.

References

1. William Stallings, *Computer Organization and Architecture*, Pearson Education Asia.
2. M. Morris Mano, *Computer System Architecture*, PHI.
3. Zaky and Hamacher, *Computer Organization*, McGraw-Hill.
4. B. Ram, *Computer Fundamentals: Architecture and Organization*, New Age International.
5. Andrew S. Tanenbaum, *Structured Computer Organization*, PHI.

SEMESTER 2

Course: Network Security and Cryptography (SET/CSE/MCS/DSC203)

Course Objectives

This course provides an understanding of the fundamentals of network security and cryptography, including encryption techniques, authentication, digital signatures, and network security protocols. It enables students to analyze security threats and apply cryptographic methods to secure communication and protect information systems.

Course Outcomes

1. Explain the fundamentals of cryptography, network security, and security threats.
2. Apply symmetric and public-key encryption techniques for secure communication.
3. Analyze hash functions, message authentication, and digital signature mechanisms.
4. Evaluate authentication protocols and network security services.

Assess system security mechanisms, including firewalls, malware protection, and trusted systems

Course Content/Syllabus

UNIT 1: Introduction of Cryptography: Introduction To security: Attacks, Services and Mechanisms, Security, Attacks, Security Services, Conventional Encryption: Classical Techniques, Conventional Encryption Model, and steganography, Classical Encryption Techniques. Modern Techniques: Simplified DES, Block Cipher Principles, DES Standard, DES Strength, Differential and Linear Cryptanalysis, Block Cipher Design Principles, Block Cipher Modes of Operations.

UNIT 2: Conventional Encryption Algorithms: Triples DES, Blowfish, International Data Encryption Algorithm, RCS, CAST-128, CR2 Placement and Encryption Function, Key Distribution, Random Number Generation, Placement of Encryption Function.

UNIT 3: Public Key Encryption: Public-Key Cryptography: Principles of Public-Key Cryptosystems, RSA Algorithm, Key, Key Management, Fermat's and Euler's Theorm, Primality, Chinese Remainder Theorem. Hash Functions: Message Authentication and Hash Functions: Authentication Requirements, Authentication Functions, Message Authentication Codes, Hash Function Birthday Attacks, Security of Hash Function and MACS, MD5 Message Digest Algorithm, Secure Hash Algorithm (SHA),

UNIT 4: Digital Signatures: Digital Signature, Authentication Protocol, Digital Signature Standard (DDS) Proof of Digital Signature Algorithm.

UNIT 5: Network and System Security: Authentication Applications: Kerberos X-509, Directory Authentication Service, Electronic Mail Security, Pretty Good Privacy (PGP),S/Mine Security: Architecture, Authentication Header, Encapsulating Security Payloads, Combining Security Associations, Key Management, Web Security: Secure Socket Layer and Transport Layer Security, Secure Electronic Transaction (Set), System Security: Intruders, Viruses, Firewall Design Principles, Trusted Systems.

References:

1.William Stallings, "Cryptography and Network Security: Principles and Practice"' Prentice hall, New Jersey

2. Johannes A. Buchmann, "Introduction to Cryptography" Springer-Verlag

3. Atul Kahate, "Cryptography and Network Security" TMH

4. Network Security Bible : Eric Cole, Wiley dreamtech India Pvt. Ltd

5.Practical Cryptography “Bruce Schneier” Wiley dreamtech India Pvt. Ltd.

SEMESTER 2

Course: Database Management System (SET/CSE/MCS/DSC204)

Course Objectives

1. Understand the fundamentals of database management systems.
2. Learn database design using the Entity-Relationship model.
3. Study relational database concepts and SQL programming.
4. Apply normalization techniques for efficient database design.
5. Understand PL/SQL and advanced database concepts.

Course Outcomes

CO1: Explain database concepts, architecture, and data models.

CO2: Design databases using the Entity-Relationship model and relational model.

CO3: Write SQL queries and perform database operations using SQL and PL/SQL.

CO4: Apply normalization techniques to eliminate redundancy and improve database design.

CO5: Analyze database integrity constraints, triggers, and advanced database features.

Course Content/Syllabus

Unit I: Introduction to Database Management System

Introduction to Database Management System, Database System vs File System, Database System Concepts and Architecture, Data Models, Schema and Instances, Data Independence, Database Languages and Interfaces, Data Definition Language (DDL), Data Manipulation Language (DML), Overall Database Structure.

Unit II: Entity Relationship Model

Data Modeling using the Entity Relationship Model, ER Model Concepts, ER Diagram Notations, Mapping Constraints, Keys, Super Key, Candidate Key, Primary Key, Generalization, Aggregation, Reduction of ER Diagrams to Tables, Extended ER Model, Higher Degree Relationships.

Unit III: Relational Data Model and SQL

Relational Data Model Concepts, Integrity Constraints, Entity Integrity, Referential Integrity, Key Constraints, Domain Constraints, Relational Algebra, Relational Calculus, Tuple Calculus, Domain Calculus.

Introduction to SQL: Characteristics of SQL, Advantages of SQL, SQL Data Types and Literals, SQL Commands, SQL Operators, Tables, Views, Indexes, Queries and Subqueries, Aggregate Functions, Insert, Update and Delete Operations, Joins, Union, Intersection, Minus, Cursors in SQL, PL/SQL, Triggers, Clusters.

Unit IV: Database Design and Normalization

Functional Dependencies, First Normal Form (1NF), Second Normal Form (2NF), Third Normal Form (3NF), Boyce-Codd Normal Form (BCNF), Inclusion Dependencies, Lossless Join Decomposition, Normalization using Functional Dependencies (FD), Multivalued Dependencies (MVD), Join Dependencies (JD), Alternative Approaches to Database Design.

Unit V: Advanced Database Concepts

Database Integrity Constraints, Database Design Strategies, Schema Refinement, Database Optimization Concepts, Applications of Database Management Systems.

References

1. C. J. Date, *An Introduction to Database Systems*, Addison-Wesley.
2. Abraham Silberschatz, Henry F. Korth and S. Sudarshan, *Database System Concepts*, McGraw-Hill.
3. V. K. Jain, *Database Management System*, Wiley Dreamtech India Pvt. Ltd.
4. Ramez Elmasri and Shamkant B. Navathe, *Fundamentals of Database Systems*, Addison-Wesley.
5. Paul Beynon-Davies, *Database Systems*, Palgrave Macmillan.
6. Bipin C. Desai, *An Introduction to Database Systems*, Galgotia Publications.
7. Paul Wilton, *Beginning SQL*, Wiley Dreamtech India Pvt. Ltd.

SEMESTER 2

Course: Graphics and Animation (SET/CSE/MCS/DSE2.1)

Course Objectives

1. Understand the fundamentals of computer graphics and animation.
2. Learn graphics primitives, display devices, and input techniques.
3. Study graphical transformations and rendering algorithms.
4. Understand clipping, filling, curves, and surface generation.
5. Explore two-dimensional and three-dimensional graphics techniques.

Course Outcomes

CO1: Explain the fundamentals of computer graphics, display devices, and input devices.

CO2: Apply mathematical concepts and algorithms for graphics generation.

CO3: Implement line drawing, clipping, and polygon filling algorithms.

CO4: Analyze curves, surfaces, and geometric transformations in computer graphics.

CO5: Demonstrate the concepts of 3D graphics, projections, and hidden surface removal.

Course Content/Syllabus

Unit I: Graphics Primitives and Input Techniques

Graphics Primitives: Display Devices – Refresh Cathode Ray Tube (CRT), Raster Scan Display, Plasma Display, Liquid Crystal Display (LCD), Plotters, Printers.

Input Devices: Keyboard, Trackball, Joystick, Mouse, Light Pen, Tablet, Digitizing Camera. Input Techniques: Positioning Techniques, Potentiometers, Constraints, Scales and Guidelines, Rubber-Band Techniques, Dragging, Dimensioning Techniques, Graphical Potentiometers, Pointing and Selection, Boundary Rectangle, Multiple Selections, Menu Selection.

Unit II: Mathematics for Computer Graphics and Line Drawing Algorithms

Mathematics for Computer Graphics: Point Representation, Vector Representation, Matrices and Matrix Operations, Vector Addition, Vector Multiplication, Scalar Product, Vector Product.

Line Drawing Algorithms: DDA Algorithm, Bresenham's Line Drawing Algorithm. Segments and Display Files: Segments, Display File Functions, Posting and Unposting Segments, Segment Naming Schemes, Error Conditions, Appending to Segments, Refresh Concurrent with Reconstruction, Free Storage Allocation, Display File Structure.

Unit III: Graphics Operations

Clipping: Point Clipping, Line Clipping, Polygon Clipping. Filling: Inside Tests, Flood Fill Algorithm, Boundary Fill Algorithm, Scan-Line Polygon Fill Algorithm.

Unit IV: Curves, Surfaces and Transformations

Conics, Curves and Surfaces: Quadric Surfaces (Sphere, Ellipsoid, Torus), Superquadrics, Superellipse, Superellipsoid, Spline Representations, Bezier Curves and Surfaces, Interpolation and Approximation Splines, Parametric Continuity, Geometric Continuity, Spline Specifications.

Transformation: Two-Dimensional Transformations, Basic Transformations, Composite Transformations, Reflection, Shearing, Transformation between Coordinate Systems.

Unit V: Three-Dimensional Graphics

3D Display Methods, Three-Dimensional Transformations, Parallel Projection, Perspective Projection, Visible Line and Surface Identification, Hidden Surface Removal.

References

1. Donald Hearn and M. Pauline Baker, *Computer Graphics*, PHI.
2. Steven Harrington, *Computer Graphics: A Programming Approach*, Tata McGraw-Hill.
3. A. K. Prajapati, *Computer Graphics*, PPM, 2nd Edition.
4. James D. Foley et al., *Computer Graphics: Principles and Practice*, Addison-Wesley, 2nd Edition.
5. Newman and Sproull, *Principles of Interactive Computer Graphics*, McGraw-Hill.
6. David F. Rogers, *Procedural Elements of Computer Graphics*, McGraw-Hill.
7. David F. Rogers and J. Alan Adams, *Mathematical Elements for Computer Graphics*, McGraw-Hill.

SEMESTER 2

Course: Compiler Design (SET/CSE/MCS/DSE2.2)

Course Objectives:

This course introduces the principles and phases of compiler design, including lexical, syntax, and semantic analysis, intermediate code generation, error handling, code optimization, and code generation. It enables students to understand the design and implementation of compilers for translating high-level programming languages into efficient machine code.

Course Outcomes

1. Explain the structure, phases, and components of a compiler.
2. Apply lexical and syntax analysis techniques using parsing methods and compiler generation tools.
3. Generate intermediate code and manage runtime memory for program execution.
4. Analyze and implement error detection and recovery techniques in different compiler phases.
5. Evaluate code optimization and code generation techniques to improve program execution efficiency.

Course Content/Syllabus

Unit 1: Compiler Structure: Compilers and Translators, Various Phases of Compiler, Pass Structure of Compiler, Bootstrapping of Compiler. Programming Language: High level languages, lexical and syntactic structure of a language, Data elements, Data Structure, Operations, Assignments, Program unit, Data Environments, Parameter Transmission. Lexical Analysis: The role of Lexical Analyzer, A Simple approach to the design of Lexical Analyzer, Regular Expressions, Transition Diagrams, Finite state Machines, Implementation of Lexical Analyzer, Lexical Analyzer Generator: LEX, Capabilities of Lexical Analyzer.

Unit 2: The Syntactic Specification of Programming Languages: CFG, Derivation and Parse tree, Ambiguity, Capabilities of EFG. Basic Parsing Techniques: Top-Down parsers with backtracking, Recursive descent Parsers, Predictive Parser, Bottom-up Parsers, Shift-Reduce Parsing, Operator Precedence Parsers, LR parsers (SLR, Canonical LR, LALR) Syntax Analyzer Generator: YACC

Unit 3: Intermediate Code Generation: Different Intermediate forms: Three address code, Quadruples and Triples, Syntax Directed Translation mechanism and attributed definition. Translation of Declaration, Assignment, Control flow, Boolean expression, Array References in arithmetic expressions, procedure calls, case statements, postifix translation. Run Time Memory Management: Static and Dynamic storage allocation, stack based memory allocation schemes, Symbol Table management.

Unit 4: Error Detection and Recovery: Lexical phase errors. Syntactic phase errors, semantic errors.

Unit 5: Code Optimization and Code Generation: Local optimization, Peephole optimization, Basic blocks and flow Graphs, DAG, Data flow analyzer, Machine Model, Order of evaluation, Register allocation and code selection.

References:

1. Alfred V Aho, Jeffrey D. Ullman, "Principles of Compiler Design", Narosa
2. A.V. Aho, R. Sethi and J.D.Ullman, "Compiler Principle, Tech and tools" AW
3. H.C. Holub "Compiler Design in C", Printice Hall Inc.
4. Apple, "Modern Computer Implementation in C: Basic Design" Cambridge Press
5. Modern Compiler Design: Dick Grune, Wiley dreamtech India Pvt. Ltd.
6. Starting Out with Modern Compiler " David Gaddis Wiley dreamtech India Pvt. Ltd.

SEMESTER 2

Course: Quantum Computation (SET/CSE/MCS/MDE2.1)

Course Objectives:

1. Understand the fundamental principles of quantum computation.
2. Learn quantum bits, quantum gates, and quantum circuits.
3. Study quantum information and quantum cryptography.
4. Explore quantum algorithms and computational complexity.
5. Understand quantum error correction and fault-tolerant computation.

Course Outcomes:

CO1: Explain the basic concepts of quantum computation and quantum mechanics.

CO2: Design and analyze simple quantum circuits using quantum gates.

CO3: Apply quantum information and cryptographic principles.

CO4: Analyze quantum algorithms and compare them with classical algorithms.

CO5: Explain quantum error correction techniques and fault-tolerant quantum computation.

Course Content/Syllabus:

Unit I: Introduction to Quantum Computation

Quantum Bits (Qubits), Bloch Sphere Representation of a Qubit, Multiple Qubits, Hilbert Space, Probabilities and Measurements, Entanglement, Density Operators and Correlation, Basics of Quantum Mechanics, Measurements in Bases Other than Computational Basis.

Unit II: Quantum Circuits

Single Qubit Gates, Multiple Qubit Gates, Design of Quantum Circuits.

Unit III: Quantum Information and Cryptography

Comparison between Classical and Quantum Information Theory, Bell States, Quantum Teleportation, Quantum Cryptography, No-Cloning Theorem.

Unit IV: Quantum Algorithms

Classical Computation on Quantum Computers, Relationship between Quantum and Classical Complexity Classes, Deutsch's Algorithm, Deutsch–Jozsa Algorithm, Shor's Factorization Algorithm, Grover's Search Algorithm.

Unit V: Noise and Error Correction

Graph States and Codes, Quantum Error Correction, Fault-Tolerant Computation.

References:

Textbooks

1. Michael A. Nielsen and Isaac L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press.
2. Giuliano Benenti, Giulio Casati, and Giuliano Strini, *Principles of Quantum Computation and Information*, Vol. I: Basic Concepts and Vol. II: Basic Tools and Special Topics, World Scientific.

Reference Books

1. Arthur O. Pittenger, *An Introduction to Quantum Computing Algorithms*.

SEMESTER 3

Course: Design and Analysis of Algorithms (SET/CSE/MCS/DSC301)

Course Objectives:

This course introduces the fundamental concepts of algorithm design and analysis, data structures, and computational complexity. It enables students to design efficient algorithms, analyze their performance, and apply advanced algorithmic techniques to solve complex computational problems.

Course Outcomes

1. Explain the fundamentals of algorithms, asymptotic analysis, and computational complexity.
2. Apply appropriate data structures and sorting techniques to solve computational problems efficiently.
3. Design algorithms using dynamic programming, greedy, backtracking, and branch-and-bound techniques.
4. Analyze and implement graph algorithms for solving real-world problems.
5. Evaluate advanced algorithmic concepts, including randomized algorithms, string matching, NP-completeness, and approximation algorithms.

UNIT 1: Introduction: Algorithms, Analysis of Algorithms, Design of Algorithms, and Complexity of Algorithms, Asymptotic Notations, Growth of function, Recurrences. Sorting in polynomial Time: Insertion sort, Merge sort, Heap sort, and Quick sort Sorting in Linear Time: Counting sort, Radix Sort, Bucket Sort Medians and order statistics.

UNIT 2: Elementary Data Structure: Stacks, Queues, Linked list, Binary Search Tree, Hash Table. Advanced Data Structure: Red Black Trees, Splay Trees, Augmenting Data Structure Binomial Heap, B-Tree, Fibonacci Heap, and Data structure for Disjoint Sets. Union-find Algorithm, Dictionaries and priority Queues, mergeable heaps, concatenable queues.

UNIT 3: Advanced Design and Analysis Techniques: Dynamic Programming, Greedy Algorithm, Backtracking, Branch-and-Bound, Amortized Analysis. Graph Algorithms: Elementary Graph Algorithms, Breadth First search, Depth First search, Minimum Spanning Tree, Kruskal's Algorithms, Prim's Algorithms, Single Source Shortest Path, All pair Shortest Path, Maximum flow and Traveling Salesman Problem. Randomized Algorithms, String Matching, NP-Hard and NP-Completeness Approximation Algorithms, Sorting Network, Matrix Operations, Polynomials and the FFT, Number Theoretic Algorithms.

References:

1. Horowitz Sahani, "Fundamentals of Computer Algorithms." Galgotia
2. Coremen Leiserson etal, "Introduction to Algorithms", PHI
3. Brassard Brately, :Fundamental of Algorithms" PHI
4. M.T. Goodrich etal, "Algorithms Design" John Wiley
5. A.V. Aho etal. "The Design and analysis of Algorithms" Person Education 6. Algorithms and Data Structure: Boldwin Scragg, Wiley dreamtech

SEMESTER 3

Course: Combinatorics and Graph Theory (SET/CSE/MCS/DSC302)

Course Objectives

1. Understand the fundamentals of combinatorics and graph theory.
2. Learn counting techniques, recurrence relations, and generating functions.
3. Study graph properties, trees, and network models.
4. Analyze graph connectivity, planarity, and graph matrices.
5. Apply graph algorithms to solve computational problems

Course Outcomes

CO1: Explain the principles of combinatorics, permutations, combinations, and recurrence relations.

CO2: Analyze graph structures, trees, and spanning trees.

CO3: Apply graph traversal, connectivity, and network flow concepts.

CO4: Evaluate graph coloring, planarity, and graph representations.

CO5: Apply graph-theoretic algorithms to solve real-world and computational problems.

Course Content/Syllabus

Unit I: Combinatorics

Rules of Sum and Product, Permutations, Combinations, Permutation Groups and Applications, Probability, Ramsey Theory, Discrete Numeric Functions, Generating Functions, Combinatorial Problems, Difference Equations.

Unit II: Recurrence Relations

Introduction, Linear Recurrence Relations with Constant Coefficients, Homogeneous Solution, Particular Solution, Total Solution, Solution by the Method of Generating Functions.

Unit III: Graphs and Trees

Graphs, Subgraphs, Basic Properties, Walks, Paths and Circuits, Connected and Disconnected Graphs, Components, Euler and Hamiltonian Graphs, Graph Operations, Trees, Fundamental Circuits, Distance, Diameter, Radius, Pendant Vertices, Rooted Trees, Binary Trees, Counting Trees, Spanning Trees, Finding All Spanning Trees, Weighted Graphs.

Unit IV: Graph Connectivity and Planarity

Cut Sets and Cut Vertices, Fundamental Circuits and Cut Sets, Connectivity and Separability, Network Flows, Planar Graphs, Combinatorial and Geometric Duals, Kuratowski's Theorem, Detection of Planarity, Graph Thickness and Crossings, Vector Space of a Graph, Basis Vectors, Cut Set Vectors, Circuit Vectors, Orthogonal Vectors, Incidence Matrix, Adjacency Matrix.

Unit V: Graph Coloring and Enumeration

Graph Coloring, Covering and Partitioning, Chromatic Number, Chromatic Partitioning, Chromatic Polynomials, Matching, Covering, Four Color Problem, Directed Graphs, Types of Directed Graphs, Directed Paths and Connectedness, Euler Digraphs, Trees and Directed Edges, Fundamental Circuits in Digraphs, Matrices A, B and C of Digraphs, Adjacency Matrix of Digraphs, Graph Enumeration, Counting Labeled and Unlabeled Trees, Polya's Theorem, Graph Enumeration using Polya's Theorem, Graph-Theoretic Algorithms.

References

1. Narsingh Deo, *Graph Theory with Applications to Engineering and Computer Science*, PHI.
2. J. P. Tremblay and R. Manohar, *Discrete Mathematical Structures with Applications to Computer Science*, Tata McGraw-Hill.
3. K. D. Joshi, *Fundamentals of Discrete Mathematics*, New Age International.
4. John Truss, *Discrete Mathematics for Computer Scientists*.
5. C. L. Liu, *Discrete Mathematics*.

SEMESTER 3

Course: Java Programming (SET/CSE/MCS/DSC303)

Course Objectives

1. Understand the fundamentals of Java programming.
2. Learn object-oriented programming concepts using Java.
3. Develop applications using exception handling and multithreading.
4. Understand Java I/O, networking, and GUI programming.
5. Build Java applications using AWT and event handling.

Course Outcomes

CO1: Explain Java programming concepts, syntax, and object-oriented features.

CO2: Develop Java programs using classes, inheritance, packages, and interfaces.

CO3: Apply exception handling and multithreading techniques in Java applications.

CO4: Implement file handling and networking applications using Java APIs.

CO5: Design GUI-based applications using Applets, AWT, and event handling mechanisms.

Course Content/Syllabus

Unit I: Overview of Java

The Genesis of Java, Overview of Java, Java Virtual Machine (JVM), Java Development Kit (JDK), Java vs C++, Data Types, Literals, Variables, Arrays, Operators, Control Statements, Classes, Methods, Nested and Inner Classes, Java.lang Package, String Handling, Constructors, Garbage Collection, Finalize() Method.

Unit II: Inheritance, Packages and Interfaces

Types of Inheritance, Access Specifiers, Class Inheritance, using super, Method Overriding, Abstract Classes, Constructors in Multilevel Inheritance, using final with Inheritance, Dynamic Method Dispatch, Defining Packages, CLASSPATH, Access Protection, Importing Packages, Defining and Implementing Interfaces, Extending Interfaces, Nested Interfaces.

Unit III: Exception Handling and Multithreading

Exception Handling: try, catch, Multiple Catch Blocks, Nested try, throw, throws, finally, Built-in Exceptions, Uncaught Exceptions, User-Defined Exception Classes.

Multithreading: Java Thread Model, Main Thread, Creating Threads, Thread Life Cycle, Thread Priorities, Synchronization, Inter-thread Communication, Suspending, Resuming and Stopping Threads.

Unit IV: Input/Output and Networking

Input/Output: Byte Streams, Character Streams, Predefined Streams, Console Input and Output, PrintWriter Class, Reading and Writing Files.

Networking: Networking Classes and Interfaces, Sockets, TCP/IP Client and Server Sockets, InetAddress, URLConnection, Datagram.

Unit V: Applets, AWT and Event Handling

Applet: Applet Life Cycle, Creating Applets, Using Images and Sound, Passing Parameters.

AWT: Overview of java.awt Package, Components and Containers, Control Components, Layout Managers.

Event Handling: Delegation Event Model, Event Classes, Event Sources, Event Listener Interfaces, Mouse and Keyboard Events, Adapter Classes.

References

1. Patrick Naughton and Herbert Schildt, *Java: The Complete Reference*, Osborne McGraw-Hill, 1997.
2. James R. Levenick, *Simply Java: An Introduction to Java Programming*, Firewall Media.
3. E. Balagurusamy, *Programming with Java*.
4. Rashmi Kanta Das, *Core Java for Beginners*, Vikas Publishing House.

SEMESTER 3

Course: Natural Language Processing (SET/CSE/MCS/DSC304)

Course Objectives:

This course introduces the fundamental concepts of Natural Language Processing (NLP), including language modeling, syntax, semantics, pragmatics, and machine translation. It enables students to develop computational techniques for analyzing, understanding, and generating natural language.

Course Outcomes

1. Explain the fundamentals of natural language processing, regular expressions, automata, and language models.
2. Apply part-of-speech tagging, parsing, and grammar-based techniques for syntactic analysis.
3. Analyze semantic representations and lexical processing for language understanding.
4. Evaluate semantic analysis, word sense disambiguation, and information retrieval techniques.
5. Apply NLP techniques to discourse analysis, conversational agents, natural language generation, and machine translation.

Unit 1: Regular expressions and automata, Morphology and Finite State transducers, N – grams.

Unit 2: Word classes and part of speech tagging, Context free grammars for English, Parsing with context free grammars.

Unit 3: Features and Unifications, Lexicalized and Probabilistic parsing. Semantics: Representing meaning,

Unit 4: Semantic analysis, Lexical semantics, Word Scene Disambiguation and Information retrieval.

Unit 5: Pragmatics: Discourse, Dialog and Conversational Agents, Natural Language Generation, Machine Translation.

References:

1. Daniel, Jurafsky and Martin, Speech and Language Processing, Pearson, 2003.

SEMESTER 3

Course: Computer Vision (SET/CSE/MCS/DSE3.1)

Course Objectives:

This course introduces the fundamentals of image processing and computer vision, including image acquisition, enhancement, feature extraction, object representation, and 3D vision. It enables students to analyze visual data and develop computer vision solutions for real-world applications such as object detection, face recognition, and motion analysis.

Course Outcomes

1. Explain the fundamentals of image processing, image formation, and preprocessing techniques.
2. Apply edge, line, and region-based methods for image analysis and feature extraction.
3. Analyze computer vision techniques for object representation, motion analysis, and 3D vision.
4. Evaluate computer vision algorithms for object detection, tracking, and scene understanding.
5. Develop computer vision solutions for applications such as face recognition, road sign detection, and gait analysis.

Unit 1: Fundamentals of Image Processing: Sources of imagery, Physics of imaging, Representing, acquiring and displaying images, Grayscale, color, noise, lens distortion, and filtering. Preprocessing, and image correction, Enhancing features and correcting imperfections, Addressing noise, lens distortion, and blurring.

Unit 2: Computer Vision Paradigms: Bottom-up, top-down, neural net, feedback, Pixels, lines, boundaries, regions, and object representations, Low-level, intermediate-level and high-level vision, Historical and illustrative examples. Finding Edges and Lines: Finding edges (low-level), Gradients, zero crossing detectors, line models, Roberts, Sobel, Cany, Finding and grouping lines (intermediate-level), Boundary tracing, line fitting, Hough transform.

Unit 3: Region Processing and 3D Vision: Finding elementary regions (low-level), Merging, splitting and grouping regions (intermediate-level), Grouping and analyzing lines and regions (high-level), Stereo and Motion, Optical Flow, Motion Understanding Representing the environment and Matching, Clouds, generalized cylinders, semantic nets, Matching line and region groups to object representations (high-level).

Unit 4: Applications: Face detection, Face recognition, Foreground and background separation, object detection and tracking, combining views from multiple cameras, Identifying Road signs, Gait analysis.

References:

1. Digital Image Processing - R.C. Gonzalez and R.E. Woods, Third Edition, Pearson, 2014

2. Computer & Machine Vision - E. R. Davies, Fourth Edition, Academic Press, 2012

3. Computer Vision - A Modern Approach - D.A. Forsyth and J. Ponce, PHI Pvt. Ltd, 2009

4. Image Processing, Analysis and Machine Vision - M. Sonka, V. Hlavac, and R. Boyle, Thomson Asia Pvt. Ltd., Singapore, 2001

SEMESTER 3

Course: Fuzzy Logic & Neural Networks (SET/CSE/MCS/DSE3.2)

Course Objectives:

This course introduces the principles of fuzzy logic and artificial neural networks for intelligent decision-making and learning. It enables students to understand fuzzy inference systems, neural network models, and hybrid intelligent techniques for solving complex real-world problems.

Course Outcomes

1. Explain the fundamentals of fuzzy logic, fuzzy sets, and fuzzy inference systems.
2. Apply fuzzy reasoning and fuzzy control techniques to decision-making problems.
3. Analyze neural network architectures, learning methods, and training algorithms.
4. Evaluate neural network models and hybrid neuro-fuzzy systems for intelligent applications.
5. Apply fuzzy logic and neural networks to solve problems in pattern recognition, prediction, data mining, and control systems.

Unit 1: Introduction to Fuzzy Logic: Introduction to artificial intelligence and soft computing, conventional logic vs fuzzy logic, uncertainty and imprecision, fundamentals of fuzzy sets, crisp sets and fuzzy sets, membership functions, linguistic variables, fuzzy relations, fuzzy operations, fuzzification and defuzzification methods, applications of fuzzy logic.

Unit 2: Fuzzy Logic Systems and Applications: Fuzzy inference systems, rule-based systems, fuzzy propositions, fuzzy implications, fuzzy reasoning, Mamdani fuzzy model, Sugeno fuzzy model, fuzzy decision making, fuzzy control systems, applications of fuzzy logic in industrial automation, pattern recognition, image processing and expert systems.

Unit 3: Fundamentals of Neural Networks: Introduction to artificial neural networks, biological neural systems, neuron model, architecture of neural networks, characteristics of neural networks, activation functions, single-layer and multilayer neural networks, feedforward and feedback networks, learning methods, supervised, unsupervised and reinforcement learning.

Unit 4: Neural Network Models and Learning Algorithms: Perceptron model, multilayer perceptron, backpropagation learning algorithm, Hebbian learning, competitive learning, associative memory networks, Hopfield network, radial basis function network, self-organizing maps, error correction and performance evaluation of neural networks.

Unit 5: Applications of Fuzzy Logic and Neural Networks: Hybrid neuro-fuzzy systems, adaptive neuro-fuzzy inference system (ANFIS), pattern recognition, speech recognition, image processing, medical diagnosis, data mining, forecasting and prediction systems, robotics and control systems, industrial and real-world applications of fuzzy logic and neural networks.

References:

1. S. Rajasekaran and G.A. Vijayalakshmi Pai – *Neural Networks, Fuzzy Logic and Genetic Algorithms*, PHI Learning.
2. Timothy J. Ross – *Fuzzy Logic with Engineering Applications*, Wiley India.
3. Simon Haykin – *Neural Networks and Learning Machines*, Pearson Education.
4. B. Yegnanarayana – *Artificial Neural Networks*, PHI Learning.
5. J.S.R. Jang, C.T. Sun and E. Mizutani – *Neuro-Fuzzy and Soft Computing*, Pearson Education.

SEMESTER 3

Course: Android Application Development (SET/CSE/MCS/DSE3.3)

Course Objectives:

This course introduces the fundamentals of Android application development, including application architecture, user interface design, data management, networking, and deployment. It enables students to design, develop, test, and deploy Android applications using Android Studio and the Android SDK

Course Outcomes

1. Explain the architecture, components, and development environment of Android applications.
2. Design Android applications using activities, intents, services, and application resources.
3. Develop interactive user interfaces using layouts, animations, and Android UI components.
4. Apply Android APIs for data storage, networking, content sharing, and web services.
5. Test, debug, and deploy Android applications for real-world use.

Unit 1: Introduction to Android: The Android Platform, Android SDK, Android Studio installation, Android Installation, building First Android application, Understanding Anatomy of Android Application, Android Manifest file.

Unit 2: Android Application Design Essentials: Anatomy of an Android applications, Android terminologies, Application Context, Activities, Services, Intents, Receiving and Broadcasting Intents, Android Manifest File and its common settings, Using Intent Filter, Permissions.

Unit 3: Android User Interface Design Essentials: User Interface Screen elements, Designing User Interfaces with Layouts, Drawing and Working with Animation.

Testing Android applications, Publishing Android application, Using Android preferences, Managing Application resources in a hierarchy, working with different types of resources.

Unit 4: Using Common Android APIs: Using Android Data and Storage APIs, managing data using Sqlite, Sharing Data between Applications with Content Providers, Using Android networking APIs, Using Android Web APIs, Using Android Telephony APIs, Deploying Android Application to the World.

References:

1. Meier Reto and Lake Ian, “Professional Android”, Wrox
2. John Horton, Android Programming for Beginners, Packt Publishing

SEMESTER 3

Course: Genetic Algorithm and Application (SET/CSE/MCS/MDE3.1)

Course Objectives:

This course introduces the principles and techniques of genetic algorithms and evolutionary computation for solving optimization problems. It enables students to understand genetic operators, hybrid optimization approaches, and real-world applications of genetic algorithms in machine learning, data mining, engineering, and intelligent systems.

Course Outcomes

1. Explain the fundamentals of genetic algorithms and evolutionary computation.
2. Apply genetic algorithm components, including encoding, selection, crossover, and mutation, to optimization problems.
3. Analyze optimization strategies and constraint-handling techniques using genetic algorithms.
4. Evaluate advanced genetic algorithm variants and hybrid evolutionary approaches.
5. Apply genetic algorithms to solve real-world problems in machine learning, data mining, scheduling, and engineering applications.

Unit 1: Introduction to Genetic Algorithms: Introduction to optimization techniques, conventional optimization vs evolutionary optimization, basics of evolutionary computation, biological inspiration of genetic algorithms, Darwin's theory of evolution, chromosome, gene, population, fitness function, genotype and phenotype, history and development of genetic algorithms, advantages and limitations of genetic algorithms, applications of genetic algorithms.

Unit 2: Genetic Algorithm Fundamentals: Structure and working of genetic algorithm, representation schemes, binary encoding, permutation encoding, value encoding, tree encoding, population initialization, fitness evaluation, parent selection techniques, roulette wheel selection, tournament selection, rank selection, elitism, reproduction strategies, termination criteria.

Unit 3: Genetic Operators and Optimization: Crossover techniques, single point crossover, two point crossover, multipoint crossover, uniform crossover, mutation techniques, bit flip mutation, swap mutation, inversion mutation, adaptive mutation, parameter tuning, convergence of genetic algorithms, optimization problems using genetic algorithms, constraints handling techniques.

Unit 4: Advanced Genetic Algorithms and Hybrid Approaches: Variants of genetic algorithms, adaptive genetic algorithms, parallel genetic algorithms, hybrid genetic algorithms, genetic programming, multi-objective optimization, comparison of genetic algorithms with simulated annealing, particle swarm optimization, neural networks, swarm intelligence approaches.

Unit 5: Applications of Genetic Algorithms: Genetic algorithms in machine learning, feature selection, data mining, scheduling problems, travelling salesman problem, robotics and control systems, network optimization, image processing applications, medical diagnosis systems, financial prediction, real-world case studies and industrial applications.

References:

1. David E. Goldberg – *Genetic Algorithms in Search, Optimization and Machine Learning*, Pearson Education.
2. Melanie Mitchell – *An Introduction to Genetic Algorithms*, MIT Press.
3. S. Rajasekaran and G.A. Vijayalakshmi Pai – *Neural Networks, Fuzzy Logic and Genetic Algorithms*, PHI Learning.
4. A.E. Eiben and J.E. Smith – *Introduction to Evolutionary Computing*, Springer.
5. Zbigniew Michalewicz – *Genetic Algorithms + Data Structures = Evolution Programs*, Springer.

SEMESTER 4

Course: Data Warehousing and Data Mining (SET/CSE/MCS/DSC401)

Course Objectives

This course introduces the concepts of data warehousing and data mining, including data preprocessing, association analysis, classification, clustering, and web mining. It enables students to analyze large datasets and apply data mining techniques to discover meaningful patterns and support decision-making.

Course Outcomes

1. Explain the fundamentals of data warehousing, OLAP, and data mining techniques.
2. Apply data preprocessing and association rule mining techniques to extract useful patterns from data.
3. Analyze classification and clustering algorithms for predictive and descriptive data analysis.
4. Evaluate web mining techniques and data mining methods for complex data sources.
5. Apply data mining techniques to solve real-world problems and support intelligent decision-making.

UNIT 1: Introduction to data mining, need for data warehousing and data mining, application potential, keywords and techniques. Data Warehousing and On line analytical Processing (OLAP): Aggregation operations, models for data warehousing, star schema, fact and dimension tables, conceptualization of data warehouse and multidimensional databases, Relationship between warehouse and mining.

UNIT 2: Data mining primitives: Data preprocessing, data integration, data transformation. Definition and specification of a generic data mining task. Description of Data mining query language with examples. Association analysis: Different methods for mining association rules in transaction based data bases. Illustration of confidence and support. Multidimensional and multilevel association rules. Classification of association rules. Association rule algorithms – A priori and frequent pattern growth.

UNIT 3: Classification and Prediction: Different classification algorithms. Use of genie index, decision tree induction, Bayesian classification, neural network technique of back propagation, fuzzy set theory and genetic algorithms. Clustering: Partition based clustering, hierarchical clustering, model based clustering for continuous and discrete data. Scalability of clustering algorithms. Parallel approaches for clustering.

UNIT 4: Web mining: Web usage mining, web content mining, web log attributes. Data mining issues in object oriented data bases, spatial data bases and multimedia data bases and text data bases.

References:

1. J. Han, M. Kamber, “Data Mining Concepts and Techniques”, Harcourt India Pvt Ltd, 2001.
2. M. Dunham, “ Data Mining : introductory and Advanced Topics”, Pearson Pub, 2003.
3. A.K. Pujari, “Data Mining Techniques”, Universities Press.

SEMESTER 4

Course: E-Commerce (SET/CSE/MCS/DSC402)

Course Objectives:

This course introduces the fundamentals of electronic commerce, including e-commerce technologies, network infrastructure, web security, electronic payment systems, and legal aspects. It enables students to understand secure online business transactions and the technologies that support modern e-commerce applications.

Course Outcomes

1. Explain the fundamentals, architecture, and business applications of electronic commerce.
2. Analyze the network infrastructure and mobile technologies that support e-commerce systems.
3. Apply web security mechanisms and firewall technologies to secure online transactions.
4. Evaluate encryption techniques and digital security methods for electronic communication.
5. Analyze electronic payment systems, e-commerce regulations, and legal issues in digital business.

UNIT 1: Introduction: Electronic Commerce - Technology and Prospects, Definition of E-Commerce, Economic potential of electronic commerce, Incentives for engaging in electronic commerce, forces behind E-Commerce, Advantages and Disadvantages, Architectural framework, Impact of E-Commerce on business.

UNIT 2: Network Infrastructure of E-Commerce: Internet and Intranet based E Commerce Issues, problems and prospects, Network Infrastructure, Network Access Equipments, Broadband telecommunication (ATM, ISDN, FRAME RELAY). Mobile Commerce: Introduction, Wireless Application Protocol, WAP Technology, Mobile Information device, Mobile Computing Applications.

UNIT 3: Web Security: Security Issues on web, Importance of Firewall, components of Firewall, Transaction security, Emerging client server, Security Threats, Network Security, Factors to consider in Firewall design, Limitation of Firewalls.

UNIT 4: Encryption: Encryption techniques, Symmetric Encryption-Keys and data encryption standard, Triple encryption. Asymmetric encryption-Secret key encryption, public and private pair key encryption, Digital Signature, Virtual Private Network.

UNIT 5: Electronic Payments: Overview, The SET protocol, payment Gateway, certificate, digital Tokens, Smart card, credit card, magnetic strip card, E Checks, Credit/Debit card based EPS, online Banking EDI Application in business, E-Commerce Law, Forms of Agreement, Govt. policies and Agenda.

References:

1. Ravi Kalakota, Andrew Winston, :Frontiers of Electronic Commerce" Addison Wesley.
2. Bajaj and Nag. "E-Commerce the cutting edge of Business". TMH.
3. P. Loshin, John Vacca, "Electronic Commerce" Firewall Media, N.Delhi. 4. E Business and Commerce: Brahm Cazner, Wiley dreamtech.

SEMESTER 4

Course: Artificial Intelligence (SET/CSE/MCS/DSE4.1)

Course Objectives:

This course introduces the fundamental concepts of Artificial Intelligence, including knowledge representation, search techniques, reasoning, planning, and intelligent system design. It enables students to understand AI methodologies and apply them to solve real-world problems involving decision-making, perception, and learning.

Course Outcomes

1. Explain the fundamentals, concepts, and applications of Artificial Intelligence.
2. Apply search strategies and heuristic techniques to solve AI problems.
3. Analyze knowledge representation and reasoning using predicate logic and rule-based systems.
4. Evaluate planning and decision-making techniques for intelligent agents and problem-solving.
5. Apply AI techniques in knowledge-based systems, image processing, and natural language processing.

Unit 1 Introduction: Definition and meaning of artificial intelligence, A.I. techniques, pattern recognition, Level of, speech recognition representation in A.I. properties of internal representation.

Unit 2: Production System: Different types of tracing, strategies, graph search strategies, Heuristic graph, search procedure, AND/OR graph, relationship between decomposional and compatible systems, searching Gate Tree, min max search game playing, actual game playing.

Unit 3: Introduction to Predicate Calculus: Predicates and Arguments, connectives, Simplifications of strategies, extracting answers from Resolution Refutation. Control strategies.

Unit 4: Rule Based Deduction Systems: Forward and backward deduction system, resoling with AND/OR graph, computation, deduction and program synthesis, central knowledge for rules based deduct systems. Managing Plans of Action: Plan interpreter, planning decisions, execution monitoring and re-planning domain of application robot motion planning and game playing.

Unit 5: Structural Object Representation: Semantic networks semantic market matching deductive operations on structured objects.

Unit 6: Architectural for A.I. Systems: Knowledge, acquisitions representation IMAGES PROCESSING, Natural language processing.

References:

1. Introduction to artificial Intelligence Eugene Charnik Drew MC mott
2. Artificial Intelligence Elaine Rice.
3. Principal of Artificial Intelligence, Nelson, Springer-Verlag.
4. Artificial Intelligence Application Programming: Tim Jones, Wiley dreamtech.

SEMESTER 4

Course: Image Processing (SET/CSE/MCS/DSE4.2)

Course Objectives:

This course introduces the fundamentals of digital image processing, including image enhancement, restoration, compression, segmentation, and object recognition. It enables students to analyze digital images and apply image processing techniques for solving real-world computer vision and multimedia applications.

Course Outcomes

1. Explain the fundamentals of digital image processing and image acquisition.
2. Apply image enhancement and filtering techniques in spatial and frequency domains.
3. Analyze image restoration and compression methods for efficient image representation.
4. Apply morphological operations and image segmentation techniques for image analysis.
5. Evaluate object representation and recognition techniques for image processing applications.

Unit 1: Processing, Examples of Digital Image Processing application, Fundamental steps in Digital Image processing, Components of Image Processing system Fundamentals: Elements of Visual Perception, Light and Electromagnetic Spectrum, Image Sensing and Acquisition, Image Sampling and Quantization, Some basic Relationships between Pixels, Linear and Nonlinear Operations.

Unit 2: Image Enhancement in the spatial domain: Background, Some basic gray level transformation, Introduction of Histogram processing, Enhancement using Arithmetic/Logic operations, Basics of spatial filtering, Smoothing spatial filters, Sharpening spatial filters, Image Enhancement in the Frequency Domain: Introduction.

Unit 3: Image Restoration: Model of the Image Degradation/Restoration process, Noise Models, Restoration in the presence of noise only spatial filtering, Inverse filtering, Minimum Mean Square Error (Wiener) filtering, Geometric mean filter, Geometric Transformations, Image Compression: Fundamentals, Lossy Compression, Lossless Compression, Image Compression models, Error-free Compression: Variable length coding, LZW coding, Bit plane coding, Run length coding, Introduction to JPEG.

Unit 4: Morphology: Dilation, Erosion, Opening and Closing, Hit-and Miss transform, Morphological Algorithms: Boundry Extraction, Region filling, Extraction of connected components, Convex Hull, Image Segmentation: Definition, characteristics of segmentation Detection of Discontinuities, Edge Linking and Boundary Detection, Thresholding, Region based segmentation. Introduction to Representation and Description, Introduction to Object Recognition.

References

1. Digital Image Processing: Rafael C. Gonzalez and Richard E.Woods. Addision Wesley.
2. Fundamentals of Digital Image Processing. Anil K. Jain, PHI.
3. Digital Image Processing and Analysis: B. Chanda and D. Dutta Majumber, PHI.
4. Image Processing in C: Dwayne Phillips, BPB.

SEMESTER 4

Course: Bioinformatics- (SET/CSE/MCS/MDE4.1)

Course Objectives

This course aims to provide students with a fundamental understanding of biological systems and bioinformatics concepts. It introduces sequence analysis, genomics, proteomics, and computational approaches for analyzing biological data. The course also develops knowledge of molecular modeling, drug design, and bioinformatics tools used in modern biological and pharmaceutical research.

Course Outcomes

1. Explain the structure and function of biological systems and molecular processes.
2. Apply sequence alignment techniques and biological database search tools.
3. Analyze genomic and proteomic data using bioinformatics methods.
4. Demonstrate the principles of molecular modeling and computer-aided drug design.

UNIT 1: Fundamentals of Biological Systems: Introduction to cells: Structure of prokaryotic and eukaryotic cells. Cell organelles and their functions. Molecules of life: Introduction to carbohydrates, proteins, lipids and nucleic acids – Different structural forms and functional organizations. DNA replication, transcription and translation. Gene regulation.

UNIT 2: Sequence Analysis: Introduction to Sequence alignment, Substitution matrices, Scoring matrices –PAM and BLOSUM. Local and Global alignment concepts, dot plot, dynamic programming methodology, Multiple sequence alignment- Progressive alignment. Database searches for homologous sequences – FASTA and BLAST versions.

UNIT 3: Genomics and Proteomics: Functional Genomics: Gene expression analysis by cDNA micro arrays, SAGE, Strategies for generating ESTs and full length inserts; EST clustering and assembly; EST databases- DBEST, UNIGENE Proteomics: Protein and RNA structure prediction, polypeptic composition, secondary and tertiary structure, algorithms for modeling protein folding, structure prediction, proteomics, protein classification, experimental techniques, ligand screening, post-translational modification prediction.

UNIT 4: Computer Aided Drug Design: Introduction to the concepts of molecular modeling. Molecular structure and internal energy. Macromolecular modeling. Design of ligands for known macromolecular target sites. Drug – receptor interactions. Classical SAR/QSAR studies and their implications to the 3-D modeler. Molecular Docking. Structure-based drug design for all classes of targets.

References:

1. Andreqas D. Baxevanis, B. F. Francis Ouellette. Bioinformatics: A Practical Guide to the Analysis of Genes and Proteins John Wiley and Sons, New York (1998).
2. C. Rastogi, Namita Mendiratta, Parag Rastogi. Bioinformatics-concepts, skills, Applications.
3. Bioinformatics Sequence and Genome Analysis. 2001. David W. Mount. Cold Spring Harbor Laboratory Press.
4. Andrew, R. Leach Molecular modeling: Principles and applications Prentice Hall Publications.