Course Structure

&

Detailed Syllabus

MASTER OF COMPUTER APPLICATIONS (MCA)

Academic Regulations-R25

Applicable for the Batches Admitted from 2025-2026

AVANTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Autonomous)

(Approved by A.I.C.T.E., New Delhi & Affiliated to J.N.T.U.H, Hyderabad)

NAAC "A" Accredited Institute

Gunthapally (V),Abdullapurmet(M),R.R(Dist),Near Ramoji film City, Hyderabad, Pin -501512.

www.aietg.ac.in,principalavih@avanthi.edu.in

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

AVANTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

Vision and Mission of the Institute

VISION

To develop highly skilled professionals with ethics & human values.

MISSION

1. To provide high-quality education along with professional training and exposure to the workplace.
2. To encourage a professional mindset that goes beyond academic achievement.
3. To promote holistic education among Department students by means of integrated pedagogy and scholarly mentoring for excellence in both personal and professional domains.
4. To consistently enhance the teaching and learning procedures in order to prepare students for successful careers in business or overseas or in further education.
5. To carefully prepare students to be globally employable professionals who will meet societal demands and contribute to the nation's technological advancement through their research and innovative talents.

QUALITY POLICY

AIET focuses a strong emphasis on the moral principles of delivering cutting edge skilling by establishing the best infrastructure through interactive & activity-based learning. It also strives for an ambitious & effective governance that is responsive in every aspect, and makes use of the latest developments in knowledge and communication technology to encourage students to adopt a global perspective

AVANTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

Program: MASTER OF COMPUTER APPLICATIONS (MCA)

Regulation: R25

Vision and Mission of the Department

DEPARTMENT VISION

To become a center of excellence the computer science and information technology discipline with a strong research and teaching environment.

DEPARTMENT MISSION

1. To provide qualitative education and generate new knowledge by engaging in cutting edge research and by offering state of the art undergraduate, post graduate, leading careers as computer professional in the widely diversified of industry, government and academia.

2. To promote a teaching and learning process that yields advancements instate of art in computer science and engineering in integration of research result and innovative into other scientific discipline leading to new products.

3. To harness human capital for sustainable competitive edge and social relevance by including the philosophy of continuous learning and innovation in computer science and engineering.

4. To provide inculcate team work, imbibe leadership qualities, professional ethics and social responsibilities.

AVANTHI INSTITUTE OF ENGINEERING & TECHNOLOGY

(Autonomous)

(Approved by A.I.C.T.E., New Delhi & Affiliated to J.N.T.U.H, Hyderabad)

NAAC "A" Accredited Institute

Gunthapally (V), Abdullapurmet (M), R.R (Dist), Near Ramoji film City, Hyderabad, Pin -501512.

www.aietg.ac.in, principalavih@avanthi.edu.in

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

Course Structure

Program– PG-Master of Computer Applications

(Applicable from the academic year 2025-2026 to 2028-2029)

Program: PG-**MCA** **Regulation: R25**

I Year I Semester-Course Structure

S.No	Category	Course Code	Course Title	Hours per Week			
				Lecture	Tutorial	Practical	Credits
1	BS&H	R25MCA1101	Mathematical Foundations of Computer Science	3	0	0	3
2	PC	R25MCA1102	C & Data Structures	3	1	0	4
3	PC	R25MCA1103	Python Programming	3	0	0	3
4	PC	R25MCA1104	Operating Systems	3	0	0	3
5	BS&H	R25MCA1105	Accounting and Financial Management	3	0	0	3
6	PC	R25MCA1106	C & Data Structures Lab	0	0	3	1.5
7	PC	R25MCA1107	Python Programming Lab	0	0	2	1
8	PC	R25MCA1108	Operating Systems Lab	0	0	3	1.5
Total				**15**	**1**	**8**	**20**

Category	Courses	Credits
BS&H-Basic Sciences Humanities Course	02	06
PC- Professional Core Course	06	14
Total	08	20

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

Program: PG-MCA **Regulation: R25**

I Year II Semester- Course Structure

S.No	Category	Course Code	Course Title	Hours per Week			
				Lecture	Tutorial	Practical	Credits
1	PC	R25MCA1201	Java Programming	3	0	0	3
2	PC	R25MCA1202	Database Management Systems	3	0	0	3
3	PC	R25MCA1203	Computer Networks	3	0	0	3
4	PC	R25MCA1204	Data Mining	3	0	0	3
5	PC	R25MCA1205	Software Engineering	3	0	0	3
6	PC	25MCA1206	Database Management Systems Lab	0	0	4	2
7	PC	25MCA1207	Computer Networks Lab	0	0	3	1.5
8	PC	25MCA1208	Java Programming Lab	0	0	3	1.5
			Total	**15**	**0**	**10**	**20**

Category	Courses	Credits
PC-Professional Courses	8	20
Total	10	20

Chairperson
Board of Studies (CSE)

MTCS1101 MATHEMATICAL FOUNDATIONS OF COMPUTERSCIENCE 3 0 0 3

(Common to MCA & M. TECH - CSE)

Course Objectives:

1. Introducestheelementarydiscretemathematicsforcomputerscienceandengineering.
2. Topics include formal logic notation,
3. Methods of proof, induction, sets, relations, graph theory,
4. Permutations and combinations, counting principles;
5. Recurrence relations and generating functions.

Course Outcomes:

Course Code	Course Outcomes	PO1	PO2	PO3	DoK
CO1	Apply propositional and predicate logic to construct precise mathematical proofs and logical arguments.	3	3	2	L1, L2, L3
CO2	Use set theory, relations, and functions to model and analyze discrete structures in computing.	3	2	2	L2, L3
CO3	Design and analyze recursive algorithms using mathematical induction and structural recursion.	3	3	3	L2, L3, L4
CO4	Solve discrete probability and advanced counting problems using recurrence relations and generating functions.	3	2	3	L3, L4, L5
CO5	Apply graph theory and tree structures to solve computational problems involving connectivity and optimization.	3	3	3	L3, L4, L5

SYLLABUS

UNIT-I: 8 Hours

The Foundations Logic and Proofs: Propositional Logic, Applications of Propositional Logic, Propositional Equivalence, Predicates and Quantifiers, Nested Quantifiers, Rules of Inference, Introduction to Proofs, Proof Methods and Strategy. **COs: CO1**

UNIT-II: 12 Hours

Basic Structures, Sets, Functions, Sequences, Sums, Matrices and Relations: Sets, Functions, Sequences & Summations, Cardinality of Sets and Matrices Relations, Relations and Their Properties , n-ary Relationsand Their Applications, Representing Relations, Closures of Relations, Equivalence Relations, Partial Orderings. **COs: CO2**

UNIT-III: 10 Hours

Algorithms, Induction and Recursion: Algorithms, The Growth of Functions, Complexity of

Algorithms. Induction and Recursion: Mathematical Induction, Strong Induction and Well-Ordering, Recursive Definitions and Structural Induction, Recursive Algorithms, Program Correctness. **COs: CO3**

UNIT-IV: **12 Hours**

Discrete Probability and Advanced Counting Techniques: An Introduction to Discrete Probability. Probability Theory , Bayes' Theorem, Expected Valueand Variance. Advanced Counting Techniques: Recurrence Relations, Solving Linear Recurrence Relations, Divide-and-Conquer Algorithms and Recurrence Relations, Generating Functions, Inclusion-Exclusion, Applications of Inclusion-Exclusion. **COs: CO4**

UNIT-V: **10 Hours**

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.

Trees: Introduction to Trees, Applications of Trees, Tree Traversal, Spanning Trees, Minimum Spanning Trees. **COs: CO5**

TEXTBOOKS:

1. Discrete Mathematical Structures with Applications to Computer Science: J.P.T remblay, R. Manohar, McGraw-Hill, 1st ed.
2. DiscreteMathematicsforComputerScientists&Mathematicians:Joel.Mott,AbrahamKandel ,Teodore P. Baker, Prentis Hall ofIndia,2nd ed.

REFERENCES:

3. Discrete and Combinatorial Mathematics- an applied introduction : Ralph.P.Grimald,Pearsoneducation, 5th edition.
4. Discrete Mathematical Structures: Thomas Kosy, Tata Mc Graw Hill publishing co.

Web References / eBook Links

1. MIT Open Course Ware – Linear Algebra
2. https://ocw.mit.edu/courses/18-06-linear-algebra-spring-2010/
    a. Free lecture notes, assignments, and video lectures by Prof. Gilbert Strang.
3. Paul's Online Math Notes – Lamar University https://tutorial.math.lamar.edu/
    a. Covers Calculus I, II, III, Linear Algebra with step-by-step notes and examples.
4. NPTEL Courses (IITs – Video Lectures) https://nptel.ac.in/courses/111/106/111106100/
    a. Engineering Mathematics I: Video lectures on Matrices, Calculus, Multivariable Calculus.

5. K.A.Stroud–Engineering Mathematics (eBook)
   https://archive.org/details/EngineeringMathematicsKA.Stroud
   a. A widely used textbook covering matrix algebra, calculus, and more (free on Internet Archive).
6. GeeksforGeeks – Engineering Mathematics Tutorials
   https://www.geeksforgeeks.org/engineering-mathematics/

Concept-wise explanations, coding implementations, and solved problems

Internal Assessment Pattern

Cognitive Level	Internal Assessment #1 (%)	Internal Assessment #2 (%)
L1 – Remembering	20%	10%
L2 – Understanding	30%	20%
L3 – Applying	30%	30%
L4 – Analyzing	15%	25%
L5 – Evaluating	5%	15%
Total (%)	**100%**	**100%**

Sample Short and Long Answer Questions by Cognitive Level

L1: Remembering

1. Define propositional logic. (Short)
2. State the rules of inference. (Short)
3. What is a predicate in logic? (Short)
4. List the properties of relations. (Short)
5. What is mathematical induction? (Short)
6. Define a graph and give an example. (Short)

L2: Understanding

1. Explain propositional equivalence with examples. (Long)
2. Describe the concept of cardinality of sets. (Short)
3. Explain the difference between strong induction and regular induction. (Short)
4. Describe Bayes' Theorem and its significance. (Long)
5. Explain graph isomorphism with a suitable example. (Long)

L3: Applying

1. Construct a truth table for a compound proposition. (Long)
2. Apply inclusion-exclusion principle to solve a counting problem. (Long)
3. Use recursive definitions to define a sequence. (Short)
4. Solve a recurrence relation using generating functions. (Long)
5. Apply Dijkstra's algorithm to find the shortest path in a graph. (Long)

L4: Analyzing

1. Compare propositional logic and predicate logic. (Short)
2. Analyze the closure properties of relations with examples. (Long)
3. Given a recursive algorithm, analyze its correctness. (Short)
4. Discuss the implications of using divide-and-conquer algorithms in recurrence relations. (Long)
5. Analyze the connectivity of a given graph and classify its components. (Long)

L5: Evaluating

1. Justify the use of structural induction in proving properties of recursive algorithms. (Short)
2. Critically evaluate the limitations of discrete probability in modeling real-world uncertainty. (Long)
3. Assess the efficiency of different graph traversal techniques. (Long)
4. Examine the role of equivalence relations in database normalization. (Short)
5. Evaluate the usefulness of spanning trees in network design. (Long)
6. Examine the role of equivalence relations in database normalization. (Short)
7. Evaluate the usefulness of spanning trees in network design. (Long)

Chairperson
Board of Studies (Mathematics)

R25MCA1102 **C & Data Structures** **3 1 0 4**

(Master of Computer Applications)

Course Objectives:

1. Understand the basics of computer systems, environments, and the software development process.
2. Learn the fundamental concepts of C programming, including control statements, functions, arrays, pointers, and strings.
3. Gain knowledge of derived data types, file handling, and program modularization in C.
4. Introduce searching and sorting algorithms with practical implementation.
5. Understand and implement data structures such as linked lists, stacks, and queues.

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	PO4	PO5	PO6	PSO1	PSO2	DoK
CO1	Develop C programs using control statements, operators, functions, arrays, and strings.	3	3	2	2	1	1	3	2	L2, L3
CO2	Implement recursion, pointers, storage classes, and modular programming with pre-processor commands.	3	3	3	2	1	2	3	2	L3, L4
CO3	Apply arrays, pointers, and structures to solve computational problems and perform file operations.	3	3	3	2	2	2	3	3	L3, L4
CO4	Implement searching and sorting algorithms, and perform operations on stacks and queues.	3	3	2	2	2	3	3	3	L3, L4, L5
CO5	Design and analyze solutions using data structures (linked lists, stacks, queues) for real-world applications.	3	3	2	2	2	3	3	3	L4, L5, L6

SYLLABUS

UNIT I: Introduction to Computers & C Language (10 Hours)

Computer Systems, Computing Environments, Computer Languages, Creating and running programs, Software Development Method, Algorithms, Pseudo code, Flowcharts, Applying the software development method.

C Language Basics: Simple programs, Identifiers, Basic data types, Variables, Constants, Input/ Output, Operators, Expressions, Precedence and Associatively, Expression Evaluation, Type conversions, Bitwise operators, Simple C Programming examples. **COs: CO1**

Self-Learning Topics: Write simple C programs; draw flowcharts for algorithms.

UNIT II: Control Structures and Functions (10 Hours)

Statements: if, switch, repetition statements (while, for, do-while), loop control (break, continue, goto).
Functions: basics, user-defined functions, inter-function communication, Recursion, Storage classes (auto, register, static, extern), scope rules, type qualifiers, Pre-processor commands, example programs. **COs: CO2**

Self-Learning Topics: Write recursive programs; practice storage class usage.

UNIT III: Arrays, Strings, and Pointers (12 Hours)

Arrays: one-dimensional, two-dimensional, and multidimensional arrays; applications in C programs.
Strings: string input/output functions, string manipulation, arrays of strings, data conversion.
Pointers: concepts, pointers for inter-function communication, pointers to pointers, arrays of pointers, function pointers, Memory allocation functions, and command-line arguments. **COs: CO3**
Self-Learning Topics: Implement string manipulation functions; practice dynamic memory allocation.

UNIT IV: Derived Types and File Handling (12 Hours)

Structures: declaration, definition, initialization, accessing members, nested structures, arrays of structures, Structures with functions, pointers to structures, self-referential structures, Unions, type def, bit fields, enumerated types.
Files: concept of streams, text and binary files, file input/output operations, error handling functions.

Self-Learning Topics: Write programs for file handling and structures with pointers. **COs: CO4**

UNIT V: Searching, Sorting, and Data Structures (12 Hours)

Sorting: selection sort, bubble sort, insertion sort.
Searching: linear and binary search.
Data Structures: Abstract Data Types, Linear Lists, singly linked list implementation (insertion, deletion, searching).
Stacks: operations, array and linked representations, applications.
Queues: operations, array and linked representations. **COs:CO5**
Self-Learning Topics: Implement stack-based infix to postfix conversion; simulate queue operations.

Board of Studies : Computer Science and Engineering
Approved in BOS No: ___, August, 2024
Approved in ACM No: 01

Text Books

1. C Programming & Data Structures, B.A. Forouzan and R.F. Gilberg, 3rd Edition, Cengage Learning.
2. Problem Solving and Program Design in C, J.R. Hanly and E.B. Koffman, 5th Edition, Pearson Education.
3. The C Programming Language, B.W. Kernighan and Dennis M. Ritchie, PHI/Pearson Education.

References Books

1. C for Engineers and Scientists, H. Cheng, McGraw-Hill International Edition.
2. Data Structures using C, A.M. Tanenbaum, Y. Langsam, and M.J. Augenstein, Pearson Education/PHI.
3. C Programming & Data Structures, P. Dey, M. Ghosh, R. Theraja, Oxford University Press.

Extensive Reading

- https://www.geeksforgeeks.org/c-programming-language/ – Comprehensive C programming tutorials with examples and problem-solving.
- https://www.tutorialspoint.com/cprogramming/ – Beginner-friendly explanations of C concepts with sample programs.
- https://www.javatpoint.com/data-structure-tutorial – Data Structures concepts and algorithms explained with C examples.
- https://www.programiz.com/c-programming – Interactive tutorials and coding practice for C.
- https://en.cppreference.com/w/c – Official reference for C standard library functions.
- https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-087-practical-programming-in-c-january-iap-2010/ – MIT Open Course Ware on Practical Programming in C.
- https://www.cs.cmu.edu/~scandal/nesl/cbook.html – Carnegie Mellon resources on C programming.
- https://www.hackerrank.com/domains/tutorials/10-days-of-c – Practice challenges for C programming.
- https://www.codesdope.com/course/c/ – Step-by-step C tutorials and exercises.

Internal Assessment Pattern

Cognitive Level	Internal Assessment #1(%)	Internal Assessment #2(%)
L1	35	--
L2	40	--
L3	25	20
L4	--	35
L5	--	35
L6	--	10
Total (%)	**100**	**100**

Sample Short and Long Answers questions of Various Cognitive Levels

L1: Remember

1. Define algorithm and pseudocode.
2. List basic data types in C.
3. What is recursion?

4. Define array.
5. What is a pointer?
6. List storage classes in C.
7. Define structure.
8. What is a union?
9. List searching techniques.
10. What is a stack?

L2: Understand

1. Explain the difference between while and do-while loops.
2. Describe scope rules in C.
3. Discuss call by value and call by reference.
4. Explain multidimensional arrays with example.
5. Differentiate between string and array.
6. Explain pointer to pointer concept.
7. Describe text vs binary file differences.
8. Explain linear vs binary search.
9. Describe stack applications.
10. Explain linked list advantages over arrays.

L3: Apply

1. Write a C program to check whether a number is prime.
2. Implement factorial using recursion.
3. Apply string functions to check palindrome.
4. Write a program to find largest element in an array.
5. Apply dynamic memory allocation using malloc().
6. Write a program to read and write a text file.
7. Implement linear search in C.
8. Write a program to sort numbers using bubble sort.
9. Implement stack operations using arrays.
10. Create a queue using linked list.

L4: Analyze

1. Differentiate between call by value and call by reference.
2. Compare static and dynamic memory allocation.
3. Analyze pros and cons of recursion.
4. Compare arrays vs linked lists.
5. Differentiate between stack and queue.

6. Analyze time complexity of bubble sort vs insertion sort.
7. Compare file handling in text vs binary modes.
8. Differentiate structure and union.
9. Compare if-else and switch-case statements.
10. Analyze binary search efficiency.

L5: Evaluate

1. Evaluate effectiveness of different sorting algorithms.
2. Assess the role of storage classes in program efficiency.
3. Critically analyze recursion vs iteration.
4. Evaluate when to use arrays vs linked lists.
5. Assess strengths and weaknesses of stack applications.
6. Evaluate linked list operations in comparison with arrays.
7. Assess binary search applicability in real-life problems.
8. Evaluate struct vs union usage scenarios.
9. Critically analyze file handling methods in large applications.
10. Evaluate pointers' importance in system-level programming.

L6: Create

1. Design a student database using structures and files.
2. Create a text editor using file handling.
3. Build a library management system with linked list.
4. Develop a stack-based expression evaluator.
5. Create a queue-based job scheduling simulation.
6. Design a payroll management system using arrays.
7. Build a phonebook using structures and pointers.
8. Create a recursive solution for Tower of Hanoi.
9. Develop a binary search application for employee records.
10. Design a sorting module to compare algorithms on datasets.

Chairperson
Board of Studies (CSE)

R25MCA1103 **Python Programming** **3 0 0 3**

(Master of Computer Applications)

Course Objectives:

The main objectives of the course are to

1. Learn syntax, semantics, and create functions in Python.
2. Handle strings and files in Python.
3. Understand lists, dictionaries, and regular expressions in Python.
4. Implement object-oriented programming concepts in Python.
5. Build web services and introduce network and database programming in Python.

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	PO4	PO5	PO6	PSO1	PSO2	DoK
CO1	Demonstrate proficiency in handling strings and file systems in Python.	3	3	2	2	1	2	3	2	L2, L3
CO2	Create, run, and manipulate Python programs using lists, dictionaries, and regular expressions	3	3	3	2	1	3	3	2	L3, L4
CO3	Interpret and implement GUI programming concepts in python	3	3	2	2	2	3	3	3	L4, L5
CO4	Implement applications related to network programming, web services, and databases in Python.	3	3	2	2	2	3	3	3	L4, L5, L6
CO5	Apply Python's database connectivity and ORM frameworks to design, implement, and manage relational and NoSQL databases for real-world applications.	3	3	2	1	3	3	3	3	L4,L5, L6

SYLLABUS

UNIT-I **(12 Hours)**

Python Basics, Objects- Python Objects, Standard Types, Other Built-in Types, Internal Types, Standard Type Operators, Standard Type Built-in Functions, Categorizing the Standard Types, Unsupported Types

Numbers - Introduction to Numbers, Integers, Floating Point Real Numbers, Complex Numbers, Operators, Built- in Functions, Related Modules Sequences - Strings, Lists, and Tuples, Mapping and Set Types **COs: CO1**

Self-Learning Topics: Learn about Python data types, operators, and built-in functions with small programs, Practice type conversions and explore mutable vs immutable objects.

UNIT-II **(12 Hours)**

Files: File Objects, File Built-in Function, File Built-in Methods, File Built-in Attributes, Standard Files, Command- line Arguments, File System, File Execution, Persistent Storage Modules, Related Modules

Exceptions: Exceptions in Python, Detecting and Handling Exceptions, Context Management, Exceptions as Strings, Raising Exceptions, Assertions, Standard Exceptions, Creating Exceptions, Exceptions and the sys Module, Related Modules, Modules and Files, Namespaces, Importing Modules, Importing Module Attributes, Module Built-in Functions, Packages, Other Features of Modules. **COs:CO2**

Self-Learning Topics: Understand file modes and practice reading/writing text & binary files, Learn exception handling with try-except and explore modules like os, sys.

UNIT-III **(10 Hours)**

Regular Expressions: Introduction, Special Symbols and Characters, Res and Python Multithreaded Programming: Introduction, Threads and Processes, Python, Threads, and the Global Interpreter Lock, Thread Module, Threading Module, Related Modules **COs: CO3**

Self-Learning Topics: Practice regex for pattern matching like email, phone, and URLs,
Explore Python's threading module and learn process vs thread concepts.

UNIT-IV **(12 Hours)**

GUI Programming: Introduction, Tkinter and Python Programming, Brief Tour of Other GUIs, Related Modules and Other GUIs Web Programming: Introduction, WebSurfing with Python, Creating Simple Web Clients,
Advanced Web Clients, CGI-Helping Servers Process Client Data, Building CGI Application, Advanced CGI, Web (HTTP) Servers **COs:CO4**

Self-Learning Topics: Build simple GUI apps with Tkinter widgets and layouts, Learn to fetch web content with requests and create basic apps with Flask.

UNIT-V **(10 Hours)**

Database Programming: Introduction, Python Database Application Programmer's Interface (DB-API), Object Relational Managers (ORMs), Related Modules **COs: CO5**

Self-Learning Topics: Practice CRUD operations with SQLite/MySQL using Python, Explore ORMs like SQLAlchemy and basics of NoSQL (MongoDB).

Board of Studies : Computer Science and Engineering
Approved in BOS No: ___, August, 2024
Approved in ACM No: 01

Text Book:

1. Core Python Programming, Wesley J. Chun, 2nd Edition, Pearson.

References:

1. Python for Programmers, Paul Deitel, Harvey Deitel, Pearson.
2. Think Python, Allen Downey, Green Tea Press.
3. Introduction to Python, Kenneth A. Lambert, Cengage.
4. Python Programming: A Modern Approach, Vamsi Kurama, Pearson.
5. Learning Python, Mark Lutz, O'Reilly.

Extensive Reading

1. https://docs.python.org/3/
2. https://www.w3schools.com/python/
3. https://www.programiz.com/python-programming
4. https://realpython.com/
5. https://www.geeksforgeeks.org/python-programming-language/

Internal Assessment Pattern

Cognitive Level	Internal Assessment #1(%)	Internal Assessment #2(%)
L1	35	--
L2	40	--
L3	25	20
L4	--	35
L5	--	35
L6	--	10
Total (%)	**100**	**100**

Sample Short and Long Answers questions of Various Cognitive Levels

L1: Remember

1. Define Python objects and standard types.
2. List the different built-in data types in Python.
3. What is a module in Python?
4. Define regular expressions.
5. What is Tkinter?
6. List the different file modes available in Python.
7. Define exceptions in Python and give two examples.
8. What are Python namespaces?
9. List any four built-in functions in Python.
10. Define multithreading in Python.
11. What is the purpose of the import statement in Python?

L2: Understand

1. Explain exception handling with an example in Python.
2. Describe the use of dictionaries in Python.

3. Explain file handling modes with examples.
4. Discuss the concept of namespaces and importing in Python.
5. Explain the working of the threading module.
6. Describe the difference between lists and tuples with examples.
7. Explain recursion in Python with an example.
8. Discuss how Python handles type conversions.
9. Explain the role of the Global Interpreter Lock (GIL).
10. Describe the structure of a Python package with an example.
11. Explain the concept of object-oriented programming in Python.

L3: Apply

1. Write a Python program to count word frequency in a text file.
2. Implement a program to find patterns using regular expressions.
3. Create a simple GUI calculator using Tkinter.
4. Apply database connectivity to insert and fetch student records.
5. Write a Python script for a simple HTTP client.
6. Implement a Python program to check if a string is a palindrome.
7. Write a program to generate Fibonacci numbers using recursion.
8. Create a program to demonstrate multithreading with two threads.
9. Apply file handling to copy contents of one file to another.
10. Write a program to perform matrix multiplication using nested lists.
11. Implement a stack using Python lists.

L4: Analyze

1. Compare Python lists and tuples.
2. Analyze differences between raising exceptions and assertions.
3. Differentiate between single-threading and multithreading in Python.
4. Compare CGI applications with HTTP server programming.
5. Analyze advantages of using ORMs in database programming.
6. Compare Python's try-except vs if-else error handling.
7. Analyze how Python handles memory management compared to C.
8. Differentiate between shallow copy and deep copy in Python.
9. Compare Python's GUI frameworks: Tkinter vs PyQt.
10. Analyze security risks in file handling operations.
11. Differentiate between import and from … import statements.

L5: Evaluate

1. Evaluate the pros and cons of Python's exception handling.

2. Critically analyze the role of the Global Interpreter Lock (GIL).
3. Evaluate the security aspects of web programming in Python.
4. Assess the efficiency of regex in large text processing.
5. Evaluate different GUI frameworks available in Python.
6. Evaluate the advantages of using Python over other scripting languages.
7. Critically analyze the performance of Python in multithreaded programs.
8. Evaluate the effectiveness of ORMs compared to raw SQL queries.
9. Assess the scalability of Python-based web applications.
10. Evaluate the use of Python in real-time systems.
11. Critically assess Python's role in modern data science applications.

L6: Create

1. Design a Python program for student management using files and GUI.
2. Create a multithreaded application for downloading multiple files.
3. Develop a Python-based web client for data scraping.
4. Implement a database-driven web service using Flask/Django.
5. Design a chatbot using Python with regex and database support.
6. Build a file encryption and decryption system using Python.
7. Create a Python GUI app to manage library books.
8. Develop a REST API using Flask with database connectivity.
9. Design a data visualization tool in Python using matplotlib.
10. Create a Python application for employee payroll management.
11. Build a personal expense tracker with file/database storage.

Chairperson
Board of Studies (CSE)

R25MCA1104 **Operating Systems** **3 0 0 3**

(Master of Computer Applications)

Course Objectives:

The main objectives of the course are to

1. Introduce operating system concepts such as processes, threads, scheduling, synchronization, deadlocks, memory management, and file systems.
2. Provide knowledge of operating system design issues and development aspects.
3. Develop practical understanding of Unix commands, system calls, and inter-process communication.

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	PO4	PO5	PO6	PSO1	PSO2	DoK
CO1	Demonstrate understanding of OS structures, processes, and threads..	3	3	2	2	1	2	3	2	L1, L2
CO2	Apply concepts of CPU scheduling, deadlock handling, and system calls.	3	3	3	2	1	3	3	2	L2, L3
CO3	Analyze and implement process synchronization and inter-process communication mechanisms.	3	3	2	2	2	3	3	3	L3, L4
CO4	Evaluate and design solutions for memory management, file systems, and I/O in OS.	3	3	2	2	2	3	3	3	L4, L5, L6
CO5	Gain practical knowledge of how programming languages, operating systems, and architectures interact and how to use each effectively.	3	3	2	2	3	3	2	3	L5, L6

SYLLABUS

UNIT I – Operating System Overview **(12 Hours)**

Operating System - Introduction, Structures - Simple Batch, Multiprogrammed, Time-shared, Personal Computer, Parallel, Distributed Systems, Real-Time Systems, System components, Operating System services, System Calls

Process - Process concepts and scheduling, Operations on processes, Cooperating Processes, Threads **COs: CO1**

UNIT II – CPU Scheduling and Deadlocks **(12 Hours)**

CPU Scheduling - Scheduling Criteria, Scheduling Algorithms, Multiple -Processor Scheduling. System call interface for process management-fork, exit, wait, waitpid, exec

Deadlocks - System Model, Deadlocks Characterization, Methods for Handling Deadlocks, Deadlock Prevention, Deadlock Avoidance, Deadlock Detection, and Recovery from Deadlock **COs: CO2**

UNIT III – Process Management & Synchronization (10 Hours)

Process Management and Synchronization - The Critical Section Problem, Synchronization Hardware, Semaphores, and Classical Problems of Synchronization, Critical Regions, Monitors
Interprocess Communication Mechanisms: IPC between processes on a single computer system, IPC between processes on different systems, using pipes, FIFOs, message queues, shared memory.

COs:CO3

UNIT IV – Memory Management and Virtual Memory (12 Hours)

Memory Management and Virtual Memory - Logical versus Physical Address Space, Swapping, Contiguous Allocation, Paging, Segmentation, Segmentation with Paging, Demand Paging, Page Replacement, Page Replacement Algorithms. **COs:CO4**

UNIT V – File Systems and I/O Management (10 Hours)

File System Interface and Operations -Access methods, Directory Structure, Protection, File System Structure, Allocation methods, Free-space Management. Usage of open, create, read, write, close, lseek, stat, ioctl system calls.

COs: CO5

Board of Studies : Computer Science and Engineering
Approved in BOS No: ___, August, 2024
Approved in ACM No: 01

Text Books

1. Operating System Principles – Abraham Silberchatz, Peter B. Galvin, Greg Gagne, 7th Edition, Wiley.
2. Advanced Programming in the UNIX Environment – W.R. Stevens, Pearson.

References Books

1. Operating Systems – Internals and Design Principles – William Stallings, 5th Edition, Pearson.
2. Operating System: A Design Approach – Crowley, TMH.
3. Modern Operating Systems – Andrew S. Tanenbaum, 2nd Edition, Pearson.
4. UNIX Programming Environment – Kernighan & Pike, PHI.
5. UNIX Internals – The New Frontiers – U. Vahalia, Pearson.

Extensive Reading

1. https://www.tutorialspoint.com/operating_system/index.htm
2. https://www.geeksforgeeks.org/operating-systems/
3. https://www.javatpoint.com/os-tutorial
4. https://www.ibm.com/docs/en/aix
5. https://man7.org/linux/man-pages/

Internal Assessment Pattern

Cognitive Level	Internal Assessment #1(%)	Internal Assessment #2(%)
L1	35	--
L2	40	--
L3	25	20
L4	--	35
L5	--	35
L6	--	10
Total (%)	**100**	**100**

Sample Short and Long Answers questions of Various Cognitive Levels

L1: Remember

1. Define an operating system.
2. List types of operating system structures.
3. What is a process?
4. Define a thread.
5. What is a system call?
6. List four CPU scheduling algorithms.
7. Define deadlock.
8. What are semaphores?
9. List types of memory allocation.
10. What is a file system?

L2: Understand

1. Explain the services provided by an operating system.
2. Describe the process states with a diagram.
3. Explain the concept of multithreading.
4. Discuss prevention vs avoidance in deadlocks.
5. Explain how semaphores solve synchronization problems.
6. Describe paging and segmentation.
7. Explain the working of open() and close() system calls.
8. Discuss protection in file systems.

9. Describe producer-consumer problem.
10. Explain role of scheduling criteria in CPU scheduling.

L3: Apply

1. Write a program using fork() to create a child process.
2. Implement First-Come-First-Served (FCFS) CPU scheduling.
3. Simulate banker's algorithm for deadlock avoidance.
4. Apply semaphores to solve the dining philosophers problem.
5. Write a program to demonstrate message queue IPC.
6. Implement page replacement using FIFO.
7. Apply file system calls to copy contents of one file to another.
8. Write a C program to demonstrate shared memory.
9. Implement Shortest Job Next (SJN) scheduling.
10. Demonstrate use of exec() system call.

L4: Analyze

1. Compare batch, time-sharing, and real-time operating systems.
2. Differentiate between process and thread.
3. Analyze differences between deadlock prevention and detection.
4. Compare FCFS and Round Robin scheduling.
5. Analyze producer-consumer vs readers-writers problem.
6. Differentiate between paging and segmentation.
7. Compare linked and indexed file allocation methods.
8. Analyze security issues in system calls.
9. Differentiate between user-level and kernel-level threads.
10. Compare shared memory and message passing IPC.

L5: Evaluate

1. Evaluate the advantages of multithreading.
2. Assess the efficiency of round robin scheduling for time-sharing systems.
3. Critically analyze the trade-offs in demand paging.
4. Evaluate deadlock handling methods in terms of performance.
5. Assess the role of monitors vs semaphores.
6. Evaluate the strengths and weaknesses of segmentation with paging.
7. Critically analyze journaling file systems.
8. Evaluate advantages of distributed OS over centralized OS.
9. Assess the impact of protection mechanisms on performance.
10. Evaluate real-time operating systems for critical applications.

L6: Create

1. Design a CPU scheduling simulator supporting multiple algorithms.

2. Create a system to demonstrate deadlock detection and recovery.
3. Develop a file management tool using Unix system calls.
4. Implement a virtual memory simulation with page replacement.
5. Design a synchronization library using semaphores.
6. Create a multi-threaded application for producer-consumer problem.
7. Develop an OS mini-project simulating process scheduling.
8. Build a program for directory structure navigation and file handling.
9. Design a deadlock avoidance simulator (banker's algorithm).
10. Create an OS module to demonstrate IPC with shared memory.

Chairperson
Board of Studies (CSE)

R25MCA1105 **Accounting and Financial Management** **3 0 0 3**

(MASTER OF COMPUTER APPLICATIONS)

Course Objectives: To learn

• To learn Financial Management and Accounting

• To learn different types of costing

Course Outcomes: After learning the contents of this paper the student must be able to

• Able to prepare balance sheets of budget.

• Get the skill to manage finances of a firm/company

SYLLABUS

UNIT-I:

Introduction to Accounting: Principles, concepts, conventions, double entry system of accounting, introduction of basic books of accounts ledgers.

Preparation of Trial Balance: Final accounts - company final accounts.

UNIT-II:

Financial Management: Meaning and scope, role, objectives of time value of money - over vitalization - under capitalization - profit maximization - wealth maximization - EPS maximization.

Ratio Analysis: Advantages - limitations - Fund flow analysis - meaning, importance, preparation and interpretation of Funds flow and cash flow statements-statement.

UNIT-III:

Costing: Nature and importance and basic principles. Absorption costing vs. marginal costing - Financial accounting vs. cost accounting vs. management accounting.

Marginal Costing and Break-even Analysis: Nature, scope and importance - practical applications of marginal costing, limitations and importance of cost - volume, profit analysis.

UNIT-IV:

Standard Costing and Budgeting: Nature, scope and computation and analysis - materials variance, labor variance and sales variance - budgeting - cash budget, sales budget - flexible Budgets, master budgets.

UNIT-V:

Introduction to Computerized Accounting System: coding logic and codes, master files, transaction files, introduction documents used for data collection, processing of different files and Outputs obtained.

TEXT BOOKS:

1. Van Horne, James, C: Financial Management and Policy, 12th Edition, Pearson Education.
2. Financial Accounting, S. N. Maheshwari, Sultan Chand Company.
3. Financial Management, S. N. Maheshwari, Sultan Chand Company.

Chairperson
Board of Studies (MBA)

R25MCA1106 **C & Data Structures Lab** **0 0 3 1.5**

(Master of Computer Applications)

Course Objectives:

The main objectives of the course are to

1. To cover concepts of C programming language.
2. To introduce searching and sorting algorithms.
3. To provide an understanding of data structures such as stacks, queues, and linked lists.

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	DoK
CO1	Develop C programs using control statements, arrays, functions, pointers, strings, and data structures.	3	3	3	L1,L2
CO2	Able to create complex programming to solve real time issues	3	3	3	L2,L3
CO3	Implement searching and sorting algorithms efficiently.	3	3	3	L2,L3

Board of Studies : Computer Science and Engineering
Approved in BOS No : ____, August, 2024
Approved in ACM No: 01

SYLLABUS

Developing the following programs:

Experiment 1:

1. Write a C program to find the sum of individual digits of a positive integer.
2. Fibonacci sequence is defined as follows: the first and second terms in the sequence are 0 and 1.
3. Subsequent terms are found by adding the preceding two terms in the sequence. Write a C program to generate the first n terms of the sequence.
4. Write a C program to generate all the prime numbers between 1 and n, where n is a value supplied by the user.
5. Write a C program to find the roots of a quadratic equation.

Experiment 2:

1. Write a C program to find the factorial of a given integer.
2. Write a C program to find the GCD (greatest common divisor) of two given integers.
3. Write a C program to solve Towers of Hanoi problem.
4. Write a C program, which takes two integer operands and one operator from the user, performs the operation and then prints the result. (Consider the operators +,-,*, /, % and use Switch Statement)

Experiment 3:

1. Write a C program to find both the largest and smallest number in a list of integers.
2. Write a C program that uses functions to perform the following:

i) Addition of Two Matrices

ii) Multiplication of Two Matrices

Experiment 4:

1. Write a C program that uses functions to perform the following operations: To insert a sub-string in to a given main string from a given position.
2. To delete n Characters from a given position in a given string.
3. Write a C program to determine if the given string is a palindrome or not
4. Write a C program that displays the position or index in the string S where the string T begins, or – 1 if S doesn't contain T.
5. Write a C program to count the lines, words and characters in a given text.

Experiment 5:

1. Write a C program to generate Pascal's triangle.
2. Write a C program to construct a pyramid of numbers.
3. Write a C program to read in two numbers, x and n, and then compute the sum of this geometric progression: 1+x+x2+x3+.....+xn

   For example: if n is 3 and x is 5, then the program computes 1+5+25+125. Print x, n, the sum Perform error checking. For example, the formula does not make sense for negative exponents – if n is less than
4. Have your program print an error message if n<0, then go back and read in the next pair of numbers of without computing the sum. Are any values of x also illegal ? If so, test for them too.

Experiment 6:

1. 2's complement of a number is obtained by scanning it from right to left and complementing all the bits after the first appearance of a 1. Thus 2's complement of 11100 is 00100. Write a C program to find the 2's complement of a binary number.
2. Write a C program to convert a Roman numeral to its decimal equivalent.

Experiment 7:

1. Write a C program that uses functions to perform the following operations: Reading a complex number
2. Writing a complex number
3. Addition of two complex numbers
4. Multiplication of two complex numbers (Note: represent complex number using a structure.)

Experiment 8:

1. Write a C program which copies one file to another.
2. Write a C program to reverse the first n characters in a file. (Note: The file name and n are specified on the command line.)
3. Write a C program to display the contents of a file.

4. Write a C program to merge two files into a third file (i.e., the contents of the first file followed by those of the second are put in the third file)

Experiment 9:

1. Write a C program that uses functions to perform the following operations on singly linked list.:
   i) Creation ii) Insertion iii) Deletion iv) Traversal

Experiment 10:

1. Write C programs that implement stack (its operations) using

i) Arrays ii) Pointers

2. Write C programs that implement Queue (its operations) using

i) Arrays ii) Pointers

Experiment 11:

1. Write a C program that implements the following sorting methods to sort a given list of integers in ascending order

i) Bubble sort ii) Selection sort

Experiment 12:

1. Write C programs that use both recursive and non recursive functions to perform the following searching operations for a Key value in a given list of integers:

i) Linear search ii) Binary search

TEXT BOOKS:

1. C Programming & Data Structures, B.A.Forouzan and R.F. Gilberg, 3rd Edition, Cengage Learning.
2. Problem Solving and Program Design in C, J.R. Hanly and E.B. Koffman, 5th Edition, Pearson Education.
3. The C Programming Language, B.W. Kernighan and Dennis M.Ritchie, PHI/Pearson Education

REFERENCES:

1. C for Engineers and Scientists, H.Cheng, Mc.Graw-Hill International Edition
2. Data Structures using C – A.M.Tanenbaum, Y.Langsam, and M.J. Augenstein, Pearson Education / PHI
3. C Programming & Data Structures, P. Dey, M Ghosh R Thereja, Oxford University Press

Chairperson
Board of Studies (CSE)

R25MCA1107 **Python Programming Lab** **0 0 2 1**

(Master of Computer Applications)

Course Objectives:

The main objectives of the course are to

1. To install and run the Python interpreter
2. To learn control structures.
3. To Understand Lists, Dictionaries in python
4. To Handle Strings and Files in Python

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	DoK
CO1	Develop the application specific codes using python	3	3	3	L1,L2
CO2	Understand Strings, Lists, Tuples and Dictionaries in Python	3	3	3	L2,L3
CO3	Verify programs using modular approach, file I/O, Python standard library	3	3	3	L2,L3
CO4	Implement Digital Systems using Python	3	3	3	L2,L3

Board of Studies : Computer Science and Engineering
Approved in BOS No : ____, August, 2024
Approved in ACM No: 01

SYLLABUS

Developing the following programs:

Experiment 1:

1. i). Use a web browser to go to the Python website http://python.org. This page contains information about Python and links to Python-related pages, and it gives you the ability to search the Python documentation.
   Start the Python interpreter and type help() to start the online help utility.
2. Start a Python interpreter and use it as a Calculator.
3. Write a program to calculate compound interest when principal, rate and number of periods are given.
4. Given coordinates (x1, y1), (x2, y2) find the distance between two points
5. Read name, address, email and phone number of a person through keyboard and print the details.

Experiment 2:

1. Print the below triangle using for loop.

   5
   4 4
   3 3
   2 2 2 2
   1 1 1 1
2. Write a program to check whether the given input is digit or lowercase character or uppercase character or a special character (use 'if-else-if' ladder)
3. Python Program to Print the Fibonacci sequence using while loop
4. Python program to print all prime numbers in a given interval (use break)

Experiment 3:

1. i) Write a program to convert a list and tuple into arrays.
   ii) Write a program to find common values between two arrays.
2. Write a function called gcd that takes parameters a and b and returns their greatest common divisor.
3. Write a function called palindrome that takes a string argument and returnsTrue if it is a palindrome and False otherwise. Remember that you can use the built-in function len to check the length of a string.

Experiment 4:

1. Write a function called is_sorted that takes a list as a parameter and returns True if the list is sorted in ascending order and False otherwise.
2. Write a function called has_duplicates that takes a list and returns True if there is any element that appears more than once. It should not modify the original list.
   i). Write a function called remove_duplicates that takes a list and returns a new list with only the unique elements from the original. Hint: they don't have to be in the same order.
   ii). The wordlist I provided, words.txt, doesn't contain single letter words. So you might want to add "I", "a", and the empty string.
   iii). Write a python code to read dictionary values from the user. Construct a function to invert its content. i.e., keys should be values and values should be keys.
3. i). Add a comma between the characters. If the given word is 'Apple', it should become 'A,p,p,l,e'
   ii). Remove the given word in all the places in a string?
   iii). Write a function that takes a sentence as an input parameter and replaces the first letter of every word with the corresponding upper case letter and the rest of the letters in the word by corresponding letters in lower case without using a built-in function?
4. Writes a recursive function that generates all binary strings of n-bit length

Experiment 5:

1. i). Write a python program that defines a matrix and prints
   ii). Write a python program to perform addition of two square matrices
   iii). Write a python program to perform multiplication of two square matrices
2. How do you make a module? Give an example of construction of a module using different geometrical shapes and operations on them as its functions.
3. Use the structure of exception handling all general purpose exceptions.

Experiment 6:

1. i). Write a function called draw_rectangle that takes a Canvas and a Rectangle as arguments and draws a representation of the Rectangle on the Canvas.
   ii). Add an attribute named color to your Rectangle objects and modify draw_rectangle so that it uses the color attribute as the fill color.
   iii). Write a function called draw_point that takes a Canvas and a Point as arguments and draws a representation of the Point on the Canvas.
   iv). Define a new class called Circle with appropriate attributes and instantiate a few Circle objects. Write a function called draw_circle that draws circles on the canvas.
2. Write a Python program to demonstrate the usage of Method Resolution Order (MRO) in multiple levels of Inheritances.
3. Write a python code to read a phone number and email-id from the user and validate it for correctness.

Experiment 7:

1. Write a Python code to merge two given file contents into a third file.
2. Write a Python code to open a given file and construct a function to check for given words present in it and display on found.
3. Write a Python code to Read text from a text file, find the word with most number of occurrences
4. Write a function that reads a file *file1* and displays the number of words, number of vowels, blank spaces, lower case letters and uppercase letters.

Experiment 8:

1. Import numpy, Plotpy and Scipy and explore their functionalities.
2. i) Install NumPy package with pip and explore it.
3. Write a program to implement Digital Logic Gates – AND, OR, NOT, EX-OR
4. Write a program to implement Half Adder, Full Adder, and Parallel Adder
5. Write a GUI program to create a window wizard having two text labels, two text fields and two buttons as Submit and Reset.

Text Books:

1. Supercharged Python: Take your code to the next level, Overland
2. Learning Python, Mark Lutz, O'reilly

References:

1. Python Programming: A Modern Approach, Vamsi Kurama, Pearson
2. Python Programming A Modular Approach with Graphics, Database, Mobile, and Web Applications, Sheetal Taneja, Naveen Kumar, Pearson
3. Programming with Python, A User's Book, Michael Dawson, Cengage Learning, India Edition
4. Think Python, Allen Downey, Green Tea Press
5. Core Python Programming, W. Chun, Pearson
6. Introduction to Python, Kenneth A. Lambert, Cengage

Chairperson
Board of Studies (CSE)

R25MCA1108 **Operating Systems Lab** **0 0 3 1.5**

(Master of Computer Applications)

Course Objectives:

1. To provide an understanding of the design aspects of operating system concepts through simulation
2. Introduce basic Unix commands, system call interface for process management, inter process communication and I/O in Unix

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	DoK
CO1	Simulate and implement operating system concepts such as scheduling, deadlock management, file management and memory management.	3	3	3	L1,L2
CO2	Able to implement C programs using Unix system calls	3	3	3	L2,L3
C03	Implement Page Replacement Algorithms	3	3	3	L2,L3

Board of Studies : Computer Science and Engineering
Approved in BOS No : ____, August, 2024
Approved in ACM No: 01

SYLLABUS

Developing the following programs:

Experiment 1:

Write C programs to simulate the following CPU Scheduling algorithms FCFS b) SJF c) Round Robin d) priority

Experiment 2:

Write programs using the I/O system calls of UNIX/LINUX operating system (open, read, write, close, fcntl, seek, stat, opendir, readdir)

Experiment 3:

Write a C program to simulate Bankers Algorithm for Deadlock Avoidance and Prevention.

Experiment 4:

Write a C program to implement the Producer – Consumer problem using semaphores using UNIX/ LINUX system calls.

Experiment 5:

Write C programs to illustrate the following IPC mechanisms Pipes b) FIFOs c) Message Queues d) Shared Memory

Experiment 6:

Write C programs to simulate the following memory management techniques Paging b) Segmentation

Experiment 7:

Write C programs to simulate Page replacement policies FCFS b) LRU c) Optimal

Text Books:

1. Operating System Principles- Abraham Silberchatz, Peter B. Galvin, Greg Gagne 7th Edition, John Wiley
2. Advanced programming in the Unix environment, W.R. Stevens, Pearson education.

References:

1. Operating Systems – Internals and Design Principles, William Stallings, Fifth Edition– 2005, Pearson Education/PHI
2. Operating System - A Design Approach-Crowley, TMH.
3. Modern Operating Systems, Andrew S Tanenbaum, 2nd edition, Pearson/PHI
4. UNIX Programming Environment, Kernighan and Pike, PHI/Pearson Education
5. UNIX Internals: The New Frontiers, U. Vahalia, Pearson Education

Chairperson
Board of Studies (CSE)

R25MCA1201 **Java Programming** **3 0 0 3**

(Master of Computer Applications)

Course Objectives:

1. The main objectives of the course are to
2. Introduce object-oriented programming principles and apply them to problem-solving.
3. Familiarize with the implementation of packages and interfaces.
4. Introduce concepts of exception handling and multithreading.
5. Design Graphical User Interfaces using Swing controls.

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	PO4	PO5	PO6	PSO1	PSO2	DoK
CO1	Apply OOP concepts like encapsulation, inheritance, polymorphism, and abstraction to solve real-world problems.	3	3	2	2	1	2	3	2	L2, L3
CO2	Implement exception handling and file I/O operations using Java libraries.	3	3	3	2	1	3	3	2	L3, L4
CO3	Utilize packages, collections, and utility classes to manage and process data.	3	3	2	2	2	3	3	3	L4, L5
CO4	Develop multithreaded applications and connect to databases using JDBC.	3	3	2	2	2	3	3		L4, L5, L6
CO5	Design interactive GUI-based applications using Swing and event handling.	3	3	3	2	2	3	3	3	L5, L6

SYLLABUS

UNIT -I **(12 Hours)**

Foundations of Java: History of Java, Java Features, Variables, Data Types, Operators, Expressions, Control Statements. Elements of Java - Class, Object, Methods, Constructors and Access Modifiers, Generics, Inner classes, String class and Annotations.

OOP Principles: Encapsulation – concept, setter and getter method usage, this keyword. Inheritance - concept, Inheritance Types, super keyword. Polymorphism – concept, Method Overriding usage and Type Casting. Abstraction – concept, abstract keyword and Interface.

COs: CO1

Self-Learning Topics: Write programs for inheritance and polymorphism; practice String operations.

UNIT-II **(10 Hours)**

Exception Handling: Exception and Error, Exception Types, Exception Handler, Exception Handling Clauses – try, catch, finally, throws and the throw statement, Built-in-Exceptions and Custom Exceptions.

Files and I/O Streams: The file class, Streams, The Byte Streams, Filtered Byte Streams, The Random Access File class. **COs: CO2**

Self-Learning Topics: Implement custom exceptions, create file read/write programs.

UNIT-III **(12 Hours)**

Packages- Defining a Package, CLASSPATH, Access Specifiers, importing packages. Few Utility Classes - String Tokenizer, BitSet, Date, Calendar, Random, Formatter, Scanner.

Collections: Collections overview, Collection Interfaces, Collections Implementation Classes, Sorting in Collections, Comparable and Comparator Interfaces. **COs: CO3**

Self-Learning Topics: Practice Java Collections (ArrayList, HashMap); implement Comparable and Comparator.

UNIT-IV **(12 Hours)**

Multithreading: Process and Thread, Differences between thread-based multitasking and process-based multitasking, Java thread life cycle, creating threads, thread priorities, synchronizing threads, inter thread communication.

Java Database Connectivity: Types of Drivers, JDBC architecture, JDBC Classes and Interfaces, Basic steps in Developing JDBC Application, Creating a New Database and Table with JDBC. **COs: CO4**

Self-Learning Topics: Write multithreaded programs, build a simple JDBC application to insert/fetch records.

UNIT-V **(12 Hours)**

GUI Programming with Swing – Introduction, limitations of AWT, MVC architecture, components, containers, Layout Manager Classes, Simple Applications using AWT and Swing.

Event Handling- The Delegation event model- Events, Event sources, Event Listeners, Event classes, Handling mouse and keyboard events, Adapter classes. **COs: CO5**

Self-Learning Topics: Create a simple Swing calculator; implement mouse/keyboard event programs.

Board of Studies : Computer Science and Engineering
Approved in BOS No : ____, August, 2024
Approved in ACM No: 01

Text Books:

1. Java The complete reference, 9th edition, Herbert Schildt, McGraw Hill Education (India) Pvt. Ltd.
2. Understanding Object-Oriented Programming with Java, updated edition, T. Budd, Pearson Education.

References Books:

1. An Introduction to programming and OO design using Java, J. Nino and F.A. Hosch, John Wiley & sons
2. Introduction to Java programming, Y. Daniel Liang, Pearson Education.
3. Object Oriented Programming through Java, P. Radha Krishna, University Press.
4. Programming in Java, S. Malhotra, S. Chudhary, 2nd edition, Oxford Univ. Press.
5. Java Programming and Object-oriented Application Development, R. A. Johnson, Cengage Learning.

Extensive Reading

1. https://www.javatpoint.com/java-tutorial
2. https://www.geeksforgeeks.org/java/
3. https://docs.oracle.com/javase/tutorial/
4. https://www.programiz.com/java-programming
5. https://www.tutorialspoint.com/java/

Internal Assessment Pattern

Cognitive Level	Internal Assessment #1(%)	Internal Assessment #2(%)
L1	35	--
L2	40	--
L3	25	20
L4	--	35
L5	--	35
L6	--	10
Total (%)	**100**	**100**

Sample Short and Long Answers questions of Various Cognitive Levels

L1: Remember

1. List features of Java.
2. Define encapsulation.
3. What is a constructor in Java?
4. Define exception.
5. List any four collection classes.
6. What is multithreading?
7. Define JDBC driver.
8. What is Swing?
9. What is the difference between *throw* and *throws*?
10. Define interface in Java.

L2: Understand

1. Explain the concept of inheritance with an example.
2. Describe different types of exceptions in Java.
3. Discuss the role of access specifiers in packages.
4. Explain difference between Comparable and Comparator.
5. Describe thread lifecycle with a diagram.
6. Explain how synchronization works in Java.
7. Describe MVC architecture in Swing.

8. Explain difference between AWT and Swing.
9. Explain try-catch-finally with an example.
10. Describe JDBC architecture.

L3: Apply

1. Write a program to demonstrate method overriding.
2. Implement a custom exception in Java.
3. Apply StringTokenizer to split a sentence.
4. Implement ArrayList sorting using Comparable.
5. Write a program to create multiple threads.
6. Apply JDBC to insert a student record in database.
7. Create a Swing-based calculator application.
8. Write a program to read and write to a file.
9. Apply Comparator to sort employee objects.
10. Create a program to handle mouse events.

L4: Analyze

1. Differentiate between method overloading and overriding.
2. Compare checked and unchecked exceptions.
3. Analyze differences between HashMap and TreeMap.
4. Compare process-based and thread-based multitasking.
5. Differentiate between file byte stream and character stream.
6. Compare abstract classes and interfaces.
7. Analyze JDBC driver types.
8. Differentiate between FlowLayout and BorderLayout.
9. Compare event handling in AWT and Swing.
10. Analyze use of synchronization vs inter-thread communication.

L5: Evaluate

1. Evaluate the benefits of OOP principles.
2. Assess exception handling mechanisms in large-scale applications.
3. Critically analyze advantages of Java Collections framework.
4. Evaluate performance trade-offs of multithreading.
5. Assess JDBC architecture for enterprise applications.
6. Evaluate strengths and limitations of Swing compared to JavaFX.
7. Critically analyze use of abstract classes in design.
8. Evaluate role of garbage collection in Java applications.
9. Assess security aspects of JDBC applications.
10. Evaluate use of interfaces in real-world Java design.

L6: Create

1. Design a payroll system using OOP concepts.
2. Create a custom exception hierarchy for banking applications.
3. Develop a student management system using Java Collections.
4. Create a multithreaded chat application.
5. Build a JDBC-based inventory management system.

6. Design a Swing application for online quiz.
7. Develop a GUI-based notepad editor with menus.
8. Create a database-driven library management system.
9. Build a multithreaded downloader using Java.
10. Design a graphical calculator using Swing.

Chairperson
Board of Studies (CSE)

R25MCA1202 **Database Management Systems** **3 0 0 3**

(Master of Computer Applications)

Course Objectives:

The main objectives of the course are to

1. To understand the basic concepts and the applications of database systems.
2. To master the basics of SQL and construct queries using SQL.
3. To cover data models, database design, relational model, relational algebra, transaction control, concurrency control, storage structures, and access techniques.

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	PO4	PO5	PO6	PSO1	PSO2	DoK
CO1	Explain DBMS concepts, architecture, and ER modeling.	3	3	2	2	1	2	3	2	L2, L3
CO2	Apply relational algebra and relational calculus for query formulation.	3	3	3	2	1	3	3	2	L3, L4
CO3	Construct SQL queries, constraints, and triggers; apply normalization.	3	3	2	2	2	3	3	3	L4, L5
CO4	Demonstrate transaction concepts, concurrency, and recovery mechanisms.	3	3	2	2	2	3	3		L4, L5, L6
CO5	Analyze file organization, indexing techniques, and B+ Trees.	3	3	3	2	2	3	3	3	L5, L6

SYLLABUS

UNIT-I **(12 Hours)**

Database System Applications: A Historical Perspective, File Systems versus a DBMS, the Data Model, Levels of Abstraction in a DBMS, Data Independence, Structure of a DBMS

Introduction to Database Design: Database Design and ER Diagrams, Entities, Attributes, and Entity Sets, Relationships and Relationship Sets, Additional Features of the ER Model, Conceptual Design With the ER Model **COs: CO1**

Self-Learning Points: Draw ER diagrams for real-life applications like library, hospital.

UNIT-II **(10 Hours)**

Introduction to the Relational Model: Integrity constraint over relations, enforcing integrity constraints, querying relational data, logical database design, introduction to views, destroying/altering tables and views. Relational Algebra, Tuple relational Calculus, Domain relational calculus. **COs: CO2**

Self-Learning Points: Practice writing queries using relational algebra.

UNIT-III **(12 Hours)**

SQL: QUERIES, CONSTRAINTS, TRIGGERS: form of basic SQL query, UNION, INTERSECT,

and EXCEPT, Nested Queries, aggregation operators, NULL values, complex integrity constraints in SQL, triggers and active databases.

Schema Refinement: Problems caused by redundancy, decompositions, problems related to decomposition, reasoning about functional dependencies, FIRST, SECOND, THIRD normal forms, BCNF, lossless join decomposition, multivalued dependencies, FOURTH normal form, FIFTH normal form. **COs: CO3**

Self-Learning Points: Normalize a student database up to BCNF.

UNIT-IV **(12 Hours)**

Transaction Concept, Transaction State, Implementation of Atomicity and Durability, Concurrent Executions, Serializability, Recoverability, Implementation of Isolation, Testing for serializability, Lock Based Protocols, Timestamp Based Protocols, Validation- Based Protocols, Multiple Granularity, Recovery and Atomicity, Log– Based Recovery, Recovery with Concurrent Transactions. **COs: CO4**

Self-Learning Points: Study examples of deadlocks and how to prevent them.

UNIT-V **(12 Hours)**

Data on External Storage, File Organization and Indexing, Cluster Indexes, Primary and Secondary Indexes, Index data Structures, Hash Based Indexing, Tree based Indexing, Comparison of File Organizations, Indexes- Intuitions for tree Indexes, Indexed Sequential Access Methods (ISAM), B+ Trees: A Dynamic Index Structure. **COs: CO5**

Self-Learning Points: Implement B+ Tree insertion with example data.

Board of Studies : Computer Science and Engineering
Approved in BOS No : ____, August, 2024
Approved in ACM No: 01

Text Books

1. Database System Concepts, Silberschatz, Korth, McGraw hill, V edition.3rd Edition
2. Database Management Systems, Raghurama Krishnan, Johannes Gehrke, Tata Mc Graw Hill

References

1. Database Systems: Design, Implementation, and Management – Peter Rob, Carlos Coronel, 7th Ed.
2. Fundamentals of Database Systems – Elmasri & Navathe, Pearson.
3. Introduction to Database Systems – C.J. Date, Pearson.
4. Oracle for Professionals – S. Shah, V. Shah, SPD.
5. Database Systems Using Oracle: A Simplified Guide to SQL & PL/SQL – Shah, PHI.
6. Fundamentals of Database Management Systems – M.L. Gillenson, Wiley.

Extensive Reading

- https://www.geeksforgeeks.org/dbms/
- https://www.javatpoint.com/dbms-tutorial
- https://www.tutorialspoint.com/dbms/
- https://www.programiz.com/sql
- https://www.sqltutorial.org/

Internal Assessment Pattern

Cognitive Level	Internal Assessment #1(%)	Internal Assessment #2(%)
L1	35	--
L2	40	--
L3	25	20
L4	--	35
L5	--	35
L6	--	10
Total (%)	**100**	**100**

Sample Short and Long Answers questions of Various Cognitive Levels

L1: Remember

1. Define DBMS and its advantages over file systems.
2. What is data independence?
3. List different data models.
4. Define entity and attribute.
5. What is a primary key?
6. Define functional dependency.
7. What is a transaction?
8. Define serializability.
9. What is an index?
10. Define B+ Tree.

L2: Understand

1. Explain levels of abstraction in DBMS.
2. Describe ER diagram with example.
3. Explain referential integrity constraint.
4. Differentiate between domain relational calculus and tuple relational calculus.
5. Explain triggers in SQL with an example.
6. Explain BCNF with suitable example.
7. Describe transaction states.

8. Differentiate between lock-based and timestamp-based protocols.
9. Explain file organization methods.
10. Describe advantages of B+ Trees.

L3: Apply

1. Construct an ER diagram for hospital management system.
2. Write SQL queries for student database (insert, update, delete, select).
3. Apply nested query to find top 3 students by marks.
4. Write an SQL trigger to check salary > 0 before insertion.
5. Normalize employee database to 3NF.
6. Write SQL query using UNION and INTERSECT.
7. Apply log-based recovery on a sample transaction.
8. Write program to implement 2-phase locking.
9. Construct a B+ Tree for given keys.
10. Apply hashing for fast searching.

L4: Analyze

1. Compare DBMS vs File systems.
2. Analyze functional dependency violations in a schema.
3. Compare relational algebra and SQL.
4. Analyze differences between views and tables.
5. Compare BCNF and 3NF.
6. Analyze deadlock situations in DBMS.
7. Differentiate between recovery with and without checkpoints.
8. Analyze ISAM vs B+ Tree indexing.
9. Compare tree-based vs hash-based indexing.
10. Differentiate between optimistic and pessimistic concurrency control.

L5: Evaluate

1. Evaluate advantages of normalization in DBMS.
2. Critically assess use of triggers in databases.
3. Evaluate timestamp protocols against lock-based protocols.
4. Assess strengths and weaknesses of relational calculus.
5. Evaluate the effectiveness of B+ Trees in indexing.
6. Assess the role of transaction management in banking systems.
7. Evaluate schema decomposition methods.

8. Critically analyze log-based recovery methods.
9. Evaluate performance of clustered vs unclustered indexes.
10. Assess query optimization in relational databases.

L6: Create

1. Design a normalized database for a university system.
2. Create an SQL trigger to maintain audit logs.
3. Develop a payroll management system using SQL.
4. Design a library management system ER model.
5. Create a database schema for online shopping.
6. Develop a transaction schedule demonstrating deadlock.
7. Create a B+ Tree implementation for a file system.
8. Design a hotel booking system database.
9. Build a recovery algorithm for transaction failure.
10. Create an airline reservation system with indexing.

Chairperson
Board of Studies (CSE)

R25MCA1203 **Computer Networks** **3 0 0 3**

(Master of Computer Applications)

Course Objectives:

The main objectives of the course are to

1. To equip students with an overview of the concepts and fundamentals of computer networks.
2. To familiarize students with layered communication models (OSI and TCP/IP) and the protocols of each layer.

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	PO4	PO5	PO6	PSO1	PSO2	DoK
C01	Explain basic concepts, hardware/software, and reference models of computer networks.	3	3	2	2	1	2	3	2	L2, L3
C02	Demonstrate error control, flow control, and medium access control protocols.	3	3	3	2	1	3	3	2	L3, L4
C03	Apply routing and congestion control algorithms in networks.	3	3	2	2	2	3	3	3	L4, L5
C04	Analyze and compare transport layer services (TCP vs UDP) and connection management.	3	3	2	2	2	3	3		L4, L5, L6
C05	Evaluate and design application layer services such as DNS, Email, and HTTP.	3	3	3	2	2	3	3	3	L5, L6

SYLLABUS

UNIT-I **(12 Hours)**

Network hardware, Network software, OSI, TCP/IP Reference models, Example Networks: ARPANET, Internet. Physical Layer: Guided Transmission media: twisted pairs, coaxial cable, fiber optics, Wireless Transmission. **COs:**

CO1

Data link layer: Design issues, framing, Error detection and correction.

Self-Learning Points: Study OSI vs TCP/IP models with real-world protocol examples.

UNIT-II **(10 Hours)**

Elementary data link protocols: simplex protocol, A simplex stop and wait protocol for an error-free channel, A simplex stop and wait protocol for noisy channel.

Sliding Window protocols: A one-bit sliding window protocol, A protocol using Go-Back-N, A protocol using Selective Repeat, Example data link protocols.

Medium Access sublayer: The channel allocation problem, Multiple access protocols: ALOHA, Carrier sense multiple access protocols, collision free protocols. Wireless LANs, Data link layer switching. **COs: CO2**

Self-Learning Points: Simulate Stop-and-Wait and Go-Back-N in C/Java/Python.

UNIT-III **(12 Hours)**

Network Layer: Design issues, Routing algorithms: shortest path routing, Flooding, Hierarchical routing, Broadcast, Multicast, distance vector routing, Congestion Control Algorithms, Quality of Service, Internetworking, The Network layer in the internet. **COs: CO3**

Self-Learning Points: Implement Dijkstra's algorithm for shortest path routing.

UNIT-IV **(12 Hours)**

Transport Layer: Transport Services, Elements of Transport protocols, Connection management, TCP and UDP protocols. **COs: CO4**

Self-Learning Points: Study TCP 3-way handshake with Wireshark capture.

UNIT-V **(12 Hours)**

Application Layer –Domain name system, SNMP, Electronic Mail; the World WEB, HTTP, Streaming audio and video. **COs: CO5**

Self-Learning Points: Explore HTTP request/response using curl or Postman.

Board of Studies : Computer Science and Engineering

Approved in BOS No : ____, August, 2024

Approved in ACM No: 01

Text Book

1. Computer Networks – Andrew S. Tanenbaum, David J. Wetherall, 5th Edition, Pearson/PHI.

References

1. An Engineering Approach to Computer Networks – S. Keshav, 2nd Edition, Pearson.
2. Data Communications and Networking – Behrouz A. Forouzan, 3rd Edition, TMH.

Extensive Reading

1. https://www.geeksforgeeks.org/computer-network-tutorials/
2. https://www.tutorialspoint.com/computer_fundamentals/computer_networking.htm
3. https://www.javatpoint.com/computer-network-tutorial
4. RFC 791 (IPv4), RFC 2460 (IPv6), RFC 793 (TCP), RFC 768 (UDP).

Internal Assessment Pattern

Cognitive Level	Internal Assessment #1(%)	Internal Assessment #2(%)
L1	35	--
L2	40	--
L3	25	20
L4	--	35
L5	--	35
L6	--	10
Total (%)	**100**	**100**

Sample Short and Long Answers questions of Various Cognitive Levels

L1: Remember

1. Define OSI reference model.
2. What are guided transmission media?
3. List functions of data link layer.
4. What is framing?
5. Define ARPANET.
6. What is error detection?
7. List sliding window protocols.
8. Define routing.
9. What is TCP?
10. Define DNS.

L2: Understand

1. Explain OSI vs TCP/IP models.
2. Describe stop-and-wait protocol.
3. Explain Go-Back-N ARQ.
4. Explain CSMA with collision detection.
5. Differentiate between unicast, multicast, and broadcast routing.
6. Explain congestion control mechanisms.
7. Explain transport layer services.
8. Explain TCP vs UDP.
9. Describe HTTP working.
10. Explain working of DNS.

L3: Apply

1. Apply CRC for error detection on given data.
2. Write an example of Selective Repeat protocol.
3. Apply Dijkstra's algorithm to find shortest path.
4. Demonstrate subnetting for a given IP address.

5. Apply TCP 3-way handshake on a sample sequence.
6. Construct a Go-Back-N sliding window example.
7. Apply congestion control using Leaky Bucket algorithm.
8. Write simple code to send HTTP GET request.
9. Apply SNMP for network device monitoring.
10. Demonstrate routing using distance vector algorithm.

L4: Analyze

1. Compare OSI vs TCP/IP reference models.
2. Analyze efficiency of ALOHA protocol.
3. Differentiate Go-Back-N vs Selective Repeat.
4. Analyze the effect of congestion on QoS.
5. Compare IPv4 vs IPv6.
6. Analyze TCP vs UDP for video streaming.
7. Compare different congestion control algorithms.
8. Analyze role of connection management in TCP.
9. Compare DNS and HTTP protocols.
10. Differentiate between circuit switching and packet switching.

L5: Evaluate

1. Evaluate strengths and weaknesses of error detection methods.
2. Critically evaluate efficiency of CSMA/CD.
3. Assess the performance of routing algorithms.
4. Evaluate congestion control techniques in TCP.
5. Assess suitability of UDP for VoIP.
6. Evaluate SNMP's effectiveness for large networks.
7. Critically analyze TCP's sliding window mechanism.
8. Evaluate scalability of hierarchical routing.
9. Assess advantages of IPv6 over IPv4.
10. Evaluate HTTP protocol in modern web applications.

L6: Create

1. Design a LAN for a university campus using subnetting.
2. Create a simulation of Go-Back-N ARQ protocol.
3. Develop a routing table using distance vector algorithm.
4. Design a congestion control scheme for a new application.
5. Build a client-server application using TCP sockets.
6. Create a DNS hierarchy for a college network.
7. Develop a simple mail client using SMTP/POP.
8. Create a network topology with switches, routers, and hosts.
9. Build an HTTP server handling GET/POST requests.
10. Design a multimedia streaming service with QoS.

Chairperson
Board of Studies (CSE)

R25MCA1204 **Data Mining** **3 0 0 3**

(Master of Computer Applications)

Course Objectives:

The main objectives of the course are to

1. Learn data mining concepts understand association rules mining.
2. Discuss classification algorithms learn how data is grouped using clustering techniques.
3. To develop the abilities of critical analysis to data mining systems and applications.
4. To implement practical and theoretical understanding of the technologies for data mining.
5. To understand the strengths and limitations of various data mining models.

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	PO4	PO5	PO6	PSO1	PSO2	DoK
CO1	Ability to perform the preprocessing of data and apply mining techniques on it.	3	3	2	2	1	2	3	2	L1, L2
C02	Ability to identify the association rules, classification and clusters in large data sets.	3	3	3	2	1	3	3	2	L2, L3
CO3	Ability to solve real world problems in business and scientific information using data mining.	3	3	2	2	2	3	3	3	L3, L4
CO4	Ability to classify web pages, extracting knowledge from the web.	3	3	3	2	2	3	3	3	L4, L5
CO5	Evaluate data mining algorithms using appropriate metrics for accuracy, efficiency, and scalability.	3	3	2	3	3	2	3	3	L5, L6

SYLLABUS

UNIT-I: **(12 Hours)**

Introduction to Data Mining: Introduction, What is Data Mining, Definition, KDD, Challenges, Data Mining Tasks, Data Pre-processing, Data Cleaning, Missing data, Dimensionality Reduction, Feature Subset Selection, Discretization and Binarization, Data Transformation; Measures of Similarity and Dissimilarity- Basics. **COs: CO1**

Self-Learning Points: Real-world case studies of data cleaning and preprocessing in industries

UNIT-II: **(12 Hours)**

Association Rules: Problem Definition, Frequent Item Set Generation, The APRIORI Principle, Support and Confidence Measures, Association Rule Generation; APRIOIRI Algorithm, The Partition Algorithms, FP-Growth Algorithms, Compact Representation of Frequent Item Set-Maximal Frequent Item Set, Closed Frequent Item Set. **COs: CO2**

Self-Learning Points: Market Basket Analysis applications in retail (Amazon, Walmart), Advanced algorithms: ECLAT Algorithm, Hash-based techniques for frequent itemset mining.

UNIT-III: **(12 Hours)**

Classification: Problem Definition, General Approaches to solving a classification problem, Evaluation of Classifiers, Classification techniques, Decision Trees-Decision tree Construction, Methods for Expressing attribute test conditions, Measures for Selecting the Best Split, Algorithm for Decision tree Induction; Naive-Bayes Classifier, Bayesian Belief Networks; K- Nearest neighbor classification-Algorithm and Characteristics. **COs: CO3**

Self-Learning Points: Implementation of Random Forests and Ensemble Methods (beyond syllabus basics), Use of ROC curves, Precision-Recall, and F1-score for classifier evaluation.

UNIT-IV: **(12 Hours)**

Clustering: Problem Definition, Clustering Overview, Evaluation of Clustering Algorithms, Partitioning Clustering- K-Means Algorithm, K-Means Additional issues, PAM Algorithm; Hierarchical Clustering-Agglomerative Methods and divisive methods, Basic Agglomerative Hierarchical Clustering Algorithm, Specific techniques, Key Issues in Hierarchical Clustering, Strengths and Weakness; Outlier Detection. **COs: CO4**

Self-Learning Points: Advanced clustering techniques: DBSCAN, OPTICS, Density-based methods.

UNIT-V: **(12 Hours)**

Web and Text Mining: Introduction, web mining, web content mining, web structure mining, we usage mining, Text mining –unstructured text, episode rule discovery for texts, hierarchy of categories, text clustering. **COs: CO5**

Self-Learning Points: Sentiment Analysis using Twitter or product reviews, PageRank Algorithm and its role in web structure mining (Google Search).

Board of Studies : Computer Science and Engineering
Approved in BOS No : ____, August, 2024
Approved in ACM No: 01

Text Books:

1. Data Mining- Concepts and Techniques- Jiawei Han, Micheline Kamber, Morgan Kaufmann Publishers, Elsevier, Edition, 2006.
2. Introduction to Data Mining, Pang-Ning Tan, Vipin Kumar, Michael Steinbanch, Pearson Education.
3. Data mining Techniques and Applications, Hongbo Du Cengage India Publishing.

References:

1. Data Mining Techniques, Arun K Pujari, 3rd Edition, Universities Press.
2. Data Mining Principles & Applications – T.V Sveresh Kumar, B.Esware Reddy, Jagadish S Kalimani, Elsevier.
3. Data Mining, Vikaram Pudi, P Radha Krishna, Oxford University Press.

Extensive Reading:

1. Data Mining – Wikipedia
2. Data Mining Tutorial – GeeksforGeeks
3. Introduction to Data Mining – PDF (Tan, Steinbach & Kumar)

Internal Assessment Pattern

Cognitive Level	Internal Assessment #1(%)	Internal Assessment #2(%)
L1	35	--
L2	40	--
L3	25	20
L4	--	35
L5	--	35
L6	--	10
Total (%)	**100**	**100**

Sample Short and Long Answers questions of Various Cognitive Levels

L1: Remember

1. Define Data Mining.
2. What is KDD?
3. List the steps in data preprocessing.
4. Define support and confidence.
5. What is Apriori principle?
6. Name any two classification techniques.
7. Define clustering.
8. List different types of web mining.
9. What is an outlier?
10. Mention two challenges of data mining.

L2: Understand

1. Explain the need for data cleaning.
2. Illustrate similarity and dissimilarity measures with examples.
3. Describe Apriori algorithm in brief.
4. Differentiate between supervised and unsupervised learning.
5. Explain how decision trees are constructed.
6. Summarize the working of Naïve Bayes classifier.
7. Explain hierarchical clustering methods.

8. Discuss the importance of dimensionality reduction.
9. Interpret the role of text mining in modern applications.
10. Explain the difference between web structure mining and web content mining.

L3: Apply

1. Apply discretization techniques to a sample dataset.
2. Use Apriori algorithm to find frequent itemsets for a supermarket dataset.
3. Construct a decision tree for student grade classification.
4. Apply Naïve Bayes for spam email detection.
5. Implement K-Means clustering for customer segmentation.
6. Use similarity measures to compare documents.
7. Apply FP-growth algorithm to transactional data.
8. Demonstrate classification of patients as “high risk” and “low risk” using KNN.
9. Apply preprocessing steps on a dataset with missing values.
10. Perform basic web usage mining on server logs.

L4: Analyze

1. Analyze the challenges in KDD process.
2. Compare Apriori and FP-growth algorithms.
3. Differentiate between closed and maximal frequent itemsets.
4. Examine strengths and weaknesses of decision tree classifiers.
5. Analyze issues in K-Means clustering.
6. Differentiate between agglomerative and divisive clustering.
7. Analyze the importance of outlier detection in clustering.
8. Compare Bayesian Belief Networks and Naïve Bayes classifiers.
9. Analyze the scalability issues in association rule mining.
10. Examine text clustering challenges.

L5: Evaluate

1. Evaluate the effectiveness of dimensionality reduction techniques.
2. Assess the efficiency of FP-Growth over Apriori.
3. Evaluate the performance of decision tree classifiers vs KNN.
4. Judge the usefulness of clustering in market basket analysis.
5. Evaluate the impact of outlier detection on clustering quality.
6. Assess the role of web content mining in search engines.
7. Evaluate the effectiveness of text mining in sentiment analysis.

8. Critique different similarity measures used in clustering.
9. Judge the limitations of Bayesian classifiers in real-world data.
10. Recommend suitable preprocessing techniques for noisy datasets.

L6: Create

1. Design a data preprocessing pipeline for a banking dataset.
2. Create an association rule-based recommender system.
3. Develop a decision tree for employee attrition prediction.
4. Design a Naïve Bayes model for medical diagnosis.
5. Build a customer segmentation system using K-Means.
6. Create a web usage mining model for an e-commerce website.
7. Develop a text classification system for news articles.
8. Design a hybrid clustering model combining partitioning and hierarchical methods.
9. Create an outlier detection framework for fraud detection.
10. Develop a mini-project: End-to-end Data Mining system (Preprocessing → Association → Classification → Clustering → Web/Text Mining).

Chairperson
Board of Studies (CSE)

R25MCA1205 **Software Engineering** **3 0 0 3**

(Master of Computer Applications)

Course Objectives:

1. The aim of the course is to provide an understanding of the working knowledge of the techniques for estimation, design, testing and quality management of large software development projects.
2. Topics include process models, software requirements, software design, software testing, software process/product metrics, risk management, quality management and UML diagrams

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	PO4	PO5	PO6	PSO1	PSO2	DoK
CO1	Ability to translate end-user requirements into system and software requirements, using e.g.	3	3	2	2	1	2	3	2	L1, L2
CO2	UML, and structure the requirements in a Software Requirements Document (SRD).	3	3	3	2	1	3	3	2	L2, L3
CO3	Ability to solve real world problems in business and scientific information using data mining.	3	3	2	2	2	3	3	3	L3, L4
CO4	Ability to classify web pages, extracting knowledge from the web.	3	3	3	2	2	3	3	3	L4, L5
CO5	Demonstrate skills in risk management and quality assurance aligned with industry standards (e.g., ISO 9000, CMMI).	3	3	3	2	2	3	3	3	L4, L5

SYLLABUS

UNIT-I: **(12 Hours)**

Introduction to Software Engineering: The evolving role of software, changing nature of software, software myths.

A Generic view of process: Software engineering- a layered technology, a process framework, the capability maturity model integration (CMMI)

Process models: The waterfall model, Spiral model and Agile methodology **COs: CO1**

UNIT-II: **(12 Hours)**

Software Requirements: Functional and non-functional requirements, user requirements, system requirements, interface specification, the software requirements document.

Requirements engineering process: Feasibility studies, requirements elicitation and analysis, requirements validation, requirements management. **COs: CO2**

UNIT-III: **(12 Hours)**

Design Engineering: Design process and design quality, design concepts, the design model. Creating an architectural design: software architecture, data design, architectural styles and patterns, architectural design, conceptual model of UML, basic structural modeling, class diagrams, sequence diagrams, collaboration diagrams, use case diagrams, component diagrams. **COs: CO3**

UNIT-IV: **(12 Hours)**

Testing Strategies: A strategic approach to software testing, test strategies for conventional software, black-box and white-box testing, validation testing, system testing, the art of debugging. **Metrics for Process and Products**: Software measurement, metrics for software quality.

COs: CO4

UNIT-V: **(12 Hours)**

Risk management: Reactive Vs proactive risk strategies, software risks, risk identification, risk projection, risk refinement, RMMM

Quality Management: Quality concepts, software quality assurance, software reviews, formal technical reviews, statistical software quality assurance, software reliability, the ISO 9000 quality standards.

COs: CO5

Board of Studies : Computer Science and Engineering
Approved in BOS No: 01, August, 2024
Approved in ACM No: 01

Textbooks:

1. Software Engineering, A practitioner's Approach- Roger S. Pressman, 6th edition, McGraw Hill International Edition.
2. Software Engineering- Sommerville, 7th edition, Pearson Education.

References Books:

1. Database Systems design, Implementation, and Management, Peter Rob & Carlos Coronel 7th Edition.
2. Fundamentals of Database Systems, Elmasri Navrate, Pearson Education
3. Introduction to Database Systems, C. J. Date, Pearson Education
4. Oracle for Professionals, The X Team, S.Shah and V. Shah, SPD.
5. Database Systems Using Oracle: A Simplified guide to SQL and PL/SQL, Shah, PHI.
6. Fundamentals of Database Management Systems, M. L. Gillenson, Wiley Student Edition.

Internal Assessment Pattern

Cognitive Level	Internal Assessment #1(%)	Internal Assessment #2(%)
L1	35	--
L2	40	--
L3	25	20

L4	--	35
L5	--	35
L6	--	10
Total (%)	**100**	**100**

Sample Short and Long Answers questions of Various Cognitive Levels

L1: Remember

1. Define software myths.
2. List different software process models.
3. Define functional and non-functional requirements.
4. What is CMMI?
5. What is UML?
6. List software testing strategies.
7. Define risk in software engineering.
8. What is a Software Requirements Document (SRD)?
9. Define black-box testing.
10. 10.What is ISO 9000?

L2: Understand

1. Explain feasibility study in requirements engineering.
2. Describe waterfall vs spiral models.
3. Explain the role of design patterns in software design.
4. Discuss differences between black-box and white-box testing.
5. Explain formal technical reviews.
6. Describe software reliability with examples.

L3: Apply

1. Create a use-case diagram for an ATM system.
2. Develop a class diagram for a student management system.
3. Apply black-box testing on a login module.
4. Write a sample risk identification report for a web application project.
5. Create a sequence diagram for online shopping.

L4: Analyze

1. Compare agile and waterfall models with respect to flexibility and risk.
2. Analyze differences between functional and non-functional requirements.

3. Differentiate between architectural styles (layered vs client-server).
4. Compare static vs dynamic testing techniques.
5. Analyze advantages of metrics in software quality assurance.

L5: Evaluate

1. Evaluate pros and cons of Agile methodology.
2. Assess effectiveness of UML in software design.
3. Critically analyze SQA methods in large projects.
4. Evaluate the efficiency of risk management strategies.
5. Assess limitations of ISO 9000 in software quality.

L6: Create

1. Design a mini SRS for an e-commerce application.
2. Create UML diagrams (use-case, class, sequence) for a library management system.
3. Develop a testing plan for a banking application.
4. Prepare a risk management model for a software startup.
5. Create a software quality checklist for a mobile app project.

Chairperson
Board of Studies (CSE)

25MCA1206 **Database Management Systems Lab** **0 0 4 2**
(Master of Computer Applications)

Course Objectives:

1. Introduce ER data model, database design and normalization
2. Learn SQL basics for data definition and data manipulation

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	DoK
CO1	Design database schema for a given application and apply normalization.	3	3	3	L1,L2
CO2	Acquire skills in using SQL commands for data definition and data manipulation.	3	3	3	L2,L3
CO3	Develop solutions for database applications using procedures, cursors and triggers.	3	3	3	L2,L3

Board of Studies : Computer Science and Engineering
Approved in BOS No : ____, August, 2024
Approved in ACM No: 01

SYLLABUS

Developing the following programs:

Experiment 1:

1. Concept design with E-R Model

Experiment 2:

1.Relational Model

Experiment 3:

1.Normalization

Experiment 4:

1.Practicing DDL commands

Experiment 5:

1.Practicing DML commands

Experiment 6:

1. i). Querying (using ANY, ALL, UNION, INTERSECT, JOIN, Constraints etc.)
   ii). Nested, Correlated subqueries

Experiment 7:

1.Queries using Aggregate functions, GROUP BY, HAVING and Creation and dropping of Views

Experiment 8:

1.Triggers (Creation of insert trigger, delete trigger, update trigger)

Experiment 9:

1. Procedures.

Experiment 10:

1.Usage of Cursors

Text Books:

1. Database Management Systems, Raghurama Krishnan, Johannes Gehrke, Tata Mc Graw Hill,3rd Edition
2. Database System Concepts, Silberschatz, Korth, McGraw Hill, V edition.

Reference Books:

1. Database Systems design, Implementation, and Management, Peter Rob & Carlos Coronel 7thEdition.
2. Fundamentals of Database Systems, Elmasri Navrate, Pearson Education
3. Introduction to Database Systems, C.J. Date, Pearson Education
4. Oracle for Professionals, The X Team, S. Shah and V. Shah, SPD.
5. Database Systems Using Oracle: A Simplified guide to SQL and PL/SQL, Shah, PHI.
6. Fundamentals of Database Management Systems, M. L. Gillenson, Wiley Student Edition.

Chairperson
Board of Studies (CSE)

25MCA1207 **Computer Networks Lab** **0 0 3 1.5**

(Master of Computer Applications)

Course Objectives:

1. To understand the working principle of various communication protocols.
2. To understand the network simulator environment and visualize a network topology and observe its performance
3. To analyze the traffic flow and the contents of protocol frames.

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	DoK
CO1	Implement data link layer farming methods	3	3	3	L1,L2
CO2	Analyze error detection and error correction codes.	3	3	3	L2,L3
CO3	Implement and analyze routing and congestion issues in network design.	3	3	3	L2,L3
CO4	Implement Encoding and Decoding techniques used in presentation layer	3	3	3	L2,L3
CO5	To be able to work with different network tools	3	3	3	L2,L3

Board of Studies : Computer Science and Engineering
Approved in BOS No : ____, August, 2024
Approved in ACM No: 01

SYLLABUS

Developing the following programs:

Experiment 1: Implement the data link layer framing methods such as character, character-stuffing and bit stuffing.

Experiment 2: Write a program to compute CRC code for the polynomials CRC-12, CRC-16 and CRC CCIP

Experiment 3: Develop a simple data link layer that performs the flow control using the sliding window protocol, and loss recovery using the Go-Back-N mechanism.

Experiment 4: Implement Dijsktra's algorithm to compute the shortest path through a network.

Experiment 5: Take an example subnet of hosts and obtain a broadcast tree for the subnet.

Experiment 6: Implement distance vector routing algorithm for obtaining routing tables at each node.

Experiment 7: Implement data encryption and data decryption.

Experiment 8: Write a program for congestion control using Leaky bucket algorithm.

Experiment 9: Write a program for frame sorting techniques used in buffers.

Experiment 10: Wireshark

i. Packet Capture Using Wire shark

ii. Starting Wire shark
iii. Viewing Captured Traffic
iv. Analysis and Statistics & Filters.

Experiment 11: How to run Nmap scan.

Experiment 12: Operating System Detection using Nmap

Experiment 13: Do the following using NS2 Simulator

i. NS2 Simulator-Introduction
ii. Simulate to Find the Number of Packets Dropped
iii. Simulate to Find the Number of Packets Dropped by TCP/UDP
iv. Simulate to Find the Number of Packets Dropped due to Congestion
v. Simulate to Compare Data Rate & Throughput.
vi. Simulate to Plot Congestion for Different Source/Destination
vii. Simulate to Determine the Performance with respect to Transmission of Packets.

Text Books:

1. Computer Networks, Andrew S Tanenbaum, David. j. Wetherall, 5th Edition. Pearson Education/PHI

Reference Books:

1. An Engineering Approach to Computer Networks, S. Keshav, 2nd Edition, Pearson Education
2. Data Communications and Networking – Behrouz A. Forouzan. 3rd Edition,

Chairperson
Board of Studies (CSE)

25MCA1208 **Java Programming Lab** **0 0 3 1.5**

(Master of Computer Applications)

Course Objectives:

1. To understand OOP principles.
2. To understand the Exception Handling mechanism.
3. To understand Java collection framework.
4. To understand multithreaded programming.
5. To understand swing controls in Java.

Course Outcomes:

At the end of the course, students will be able to:

Course Code	Course Outcomes (COs)	PO1	PO2	PO3	DoK
CO1	Able to write the programs for solving real world problems using Java OOP principles.	3	3	3	L1,L2
CO2	Able to write programs using Exceptional Handling approach.	3	3	3	L2,L3
CO3	Able to write multithreaded applications.	3	3	3	L2,L3
CO4	Able to write GUI programs using swing controls in Java.	3	3	3	L2,L3

Board of Studies : Computer Science and Engineering
Approved in BOS No : ____, August, 2024
Approved in ACM No: 01

SYLLABUS

Developing the following programs:

Experiment 1:

Use Eclipse or Net bean platform and acquaint yourself with the various menus. Create a test project, add a test class, and run it. See how you can use auto suggestions, auto fill. Try code formatter and code refactoring like renaming variables, methods, and classes. Try debug step by step with a small program of about 10 to 15 lines which contains at least one if else condition and a for loop.

Experiment 2:

Write a Java program to demonstrate the OOP principles. [i.e., Encapsulation, Inheritance, Polymorphism and Abstraction]

Experiment 3:

Write a Java program to handle checked and unchecked exceptions. Also, demonstrate the usage of custom exceptions in real time scenario.

Experiment 4:

Write a Java program on Random Access File class to perform different read and write operations.

Experiment 5:

Write a Java program to demonstrate the working of different collection classes. [Use package structure to store multiple classes].

Experiment 6:

Write a program to synchronize the threads acting on the same object. [Consider the example of any reservations like railway, bus, movie ticket booking, etc.]

Experiment 7:

Write a program to perform CRUD operations on the student table in a database using JDBC.

Experiment 8:

Write a Java program that works as a simple calculator. Use a grid layout to arrange buttons for the digits and for the +, -,*, % operations. Add a text field to display the result. Handle any possible exceptions like divided by zero.

Experiment 9:

Write a Java program that handles all mouse events and shows the event name at the center of the window when a mouse event is fired. [Use Adapter classes

Text Books:

1. Java The complete reference, 9th edition, Herbert Schildt, McGraw Hill Education (India) Pvt. Ltd.
2. Understanding Object-Oriented Programming with Java, updated edition, T. Budd, Pearson Education.

Reference Books:

1. Java for Programmers, P. J. Deitel and H. M. Deitel, 10th Edition Pearson education.
2. Thinking in Java, Bruce Eckel, Pearson Education.
3. Java Programming, D. S. Malik and P. S. Nair, Cengage Learning.
4. Core Java, Volume 1, 9th edition, Cay S. Horstmann and G Cornell, Pearson.

Chairperson
Board of Studies (CSE)