ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES (A)

(UGC Autonomous)

Approved by AICTE, Affiliated to Andhra University, Accredited by N.B.A & NAAC with A+ Grade

(Estd: 2001)

PRAGNANAM BRAHMA

ANITS

2023-2024

Academic Regulations (R23)

III Year B.Tech Syllabi

Department of Computer Science & Engineering

III Year Course structure – CSE

Semester –I

CODE	SUBJECT NAME	Category	Periods				Sessional Marks	Semester end Exam marks	Total Marks	Credits
			L	T	P	Total				
23DP6111	Open Elective- I	OE	3	0	0	3	40	60	100	3
23CS5111-23CS5115	Professional Elective-1	PE	3	0	0	3	40	60	100	3
23CS4119	Software Engineering	PC	3	0	0	3	40	60	100	3
23CS4120	Compiler Design	PC	3	0	0	3	40	60	100	3
23CS4121	Competitive Programming	PC	3	0	0	3	40	60	100	3
23CS4221	Competitive Programming Lab	PC	0	0	3	3	50	50	100	1.5
23CS4219	Software Engineering Lab	PC	0	0	3	3	50	50	100	1.5
23CS9213	Data Analytics and Visualization	SC	0	0	2	2	50	50	100	1
23CR9103	Quantitative Aptitude and Effectual Communication Skills	HS	0	0	2	2	50	50	100	1
23CS9401	Summer Internship -I	PR	0	0	3	3	0	100	100	1.5
Total			15	0	13	28	400	600	1000	21.5

DATA MINING (Open Elective – I)	
23DP6111(A)	Credits: 3
Instruction: 3 Periods / Week	Sessional Marks: 40
End Exam: 3 hours	End Exam Marks: 60

Prerequisites:

- Knowledge on Database Management Systems.
- Probability and Statistics

Course Objectives:

- Describe the basic concepts and applications of Data Warehouse and Data Mining techniques.
- Examine the types of the data to be mined and apply pre-processing methods on raw data.
- Discover interesting patterns, analyse supervised and unsupervised models and estimate the accuracy of the algorithms

Course Outcomes:

By the end of the course, the student will be able to:

1. Identify the challenging issues in Data Mining data warehousing.
2. Remove redundancy and incomplete data from the dataset using data pre-processing methods.
3. Apply classification by using decision tree induction, Bayesian, back propagation and prediction methods for data analysis.
4. Analyse association rule mining in various dimensional databases.
5. Apply various clustering techniques.

Mapping of course outcomes with program outcomes:

CO	**PO**												**PSO**	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
1	2	1	2	-	-	-	-	-	-	-	-	-	-	2
2	2	2	2	-	2	-	-	-	-	-	-	2	1	2
3	2	2	3	2	2	-	-	-	-	2	-	2	1	2
4	2	2	3	2	2	-	-	-	-	2	-	2	1	2
5	2	2	3	2	2	-	-	-	-	2	-	2	1	2

COURSE CONTENTS

UNIT- I: **12 Periods**

Data Warehouse and OLAP Technology:

Data Warehouse Concepts: Introduction to Data Warehouse, Differences between Operational Database Systems and Data Warehouses - Comparison of OLTP and OLAP Systems - A three-tier data warehousing architecture, Data Warehouse Modelling: Data Cube-Stars, Snowflakes, and Fact Constellations Schemas for Multidimensional Data Models- Role of Concept Hierarchy, Typical OLAP Operations.

Introduction to Data Mining:

Knowledge Discovery of Database process, Types of Data for mining, Data Mining Functionalities, Techniques, Applications, Major Issues in Data Mining.

Learning Outcomes: At the end of this unit, students will be able to:

- Design the data warehouse and apply OLAP operations
- Classify data mining systems and identify the major issues in data mining.

UNIT-II: **12 Periods**

Data Pre-processing:

Data Exploration: Data Objects and attribute types, Basic Statistical description of data, Data Visualization, Data similarity and dissimilarity measures.

Data Pre-processing: Data Cleaning, Data Integration, Data Reduction Data Transformation and Data Discretization.

Learning Outcomes: At the end of this unit, students will be able to:

- Apply pre-processing steps.
- Find the importance of data pre-processing methodologies.

UNIT –III: **12 Periods**

Classification:

Basic Concepts, General Approach to solve a classification problem, Decision Tree Induction: Working of Decision Tree, building a decision tree, methods for expressing an attribute test conditions, measures for selecting the best split, Support vector Machines, Regression-Introduction, Types of Regression, Evaluating the performance of classification model -Accuracy, error measures.

Learning Outcomes: At the end of this unit, students will be able to:

- Relate the importance of classification in data analysis
- Apply techniques in real time data for analysis.

UNIT-IV: **12 Periods**

Association Analysis:

Problem Definition-Market basket Analysis, Frequent Item Set generation - Apriori Algorithm-Candidate generation and pruning, support counting, Rule generation, Compact representation of frequent item sets- Maximal Item frequent sets, closed frequent Item sets, FP-Growth Algorithm-Frequent Item set Generation.

Learning Outcomes: At the end of this unit, students will be able to:

- Generate association rules to given data and analyse market basket analysis.
- Apply Apriori & FP growth algorithms to generate association rules

UNIT-V: **12 Periods**

Cluster Analysis:
Introduction to Cluster Analysis, Different Types of Clustering, Different Types of Clusters, K-means: The Basic K-means Algorithm, K-means Additional Issues, Strengths and Weaknesses, Agglomerative Hierarchical Clustering Algorithm, DBSCAN Algorithm- Strengths and Weaknesses.

Learning Outcomes: At the end of this unit, students will be able to:

1. Outline various types of clustering methods.
2. Apply and analyse various types of clustering methods to real time data set.

TEXT BOOKS:

1. Data Mining concepts and Techniques, 3/e, Jiawei Han, Michel Kamber, Elsevier (Unit-I, II)
2. Introduction to Data Mining: Pang-Ning Tan & Michael Steinbach, Vipin Kumar, Pearson (Unit-III, IV, V)

REFERENCE BOOKS:

1. Data Mining Techniques and Applications: An Introduction, Hongbo Du, Cengage Learning.
2. Data Mining: Vikram Pudi and P. Radha Krishna, Oxford.
3. Data Warehousing Data Mining & OLAP, Alex Berson, Stephen Smith, TMH

WEB RESOURCES:

1. http://onlinecourses.nptel.ac.in/noc18_cs14/preview (NPTEL course by Prof.Pabitra Mitra)
2. http://onlinecourses.nptel.ac.in/noc17_mg24/preview (NPTEL course by Dr. Nandan Sudarshanam & Dr. Balaraman Ravindran)

Operating Systems

(Open Elective – I)

Code: 23DP6111(B)	**Credits:3**
Instruction: 3 periods/week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Prerequisites:

- Basic programming language and Computer Organization.

Course Objectives:

- Identify the main components of an operating system and explain their functions.
- Explore process basics, scheduling, operations, inter-process communication, and multithreading models.
- Demonstrate the concepts of process synchronization and Deadlocks.
- Analyze various memory management mechanisms for contiguous and non-contiguous memory.
- Analyze the structure and organization of the file systems, the secondary storage structure, and the protection of the system.

Course Outcomes:

Students will be able to

1. Describe the fundamental concepts of operating systems, including their functionalities, types, and structures, and explain system calls.
2. Explain the concepts of processes and process scheduling, identify operations on processes and inter-process communication mechanisms, and differentiate between various multithreading models.
3. Demonstrate the concepts of process synchronization and Deadlocks.
4. Apply and analyze various memory management mechanisms for contiguous and non-contiguous memory.
5. Demonstrate the structure and organization of the file systems and analyze the implementation of the file systems. Analyze the secondary storage structure, protection of the system.

Mapping of Course Outcomes with Program Outcomes:

Mapping		**PO**												**PSO**	
		1	2	3	4	5	6	7	8	9	10	11	12	1	2
CO	1	1	1	1	1	-	-	-	-		-	-	-	-	1
	2	2	2	2	1	-	-	-	-	-	-	-	-	1	1
	3	2	2	2	1	-	-	-	-	-	-	-	-	1	1
	4	2	2	2	1	-	-	-	-	-	-	-	-	1	1
	5	2	2	2	1	-	-	-	-	-	-	-	-	-	1

COURSE CONTENTS

UNIT I **10 Periods**

Introduction to OS: Operating system Definition, Operating system Functionalities, Types of Operating system, operating system structures, system calls.

Learning outcomes:

1. Describe the definition, functionalities, and various types of operating systems.
2. Explain operating system structures and the role of system calls in system operations.

UNIT II **12 Periods**

Process Management: Process concept, Process scheduling, Operations on process, Inter process communication.
Threads: Overview, Multithreading models
CPU Scheduling: Scheduling criteria, Scheduling algorithms, Algorithm Evaluation.

Learning outcomes:

1. Explain process management concepts, including process lifecycle, scheduling, operations, and inter-process communication mechanisms.
2. Analyse multithreading models and evaluate various CPU scheduling algorithms based on scheduling criteria.

UNIT III **14 Periods**

Process Synchronization: The Critical Section Problem, Peterson ‘s Solution problem, Synchronization Hardware, Semaphores, Classical Problems of Synchronization, Monitors.
Deadlocks: Avoidance, Deadlock Detection, Recovery from Deadlocks and handling dead locks.

Learning outcomes:

1. Apply synchronization techniques such as semaphores and monitors to solve critical section problems and classical synchronization challenges.
2. Analyze deadlock conditions and evaluate strategies for deadlock avoidance, detection, and recovery.

UNIT IV **12 Periods**

Memory Management: Logical versus Physical Address, allocation (Contiguous and Non- Non-Contiguous allocation), protection, hardware support, paging, swapping, segmentation.
Virtual Memory: Demand Paging, Allocation, Page Replacement, Thrashing.

Learning outcomes:

1. Explain memory management concepts, including addressing, allocation methods, paging, segmentation, and hardware support.
2. Analyze virtual memory techniques like demand paging, page replacement, and thrashing.

UNIT V **12 Periods**

File Systems: Implementation: Basic Concepts of a file, Access Methods, Directory Structure, Protection, File System Structure, File System design and Implementation allocation methods, directory management, mounting.
Secondary-Storage Structure: Disk attachment, Disk scheduling, Disk management. I/O Systems, Goals of Protection, Access Matrix, Implementation of Access Matrix.

Learning outcomes:

1. Explain file system implementation concepts, including file structures, access methods, directory management, and protection mechanisms.
2. Analyze secondary storage management, disk scheduling, I/O systems, and the implementation of an access matrix for system protection.

TEXT BOOKS

1. Abraham Silberschatz, Peter B. Galvin and Greg Gagne, "Operating System Concepts", 10th Edition, Wiley India Pvt Ltd, 2018.

REFERENCES

1. Andrew S. Tanenbaum, "Modern Operating Systems", 4th Edition, Pearson Education, 2015.
2. William Stalling, "Operating Systems: Internals and Design Principles", 9th edition, PHI, 2018.

WEB REFERENCES:

1. https://nptel.ac.in/courses/106/106/106106144/
2. https://nptel.ac.in/content/storage2/courses/106108101/pdf/PPTs/Mod_13.pdf

DATA BASE MANAGEMENT SYSTEM (Open Elective – I)	
23DP6111(C)	**Credits: 3**
Instruction: 3 Periods /week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Prerequisites:

Basic computer literacy, fundamental mathematics (set theory, logic, probability), basic programming knowledge, and an understanding of data structures.

Course Objectives:

1. Study the foundational concepts of Database Management Systems.
2. Learn various data models and relational database design.
3. Gain proficiency in SQL for data retrieval and manipulation.
4. Examine the principles of database normalization and data integrity.
5. Explore transaction management, concurrency control, and database recovery**.**

Course Outcomes:

1. Master the key principles of database management systems.
2. Design relational databases using the E-R model.
3. Write and execute SQL queries for data manipulation.
4. Apply normalization techniques to enhance database efficiency.
5. Gain insight into transaction management, concurrency control, and recovery mechanisms.

Mapping of Course Outcomes with Program Outcomes:

CO	**PO**												**PSO**	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
1	3	-	-	-	-	-	-	-	1	-	-	-	-	-
2	1	3	2	1	-	-	-	-	2	-	-	-	1	1
3	-	-	3	1	-	-	-	-	1	-	-	1	3	2
4	2	2	-	2	-	-	-	-	1	-	-	-	1	-
5	-	-	3	2	-	-	-	-	2	-	-	-	2	-

COURSE CONTENTS

UNIT-I **12 periods**

Introduction to Database Systems: Data, Information, and Databases, File-Based System vs. Database Management System, Advantages of DBMS, Components of DBMS: Storage Manager, Query Processor, Transaction Manager, Users of DBMS (Administrators, Designers, End Users), DBMS Architecture (1-Tier, 2-Tier, 3-Tier)

Learning Outcomes:

1. Differentiate between data, information, and databases and explain how databases store and manage structured data.
2. Discuss the key benefits of DBMS, including data integrity, security, consistency, and reduced redundancy and recognize how DBMS improves data retrieval and sharing across multiple users.

UNIT-II **12 periods**

Data Models and Database Design: Introduction to Data Models (Hierarchical Model, Network Model, Relational Model, Object-Oriented Model) Entity-Relationship (E-R) Model (Entities, Attributes, Relationships, E-R Diagrams and Mapping Cardinalities, Conversion of E-R Model to Relational Model) Relational Model and Constraints (Tables, Tuples, Attributes, Keys: Primary Key, Foreign Key, Candidate Key, Integrity Constraints (Domain, Referential, Key Constraints))

Learning Outcomes:

1. Explain various data models (Hierarchical, Network, Relational, Object-Oriented), Identify the significance of the Relational Model in modern database design.
2. Develop E-R diagrams to represent real-world scenarios, Map E-R models to relational schemas by converting entities and relationships into tables.

UNIT-III **12 periods**

Structured Query Language (SQL): Introduction to SQL , Data Definition Language (DDL)(CREATE, ALTER, DROP, TRUNCATE), Data Manipulation Language (DML) (INSERT, UPDATE, DELETE), Data Query Language (DQL) (SELECT Statement, WHERE, ORDER BY, GROUP BY, HAVING, DISTINCT), Data Control Language (DCL) (GRANT, REVOKE), Transaction Control Language (TCL) (COMMIT, ROLLBACK, SAVEPOINT), Aggregate Functions (COUNT, SUM, AVG, MAX, MIN), Joins (INNER, OUTER, LEFT, RIGHT, SELF), Subqueries and Nested Queries.

Learning Outcomes:

1. Explain the purpose and significance of SQL in database management, Differentiate between DDL, DML, DQL, DCL, and TCL commands.
2. Construct optimized queries for business intelligence, reporting, and decision-making.

UNIT IV: **12 periods**

Database Normalization and Indexing: Functional Dependencies, Normalization (1st Normal Form (1NF), 2nd Normal Form (2NF), 3rd Normal Form (3NF), Boyce-Codd Normal Form (BCNF). **Transaction Management and Concurrency Control:** ACID Properties (Atomicity, Consistency, Isolation, Durability).

Learning Outcomes:

1. Define functional dependencies and their role in database design and identify determinants and dependent attributes in a relation.
2. Evaluate a database schema and suggest normalization improvements.
3. Define a transaction and identify different states of a transaction (Active, Partially Committed, Failed, Aborted, Committed).

UNIT V **12 periods**

NoSQL: What is NoSQL? Differences between SQL and NoSQL databases, CAP Theorem (Consistency, Availability, Partition Tolerance), BASE vs ACID Properties, Types of NoSQL Databases (Document-based, Key-Value Store, Column-Oriented, Graph-Based).

Learning Outcomes:

1. Define NoSQL and identify differences with SQL.
2. Knowledge gaining on ACID and BASE properties

TEXT BOOKS:

1. Raghu Ramakrishnan, Johannes Gehrke" Database Management Systems", 4th Edition, McGraw- Hill 2003. (UNIT- I to IV)
2. Pramod J.Sadalag and Martin Fowler," NoSQL Distilled, A Brief Guide to the Emerging World of Polyglot Persistence" ,1st Edition, Addison Wesley (UNIT-V)

REFERENCE BOOKS:

1. Ramez Elmasri & Shamkant B. Navathe – Fundamentals of Database Systems
2. Abraham Silberschatz, Henry F. Korth, S. Sudarshan – Database System Concepts
3. C.J. Date – An Introduction to Database Systems
4. Ivan Bayross – SQL, PL/SQL – The Programming Language of Oracle

WEB RESOURCES:

1. https://www.oreilly.com/library/view/web-database-applications/0596005431/ch01.html
2. http://nptel.ac.in/courses/IIT-MADRAS/Intro_to_Database_Systems_Design/pdf/1_Intro duction.pdf
3. https://www.edx.org/learn/databases
4. https://www.youtube.com/watch?v=1057YmExS-I

DATA STRUCTURES AND APPLICATIONS (Open Elective – I)	
23DP6111(D)	**Credits : 3**
Instruction: 3 Periods & 1 Tut/Week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Prerequisites:

Basic Knowledge of Programming Fundamentals and Problem Solving

Course Objectives:

The course should enable the students:

1. To acquire knowledge on several linear and nonlinear data structures like stacks, queues, linked list, trees and graphs.
2. To have better insight into to learn various sorting and searching techniques.
3. To exercise the applications of data structures.
4. To have a good understanding of problem-solving using data structure tools and techniques.

Course Outcomes:

Students will be able to.

1. Analyze the complexities of recursive and non-recursive algorithms and implement linear, binary, interpolation, hashing searching techniques and sorting techniques namely bubble, insertion, selection, quick, merge sort.
2. Apply ADT concepts such as stacks and queues for solving infix to postfix, postfix evaluation and queue applications.
3. Apply the concepts of dynamic memory allocation to implement Linked Lists.
4. Design and implement the Nonlinear data structures (trees) to optimize the solution.
5. Design and Implement Warshall's Algorithm, Shortest path Algorithm-Dijkstra's Algorithm, Minimum cost spanning trees (Prims and Kruskal's algorithms), Graph traversals (Breadth first search and Depth first Search algorithms.)

Mapping of Course Outcomes with Program Outcomes:

Mapping		**PO**												**PSO**	
		1	2	3	4	5	6	7	8	9	10	11	12	1	2
CO	1	3	3	2	3	-	-	-	-	-	-	-	1	3	-
	2	2	2	3	2	-	-	-	-	-	-	-	1	2	-
	3	2	2	3	2	-	-	-	-	-	-	-	1	3	-
	4	2	2	3	2	-	-	-	-	-	-	-	1	2	-
	5	2	3	3	3	-	-	-	-	-	-	-	1	3	-

SYLLABUS

UNIT-I: **12 periods**

Introduction : Data Structure operations, Fundamentals of analysis of algorithms and efficiency – Asymptotic Notations and Basic Efficiency classes.

Searching &Sorting: Sequential search, binary search, Interpolation Search, comparison and analysis, Hash Table, Hash Functions, Collision Resolution Techniques- Open hashing, Closed hashing. Insertion Sort, Shell sort, Quick Sort, Merge Sort.

Learning Outcomes:

1. Analyze the complexity of Algorithms.
2. Implement the searching and sorting algorithms.

UNIT-II: **12 periods**

Stacks: Array Representation and Implementation of stack, Operations on Stacks: Push & Pop, Applications of stack: Conversion of Infix to prefix and Postfix Expressions, Evaluation of Postfix & Prefix expressions using stack, Recursion, Towers of Hanoi Problem.

Queues: Array representation and implementation of queues, Operations on Queue: Insert, Delete, Full and Empty. Circular queue, Applications of Queues.

Learning Outcomes:

1. Apply ADT to implement Stack and queue
2. Apply ADT to implement applications of stack and queue.

UNIT-III: **12 periods**

Linked list: Representation and Implementation of Singly Linked Lists, Traversing and Searching of Linked List, Insertion and deletion to/from Linked Lists, doubly linked list, Circular doubly linked list, implementing priority queue using Linked List, Polynomial Representation using Linked list & addition.

Learning Outcomes:

1. Implement singly linked list, Doubly Linked List, Circular doubly linked list and applications.
2. Develop the skills to implement data structures and algorithms using linked lists, such as priority queues and polynomial representations.

UNIT-IV: **12 periods**

Trees: Basic terminology, Binary Trees-Full Binary Tree, Complete Binary Tree, Extended Binary Tree, Array and Linked Representation of Binary trees, Traversing Binary trees, Threaded Binary trees. Binary Search Tree (BST), Insertion and Deletion in BST, AVL Trees-Rotations in AVL trees, Insertion and Deletion in AVL.

Learning Outcomes:

1. Design and implement BST along with its operations.
2. Implement AVL tree along with its operations.

UNIT-V: **12 periods**

Graphs: Graphs Terminology, Types of Graphs, Representations- Adjacency Matrices, Adjacency List, Path or Transitive Closure of a Graph, Warshall's Algorithm, Graph Traversals, Shortest path Algorithm- Dijkstra's Algorithm, Connected Component and Spanning Trees, Minimum Cost Spanning Trees.

Learning Outcomes:

1. Implement Graph Traversals algorithm and Minimum Cost Spanning Trees algorithms.
2. Implement Warshall's Algorithm, Shortest path Algo-rithm Dijkstra's Algorithm.

TEXT BOOKS

1. Mark Allen Weiss, "Data Structures and Algorithm Analysis in C", Pearson Education, 2nd Edition, 1996

REFERENCE BOOKS

1. E.Horowitz and Sahani, "Fundamentals of Data Structures", W H Freeman& Co Publication, 1983.
2. S. Lipschutz, "Data Structures", McGraw Hill Publications, 1986.
3. P. Dey & M. Ghosh, "Programming in C" , Oxford Univ. Press, 2012
4. ISRD Group, "Data Structures through C++", McGraw Hill, 2011.

WEB RESOURCES:

1. https://nptel.ac.in/courses/106/102/106102064/
2. https://www.coursera.org/learn/data-structures?action=enroll&specialization=data-structures-algorithms
3. https://www.udacity.com/course/data-structures-and-algorithms-nanodegree-- nd256

COMPUTER GRAPHICS (Professional Elective -I)	
23CS5111	**Credits: 3**
Instruction: 3 Periods /week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Pre-requisites:

1. Elementary knowledge in C programming, Solving mathematical expressions, algorithm or pseudo code

Course-Objective:

1. Gain insights into real-world applications and the graphics systems essential for developing graphics.
2. Exploration of fundamental concepts in 2D and 3D computer graphics.
3. Learn two dimensional and three-dimensional computer graphics with comprehend advanced software tools of computer graphics

Course-Outcome:

Students will be able to

1. Discuss computer graphics, including its applications and hardware components.
2. Design 2D objects using various algorithms.
3. Apply geometric and viewing transformations on 2D objects.
4. Design 3D objects and apply geometric and viewing transformations on 3D objects.
5. Compare various visible surface methods.

Mapping of Course Outcomes with Program Outcomes:

Mapping		**PO**												**PSO**	
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
CO	**1**	1	1		-	-	-	-	-			-	1	-	-
	2	3	2	2	2	-	-	-	-	2	1	-	1	-	2
	3	3	2	2	2	-	-	-	-	2	1	-	1	-	2
	4	3	2	2	2	-	-	-	-	2	1	-	1	-	2
	5	2	1	1	-	-	-	-	-	1	1	-		-	2

COURSE CONTENTS

UNIT-I **12 periods**

Introduction: Basics of computer graphics, Applications. **Over view of Graphics systems:** Video Display Devices, Raster Scan systems, Random scan systems, Graphics monitors and workstations, Input devices, Graphics software.

Learning Outcomes: At the end of this unit, student will be able to

1. Express about the application in the real world and the computer Graphics.
2. Summarize the different graphic systems

UNIT-II

12 periods

Output primitives & its attributes: Points and Lines-Line Drawing Algorithms, Loading the Frame buffer, Line function, Circle Generating Algorithms, Ellipse Generating Algorithms, Filled Area Primitives, Filled Area Functions, Cell Array, Character Generation.

Attributes of Output Primitives: Line and Curve Attributes, Colour and Gray scale levels, Area Fill Attributes, Character Attributes, Bundled Attributes, Anti-aliasing.

Learning Outcomes: At the end of this unit, student will be able to

1. Observe various 2d output primitive algorithms
2. Interpret the attributes of output primitives

UNIT-III **12 periods**

Two Dimensional Geometric Transformations: Basic Transformations, Matrix Representations, Homogeneous Coordinates, Composite Transformations, Other Transformations, Transformations between Coordinate Systems.

Two-Dimensional Viewing: The Viewing Pipeline, Viewing Coordinate Reference Frame, Window to Viewport Coordinate Transformation, Two-Dimensional Viewing Functions, Clipping Operations, Point Clipping, Line Clipping, Polygon Clipping, Curve Clipping, Text and Exterior Clipping.

Learning Outcomes: At the end of this unit, student will be able to

1. Apply 2-dimensional geometric transformation to the real-world scenario
2. Evaluate various clipping algorithms and outline 2D viewing transformation

UNIT-IV **12 periods**

Three Dimensional Concepts and Object representations & Transformation: 3D display methods, 3D Graphics, Polygon Surfaces, Curved Lines and Surfaces, Quadratic Surfaces, Super Quadrics, Blobby Objects, Spline Representations, Cubic Spline methods, Bezier Curves and Surfaces, B Spline Curves and Surfaces.

Three Dimensional Geometric and Modelling Transformations: Translation, Rotation, Scaling, Other Transformations, Composite Transformations.

Three-Dimensional Viewing: Viewing Pipeline, Viewing Coordinates, Projections, View Volumes, General Projection Transformations, Clipping

Learning Outcomes: At the end of this unit, student will be able to

1. Review the 3d object representations and the analyse various surface representation.
2. Compare the 2D and 3D geometric and modelling transformations and explain the 3D viewing transformation.

UNIT-V **12 periods**

Visible Surface Detection & Hidden Surface Detection Methods: Classification of visible, Surface detection algorithms, Back face method, Depth buffer method, Scan line method, Depth Sorting Method, Z-buffer method, Area sub-division method, Comparison of hidden surface methods.
Polygon Rendering Methods: Constant-Intensity Method, Gouraud Method, Phong Method

Learning Outcomes: At the end of this unit, student will be able to

1. Compare and differentiate visible surface detection methods
2. Differentiate various polygon rendering methods.

TEXT BOOKS:

1. Computer Graphics C Version by Donald Hearn & M. Pauline Baker Pearson Education, New Delhi, 2004

REFERENCES BOOKS:

1. Procedural Elements for Computer Graphics by David F. Rogers, Tata McGraw Hill Book Company, New Delhi, 2003
2. Computer Graphics: Principles & Practice in C by J. D. Foley, S. K Feiner, A Van Dam F. H John, Pearson Education, 2004

WEB REFERENCES:

1. http://nptel.ac.in/courses/106106090/
2. https://www.coursera.org/courses?languages=en&query=computer+graphics
3. https://courses.edx.org/courses/BerkeleyX/CS-184.1x/2013_October/syllabus/

DISTRIBUTED OPERATING SYSTEMS (Professional Elective – I)	
23CS5112	**Credits: 3**
Instruction: 3 Periods /Week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Prerequisites:

- Fundamentals of Operating Systems and Computer networks and protocols.

Course Objectives:

1. To provide an overview of the concepts of distributed operating systems and challenges that includes Architecture and Fundamental Models.
2. To explore about various types of communication procedures and protocols in a distributed operating systems environment.
3. To interpret the concept of communication between distributed objects and remote procedural calls.
4. To analyze and understand the concepts of Distributed File system.
5. To demonstrate the idea of Transactions and Replications in distributed operating system.

Course Outcomes:

students will be able to:

1. Analyze the system model, software layers of distributed operating systems and its challenges.
2. Examine the inter-process communication, TCP stream communication procedures and protocols.
3. Evaluate the concepts of Remote procedural calls and communication among objects in distributed operating system.
4. Apply the knowledge of peer-to-peer system, distributed mutual exclusion of distributed file system in real world scenario.
5. Apply concurrency control, deadlock management techniques in distributed operating system for group communication.

CO-PO Mapping:

CO	**PO**												**PSO**	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
1	2	3	1	2	-	-	-	1	1	-	-	1	1	1
2	1	3	2	2	-	-	-	1	1	-	-	1	1	1
3	1	2	3	3	-	-	-	1	1	-	-	1	1	1
4	3	1	1	3	-	-	-	1	1	-	-	1	1	1
5	3	1	1	3	-	-	-	1	1	-	-	1	1	1

COURSE CONTENTS

UNIT-I: **12 periods**

Characterization of Distributed Systems: Introduction, Examples of Distributed Systems, Resource Sharing and the Web, System Models: Introduction, Architectural Models- Software Layers, System Architecture, Variations, Interface and Objects, Design Requirements for Distributed Architectures

Learning Outcomes: At the end of this unit, Students are able to

1. Explore the concepts of Distributed Systems.
2. Analyze the Various System models in distributed systems.

UNIT-II: **12 periods**

Inter process Communication: Introduction, The API for the Internet Protocols- The Characteristics of Inter process communication, Sockets, External Data Representation and Marshalling, Client Server Communication, Group Communication- IP Multicast- an implementation of group communication.

Learning Outcomes: At the end of this unit, Students are able to

1. Analyze the concept of Inter process communication.
2. Summarize the concept of client server communication.

UNIT-III: **12 periods**

Distributed Objects and Remote Invocation: Introduction, Communication between Distributed Objects- Object Model, Distributed Object Model, Design Issues for RMI, Implementation of RMI, Distributed Garbage Collection, Remote Procedure Call, Events and Notifications, Case Study: JAVA RMI

Learning Outcomes: At the end of this unit, Students are able to

1. Determine the design issues and implementation of RMI.
2. Summarize the concept of distributed garbage collection and remote procedure call.

UNIT-IV: **12 periods**

Distributed File Systems: Introduction, File Service Architecture, Peer-to-Peer Systems: Introduction, Napster and its Legacy, Peer-to-Peer Middleware, Routing Overlays. Coordination and Agreement: Introduction, Distributed Mutual Exclusion, Elections, Multicast Communication.

Learning Outcomes: At the end of this unit, Students are able to

1. Demonstrate the architecture of file systems and build knowledge on peer-to-peer systems.
2. Identify the various algorithms of Distributed mutual exclusion

UNIT-V: **12 periods**

Transactions & Replications: Introduction, System Model and Group Communication, Concurrency Control in Distributed Transactions, Distributed Dead Locks, Transaction Recovery, Replication-Introduction, Passive (Primary) Replication, Active Replication.

Learning Outcomes: At the end of this unit, Students are able to

1. Determine the concurrency control in distributed transactions.
2. Explore the concept of active and passive replication

TEXT BOOKS:

1. Ajay D Kshemkalyani, Mukesh Sighal, “Distributed Computing, Principles, Algorithms and Systems”, Cambridge.
2. George Coulouris, Jean Dollimore, Tim Kindberg, “Distributed Systems- Concepts and Design”, Fourth Edition, Pearson Publication.

REFERENCE BOOKS:

1. Advanced Concepts in Operating Systems, Makes Singhal and Niranjan G.Shivaratna, Tata McGraw Hill Edition.

WEB RESOURCES:

1. https://www.coursera.org/learn/distributed-programming-in-java
2. https://www.edx.org/course/javacheng-xu-she-ji-java-programming-pekingx-04830340x
3. https://www.coursera.org/courses?languages=en&query=java

BIG DATA ANALYTICS

(Professional Elective – I)

23CS5113	**Credits: 3**
Instruction**:** 3 Periods / Week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Prerequisites:

- Big Data Analytics requires a strong foundation in several key areas to ensure effective data processing, analysis, and Storage Technology.

Course Objectives:

The course should enable the students:

1. Learn the concepts of Big Data and Hadoop.
2. Exploring Hadoop Distributed File system
3. Data Processing Frameworks
4. Explore the fundamentals of storage technology

Course Outcomes:

Students will be able to.

1. Gain insights into Big Data and Hadoop
2. Analyze about Hadoop Distributed File system
3. Perform Data Processing Frame work
4. Develop the storing and analysis process
5. Analyze about storage technology.

Mapping of Course Outcomes with Program Outcomes:

Mapping		**PO**												**PSO**	
		1	2	3	4	5	6	7	8	9	10	11	12	1	2
CO	1	1	2	-	1	1	-	-	-	2	-	1	1	-	1
	2	2	2	-	2	1	2	-	-	1	-	1	1	-	2
	3	-	2	-	2	2	-	-	-	1	-	1	1	-	3
	4	1	2	-	1	1	-	-	-	2	-	1	1	-	3
	5	3	3	2	2	2	-	-	-	2	-	1	-	-	3

COURSE CONTENTS

UNIT-I: **12 periods**

Introduction To Big Data and Hadoop: Definition and Characteristics of Big Data, History Of Hadoop, Apache Hadoop, Analyzing Data with Hadoop, Hadoop Streaming, Hadoop Echo System. Introduction to Data Processing Architectures (Batch Processing, Real Time Processing, Stream Processing)

Learning Outcomes:

1. Gain insights into Big Data and Hadoop.
2. Articulate about the Architecture system of Data Processing.

UNIT-II: **12 periods**

HDFS (Hadoop Distributed File System): The Design of HDFS, HDFS Concepts, Command Line Interface, Hadoop file system interfaces, Data flow, Data Ingest with Flume and Scoop and Hadoop archives, Hadoop I/O: Compression, Serialization, Avro and File-Based Data structures.

Learning Outcomes:

1. Analyze about the Hadoop Distributed File Systems.
2. Acquire about the Apache Spark.

.

UNIT-III: **12periods**

Data Processing Frameworks: Apache Spark: RDDs (Resilient Distributed Datasets), Spark SQL, Data Frames & Datasets, Spark Streaming (Real-time Processing).

Learning Outcomes:

1. Approximate the role of Apache Spark.
2. Handle Real Time Processing data.

UNIT-IV: **12 periods**

Storing and Analysis process: Clusters, File systems and distributed file systems, NoSQL, Sharding and replication, CAP theorem, ACID, BASE, Case study example. Processing concepts: Parallel and distributed data processing, Hadoop, processing workloads, clusters, Map reduce, case study example

Learning Outcomes:

1. Define about file system.
2. Analyze about the Processing concepts.

UNIT-V: **12 periods**

Storage technology: On-disk storage devices, NoSQL databases, In-memory storage devices, Case study example

Learning Outcomes:

1. Analyze and develop storage technology.
2. Adapt work with a case study.

TEXT BOOKS

1. "Big Data Fundamentals: Concepts, Drivers & Techniques" Thomas Erl, Wajid Khattak, Paul Buhler1st Edition 2016
2. Tom White “Hadoop: The Definitive Guide” Third Edit on, O’reily Media, 2012.

REFERENCE BOOKS

1. "Big Data: Principles and Best Practices of Scalable Real-Time Data Systems"
   Nathan Marz and James Warren,1st Edition, 2013
2. "Big Data Fundamentals: Concepts, Drivers & Techniques", Thomas Erl, Wajid Khattak,
3. and Paul Buhler,1st Edition, 2016
4. "Spark: The Definitive Guide: Big Data Processing Made Simple", Bill Chambers and
5. Matei Zaharia,1st Edition,2018
6. R1 A Semantic Web Premier by Grigoris Antoniou, 2nd edition, MIT Press

WEB RESOURCES:

1. https://onlinecourses.nptel.ac.in/noc25_cs17/unit?unit=20&lesson=45
2. https://aws.amazon.com/sagemaker/?gclid=CjwKCAiAnpy9BhAkEiwA-P8N4i1pm9L1CFlLc9-ZQDZuh4kyPLKlPDmDKbinR0uRiV24gULrUevXyhoCy-AQAvD_BwE&trk=2c2d1ca6-4b04-48e8-9671-314f39728980&sc_channel=ps&ef_id=CjwKCAiAnpy9BhAkEiwA-P8N4i1pm9L1CFlLc9-ZQDZuh4kyPLKlPDmDKbinR0uRiV24gULrUevXyhoCy-AQAvD_BwE:G:s&s_kwcid=AL!4422!3!641436920373!p!!g!!analytic%20tools!19256168540!144671882779
3. https://www.analyticsvidhya.com/resources-big-data

ADVANCED DATA STRUCTURES (Professional Elective – I)	
23CS5114	**Credits: 3**
Instruction: 3 periods /Week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Pre-requisites:

- Knowledge of Data structures.
- knowledge of any programming languages (such as C, C++,Java).

Course Objectives:

1. Explore various advanced data structures, including skip lists, hash tables, priority queues, balanced search trees, and graphs.
2. Give the advantages and dis-advantages of each of the Advanced Data Structure.
3. Learn how to apply algorithm design techniques and data structures to solve problems.
4. Learn different external sorting techniques and analyze their efficiency.

Course Outcomes (CO):

Students will be able to:

1. Implement the ADTs like linear list, skip list and hash tables and their operations
2. Design and apply binary heaps, leftist heaps, Binomial queues and sorting for solving the real-world scenarios
3. Describe the methods to balance a binary search tree using Rotation methods, Color changing methods, splitting, and merging of nodes.
4. Apply Algorithms for solving problems by using string matching techniques
5. Solve problems using graph algorithms such as unweighted shortest path, graphs with negative edge cost.

Mapping of course outcomes with program outcomes:

CO	**PO**												**PSO**	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
1	2	3	2	-	1	-	-	-	-	-	-	1	2	1
2	2	3	3	-	1	-	-	-	-	-	-	1	2	1
3	2	3	2	-	1	-	-	-	-	-	-	1	2	1
4	2	3	2	-	1	-	-	-	-	-	-	1	2	1
5	2	3	2	-	1	-	-	-	-	-	-	1	2	1

COURSE CONTENTS

UNIT-I: **12 Periods**

Skip lists and Hashing: Sets, Map, Dictionaries, representation of dictionary as ADT, Linear list, skip list, hash table representation, an application-text compression using dictionary.

Learning Outcomes:

1. Explain ADT, sets, maps, and dictionaries.
2. Implement lists and hashing techniques.

UNIT-II: **12 Periods**

Priority Queues: Binary heap, applications of binary heap, Applications of priority queues, leftist heaps, Binomial queues.
Sorting: Shell sort, Indirect sorting, bucket sort, External sorting.

Learning Outcomes:

1. Explain the operations of the priority queue.
2. Apply different sorting algorithms

UNIT-III: **12 Periods**

Balanced Search Trees: Red-black trees, Representation of Red-black tree, Insertion, Deletion and searching of nodes in Red-black tree. Splay trees, B-Trees, Indexed Sequential Access Method (ISAM), B-Trees of order m, Representation of B-Tree, Insertion, deletion and searching a node in B-Tree, B+ trees of order m, Representation of B+ -Tree, Insertion, deletion and searching a node in B+-Tree.

Learning Outcomes:

1. Explore balanced search trees.
2. Implement basic operations on the balanced search trees.

UNIT-IV: **12 Periods**

Digital Search Structures: Digital Search trees, Binary Tries and Patricia, Multiway Tries.
String Matching: Exact String Matching- Straight forward Algorithms, The Knuth-Morris- Pratt Algorithm, The Boyer-Moore Algorithm.

Learning Outcomes:

1. Explore digital search structures
2. Apply string matching algorithms to solve real time problems

UNIT-V: **12 Periods**

Graphs: Graph algorithms-Topological sorting, shortest-path algorithms- unweight shortest path, graphs with negative edge cost (Bellman–Ford), acyclic graphs, Network flow problems, Applications of BFS, DFS,

Learning Outcomes:

1. Construct the various types of graphs.
2. Apply the graph concepts to solve the real time problems.

TEXTBOOKS:

1. Data Structures, Algorithms, and Applications in C++, Second Edition " by Sartaj Sahni, published by University Press (Unit I, Unit II & Unit III)
   Data Structures and Algorithms in C++" by Adam Drozdek, Fourth Edition, published by Cengage Learning (Unit I, Unit IV & Unit V)

REFERENCE BOOKS:

1. "Data Structures: A Pseudocode Approach with C, Second Edition" by Richard F. Gilberg and Behrouz A. Forouzan, published by Cengage Learning.
2. Data Structures and Algorithm Analysis in C++, Fourth Edition by Mark Allen Weiss, Pearson Education.

ARTIFCIAL INTELLIGENCE (Professional Elective-I)	
23CS5115	**Credits: 3**
Instruction**:** 3 Periods / Week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Prerequisites:

- Mathematics, Programming and complex problem-solving techniques

Course Objectives:

1. To discuss about the Basic principles, techniques, and applications of artificial intelligence.
2. To analyze and apply the insights into knowledge representation, problem-solving, and learning methods in Science and Engineering domains.
3. To acquire knowledge of various searching techniques and linked list, trees and graphs.
4. To gain the knowledge of propositional and predicate calculus.
5. To analyze the importance of expert systems and its applications.

Course Outcomes:

Students will be able to.

1. Illustrate foundations of Artificial Intelligence (AI) and its applications.
2. Apply Search Techniques (Brute-Force and Heuristic) of Artificial Intelligence.
3. Interpret various ways of Knowledge Representation with examples.
4. Acquire the knowledge of Propositional and Predicate Calculus with planning attributes.
5. Illustrate the concepts of Uncertainty and Expert systems with respect to real-time case studies.

Mapping of Course Outcomes with Program Outcomes:

Mapping		**PO**												**PSO**	
		1	2	3	4	5	6	7	8	9	10	11	12	1	2
CO	1	2	2	1	2	-	2	2	-	1	1	-	-	1	1
	2	2	1	2	3	-	1	-	-	2	2	1	-	2	-
	3	2	2	1	2	-	-	1	-	2	2	-	1	1	-
	4	1	1	2	3	-	1	-	-	1	2	2	-	2	1
	5	2	2	2	2	-	-	1	-	2	1	1	1	1	1

CONTENTS

UNIT-I: **12 periods**

Foundations of Artificial Intelligence: AI problems, foundation of AI and history of AI intelligent agents: Agents and Environments, the concept of rationality, the nature of environments, structure of agents, problem solving agents, problem formulation, AI Techniques, Problem Types and Characteristics, State Space Search, Production Systems and its characteristics, Applications of Artificial Intelligence.

Learning Outcomes: At the end of this unit, Student will be able to:

1. Acquire the knowledge of AI and its applications.
2. Acquire the knowledge of problem solving agents and types.

UNIT-II: **12 periods**

Searching: Searching- Searching for solutions, uniformed search strategies – Breadth first search, depth first Search, Bi-Directional Search and Uniform-Cost Search. Informed Search Algorithms (Heuristic search): Introduction, Heuristic evaluation function, Generate-and-Test, Best-First Search, A* Algorithm, Problem Reduction Algorithm, AO* Algorithms, Hill climbing, Simulated Annealing, Constraint Satisfaction Algorithm (CSP).

Learning Outcomes:

1. Apply Uninformed search algorithms on AI problems
2. Apply Heuristic and search techniques on selective AI Problems.

UNIT-III: **12 periods**

Game Playing - Adversarial search, Games, mini-max algorithm, optimal decisions in multiplayer games, Problem in Game playing, Alpha-Beta pruning, Evaluation functions.

Knowledge and Reasoning: Knowledge Representation Issues: Representations and Mappings, Approaches to Knowledge Representation, Issues in Knowledge Representation;

Learning Outcomes:

1. Interpret the game playing technique with real world problems.
2. Identify the issues in Knowledge representation and reasoning.

UNIT-IV: **12 periods**

Predicate Logic: Representing Simple Facts in Logic, Representing Instance and ISA Relationships, Computable Functions and Predicates, Resolution, Natural Deduction. Representing Knowledge Using Rules: Procedural versus Declarative Knowledge, Logic Programming, Forward versus Backward Reasoning, Matching, Control Knowledge.

Planning: Overview, An Example Domain-The Blocks World, Components of a Planning System, Goal Stack Planning, Hierarchical Planning.

Learning Outcomes:

1. Apply Propositional and Predicate logic through Knowledge Representation and case studies.
2. Apply Planning on real-world problems such as Block-World, Wearing a shoe, etc.

UNIT-V: **12 periods**

Probabilistic Reasoning: Statistical Reasoning, Probability and Bayes' Theorem, Bayesian Networks, Dempster-Shafer Theory.
Expert systems: Introduction, basic concepts, structure of expert systems, the human element in expert systems, how expert systems works, problem areas addressed by expert systems, expert systems success factors, types of expert systems and Applications.

Learning Outcomes:

1. Solve Uncertainty issues with suitable examples.
2. Illustrate foundations of Expert Systems and how to apply in real-time Scenario.

TEXT BOOKS

1. "Artificial Intelligence: A Modern Approach", by Stuart J. Russell and Peter Norvig, 4th Edition, Prentice Hall series of Artificial Intelligence, 2020.
2. "Artificial Intelligence", by Elaine Rich, Kevin Knight, Shivashankar B. Nair, McGraw Hill, 3rd Edition, 2009.

REFERENCE BOOKS

1. "Artificial Intelligence: Foundations of Computational Agent", by David L Poole, Alan K. Mackworth, Cambridge University Press, 2017.
2. "Artificial Intelligence, Structures, Strategies for Complex Problem Solving", by George F Luger, Addison-Wesley Publishing Company, United States, 6th Edition, 2009.

SOFTWARE ENGINEERING (Professional Core)	
23CS4119	**Credits: 3**
Instruction**:** 3 Periods / Week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Pre-requisites:

- Basic mathematical knowledge
- Basic Knowledge of Programming Fundamentals and Problem Solving
- Basic knowledge of procedural and object-oriented programming.

Course Objectives:
The Course should enable the students:

1. To Analyze, Design, Develop, and Maintain Software Projects
2. To provide knowledge on software Life Cycle and Development Models with object-oriented concepts.
3. To explain the role of UML and Testing in Software Development.
4. To avoid risks by using Risk Management Techniques.

Course Outcomes (CO):
Students will be able to

1. Analyze the different software process models and their significance
2. Interpret the functional, non-functional requirements and requirement Engineering Process.
3. Choose the Architecture for a given software application & Design UML diagrams.
4. Demonstrate skills in Object-oriented Modeling and Plan software project management activities
5. Identify the Testing Strategies and design test suits for the given Test Scenarios

Mapping of Course Outcomes with Program Outcomes:

Mapping		**PO**												**PSO**	
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
CO	**1**	2	2	-	-	-	-	-	-	-	1	-	-	-	1
	2	2	3	2	-	-	-	-	-	1	2	-	-	-	1
	3	2	2	2	-	-	-	-	-	1	2	-	-	-	3
	4	2	2	3	2	1	-	-	-	2	2	2	1	-	3
	5	2	2	3	2	1	-	-	-	2	2	2	1	-	3

COURSE CONTENTS

UNIT-I **12 periods**

Importance of Software Engineering: Problem Solving Activity, Modelling Activity, Knowledge Acquisition activity, Rationale Driven Activity, Umbrella Activities, Software Engineering Challenges, Software Development Life Cycle.

Process Models: Waterfall Model, Incremental Model, Prototype Model, V Model, Spiral Model, RapidModel, Agile Model.

Learning Outcomes: At the end of this unit, Student will be able to

1. List Various Software Development activities.
2. Classify the Working of Software Process Models.

UNIT-II: **12 periods**

Software Requirement Analysis & Specification: Need for SRS, Characteristics of Requirements, Functional Requirements, Non-Functional Requirements, Requirement Specification Document - IEEE Format.

Function Oriented Design: Structured Design Methodology –DFD Diagram, Design Principles, Module- Level Concepts- Cohesion & Coupling.

Learning Outcomes: At the end of this Unit the student will be able to:

1. Find Functional and Non-functional Requirements and prepare SRS Document.
2. Design Data flow in Functional Approach (DFD).

UNIT-III: **12 periods**

Software Architecture: Architecture Views, Architectural Styles for C & C View- Pipe and Filter, Client and Server, Shared Data.

Object Oriented Design: Objects and Object classes, Object Oriented Design Process, Design Evolution.

Overview of UML: Building Blocks of UML (Things, Relationships, Diagrams), Class Diagram, Class Relationships, Use Case Scenario, Use Case Diagram, Relationships among Use Cases.

Interaction Diagrams: Sequence Diagram, Collaboration Diagram.

Learning Outcomes: At the end of this Unit the student will be able to:

1. Make use of Various Software Architectures.
2. Design UML Diagrams with Object Oriented Approach.

UNIT-IV: **12 periods**

UML Diagrams: Activity Diagram, State Chart Diagram, Component Diagram & Deployment Diagram. Object Oriented Methodologies: Unified Methodology, Rambaugh Methodology, Booch Methodology.

Software Testing Strategies: A Strategic Approach to Software Testing, Test Strategies for Conventional Software and Object-Oriented Software, Validation Testing, White- Box Testing, Basis Path Testing, Black-Box Testing, System Testing.

Learning Outcomes: At the end of this Unit the student will be able to:

1. Illustrate Various object-oriented Methodologies.
2. Design Test Suites for Test Scenarios.

UNIT-V: **12 periods**

Product Metrics: A Framework for Product Metrics, Metrics for Requirements Model, Metrics for Design Model, Metrics for Source Code, Metrics for Testing, Metrics for Maintenance.
Software Project Management: Definition of Software Project, Need of Software Project Management, Software Project management Plan, Effort Estimation, Case Study on Effort Estimation with COCOMO, Risk Management Plan, Project Tracking & Control.

Learning Outcomes: At the end of this Unit the student will be able to:

1. Summarize various Software Testing Techniques.
2. Adapt the approaches in Software Project Management to Develop the Software Project

TEXT BOOKS:

1. Roger S Pressman, Software Engineering: A Practitioner's approach, Tata McGraw Hill Education,9th edition,2023 (Unit1,2)
2. Timothy C. Lethbridge, "Object Oriented Software Engineering" (Practical Software Development using UML and Java" Tata McGraw-Hill, 2nd Edition, 2019. (unit 3,4)
3. Booch, Maksimchuk, Engle, Young, Conallen and Houston, "Object Oriented Analysis and Design with Applications", Pearson Education, 3rd Edition, 2009. (unit 4)
4. Pankaj Jalote, An integrated Approach to Software Engineering, Springer, 3rd edition, 2005 (Unit 5).

REFERENCE BOOKS:

1. Ivar Jacobson, "Object Oriented Software Engineering", Pearson, 2009.
2. Rumbaugh et. al, "Object Oriented Modeling and Design", Pearson.
3. Bertrand Meyer, Object-Oriented Software Construction, Prentice Hall, 2nd edition, 1998.
4. Edwards Yourdon, Carl Argila,"Case Studies in object oriented analysis and design" Prentice Hall.

WEB REFERENCES:

1. http://nptel.iitm.ac.in/courses/106101061/
2. http://nptel.iitm.ac.in/courses/Webcoursecontents/ IIT%20Kharagpur/Soft%20Engg/New_index1.html

COMPILER DESIGN (Professional Core)	
23CS4120	**Credits: 3**
Instruction: 3 Periods / Week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Pre-requisites:

- Basic fundamentals of Discrete Mathematics, Basic structure of programming languages and Principles of Automata Theory.

Course Objectives:

The course should enable the students to:

1. Introduce the major concept areas of language translation and compiler design.
2. Learn the design of lexical analyzer, syntax analyzer.
3. Enrich the knowledge in various phases of compiler and its use, intermediate code generation, optimization techniques, machine code generation, and use of symbol table.
4. Provide practical programming skills necessary for constructing a compiler

Course Outcomes (CO):

Students will be able to:

1. Analyse the various phases of a compiler and tools of a Lexical analyser.
2. Apply Top-down parsing Techniques for various types of grammars.
3. Identify the differences in functioning of various bottom-up parsing Techniques.
4. Differentiate the different intermediate code forms and apply a variety of optimization techniques to improve the code.
5. Compare different code generation techniques and how symbol table, run time storage and error detecting are managed in various phases of a compiler.

Mapping of Course Outcomes with Program Outcomes:

CO	**PO**												**PSO**	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
1	1	2	-	1	-	-	-	-	-	-	-	1	-	-
2	2	1	2	2	-	-	-	-	-	-	-	1	-	-
3	2	3	-	2	-	-	-	-	-	-	-	2	-	-
4	2	3	-	2	-	-	-	-	-	-	-	1	-	-
5	3	3	-	3	-	-	-	-	-	-	-	2	-	-

COURSE CONTENTS

UNIT-I : 12 Periods

Introduction to Compiler: Introduction to Language processing System, Cousins of the Compiler, Analysis-synthesis model of compilation, Phases of a compiler, Compiler construction tools. **Lexical analysis:** Role of Lexical Analyzer, Input buffering, Specification & Recognition of Tokens, Design of a Lexical Analysis Generator.

Learning Outcomes:

1. Explain the process of Compilation i.e., conversion from High level language to low-level language based on different phases of compiler.
2. Explain the Phases of compiler and implementation of Lexical Analyzer phase

UNIT-II: 12 Periods

Top-down Parsers: Problems with Top-Down Parsers, Recursive Descent Parser, Left Recursion, Left Factoring, Ambiguity, LL (1) Or Non-Recursive Descent Parser.

Learning Outcomes:

1. Identify the problems of Top-down Parser and explain the working of Top-down Parser
2. Explain the working procedure of syntax analysis phase with help of Top-Down Parsing methods using Top-Down Parsers

UNIT-III: 12 Periods

Bottom-up parser: Introduction to LR Parsing, Shift Reduce parser, Operator Precedence Parser, LR parser: LR (0), SLR, CLR parser, LALR parser. Parser Generator - YACC.

Learning Outcomes:

1. Identify different Bottom- up Parsers and implement Syntax analysis phase by using bottom-up parsers.
2. Explain the working procedure of syntax analysis phase with help of Bottom-up Parsing methods on various grammars.

UNIT-IV: 12 Periods

Syntax Directed Translation: Syntax Directed Definitions and Translation, Application of SDT and SDT schemes. **Intermediate code generation:** Different Intermediate code forms and implementation of TAC. **Code Optimization:** Classification of optimization, Principal sources of optimization, Basic blocks and flow graphs, Optimization of basic blocks, Loops and Local optimization, Global data flow analysis.

Learning Outcomes:

1. Describe different semantic analysis techniques.
2. Illustrate different representations of intermediate code and machine-independent code optimization methods.

UNIT-V: **12 Periods**

Code Generation: Issues in the design of code generator, register allocation, and assignment, DAG representation of basic blocks, Peephole Optimization, code generation using DAG.
Storage Organization and Error Recovery: Symbol tables, Run-time storage administration, Stack, Activation record, Error detecting and Reporting in various Phases.

Learning Outcomes:

1. Analyse the techniques involved in code generation and machine-dependent optimizations.
2. Illustrate how the symbol table functions, along with error detection and recovery in compilation.

TEXT BOOK:

1. Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman “Compilers: Principles, Techniques, and Tools” (Updated Second Edition, 2023). - (UNIT- I to III).
2. Keith D. Cooper and Linda Torczon "Engineering a Compiler, Third Edition, 2023” - (UNIT IV & V)

REFERENCE BOOKS:

1. Randy Allen, Ken Kennedy, Optimizing Compilers for Modern Architectures: A Dependence based Approach, Morgan Kaufmann Publishers, 2002.
2. Steven S. Muchnick, Advanced Compiler Design and Implementation‖, Morgan Kaufmann Publishers – Elsevier Science, India, Indian Reprint 2003.
3. Keith D Cooper and Linda Torczon, Engineering a Compiler‖, Morgan Kaufmann Publishers Elsevier Science, 2004.
4. V. Raghavan, Principles of Compiler Design‖, Tata McGraw Hill Education Publishers, 2010.
5. Allen I. Holub, Compiler Design in C‖, Prentice-Hall Software Series, 1993.

WEB RESOURCES:

1. http://nptel.ac.in/courses/106104123/.
2. https://www.youtube.com/playlist?list=PLEbnTDJUr_IcPtUXFy2b1sGRPs
   LFMghhS.
3. http://www.nptelvideos.in/2012/11/compiler-design.html.
4. https://www.udacity.com/course/compilers-theory-and-practice--ud16

COMPETITIVE PROGRAMMING (Professional Core)	
23CS4121	**Credits:3**
Instruction: 3 Periods / Week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Pre-requisites:

- Problem solving techniques and Data Structures

Course Objectives:

1. Develop and implement advanced algorithms
2. Develop the skills required for programming competitions.
3. Learn to select appropriate algorithms for a given problem, integrate multiple algorithms
4. Solving a complex problem, designing new algorithms and implementing them.
5. Solving problems in teams and working under time pressure.

Course Outcomes (CO):

Students will be able to

1. Identify type of the problem and Apply sorting, searching techniques to solve problems.
2. Perform bit manipulation and string manipulation operations.
3. Compute the given complex problem using mathematical theorem and graph algorithms.
4. Solve given problems using dynamic programming and Greedy Approach.
5. Analyze and develop backtracking algorithms and Geometrical Algorithms.

Mapping of Course Outcomes with Program Outcomes:

CO	PO												PSO	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
1	2	3	1	3	-	-	-	-	-	-	-	3	2	-
2	3	2	1	3	-	-	-	-	-	-	-	3	3	-
3	3	3	2	3	-	-	-	-	-	-	-	3	3	-
4	3	3	3	3	-	-	-	-	-	-	-	3	2	-
5	3	2	1	2	-	-	-	-	-	-	-	3	1	-

COURSE CONTENTS

UNIT-I **12 Periods**

Problem Solving Paradigms: Overview and Motivation, Complete Search, Divide and Conquer, Greedy, Dynamic Programming, Brute force. **Sorting & Searching**: Sorting Theory, Counting Sort, Radix Sort, Ternary Search.

Learning Outcomes:

1. Identify the type of the problem statement.
2. Apply sorting and searching techniques to solve the given problem.

UNIT -II **12 Periods**

Bit Manipulation: Bit representation, bit operations, representing sets. **Strings:** String terminology, Trie structures, Trie insertion, Deletion, Types of Tries String hashing, Z-algorithm

Learning Outcomes:

1. Apply the bit operators to solve given problem.
2. Implement various string operations and patter matching algorithms.

UNIT-III **12 Periods**

Number theory: Primes and factors, Modular arithmetic **Graphs:** Flows and Cuts: Ford-Fulkerson algorithm, Path Covers. **Paths, and circuits**: Eulerian paths.

Learning Outcomes:

1. Solve the complex problem using mathematical theorem.
2. Analysing Graph algorithms to solve suitable problems.

UNIT- IV **12 Periods**

Dynamic programming: Coin problem, longest increasing subsequence, Paths in a grid, edit distance, Counting Tiling. **Greedy Algorithms:** Coin problem, Tasks and deadlines, Minimizing sums

Learning Outcomes:

1. Solve complex problem using dynamic programming approach.
2. Solve complex problem using Greedy approach.

UNIT- V **12 Periods**

Computational Geometry: Points and lines, Polygon (Convex, Concave), Distance functions.
Backtracking: Constructing All Subsets, Constructing All Permutations.
Learning Outcomes:

1. Implement Geometrical algorithms.
2. Analyze and develop backtracking algorithms.

TEXT BOOK:

1. Competitive programming 3, by Steven Halim, Felix Halim, Handbook for ACM ICPC and IOI contestants.

REFERENCE BOOKS:

1. Data Structures and Algorithms Made Easy by Narasimha Karuma chi, Career Monk.
2. Programming challenges, by Steven S. Skiena Miguel A. Revilla, The Programming Contest Training Manual, Springer.
3. Competitive Programmer’s Handbook, by Antti Laaksonen, Code Submission Evaluation System.

COMPETITIVE PROGRAMMING LAB (Professional Core)	
23CS4221	**Credits: 1.5**
Instruction: 3 periods / week	Sessional Marks: 50
End Exam: 3 Hours	End Exam Marks: 50

Prerequisites:

- C and data Structures Design and Analysis of Algorithms

Course objectives:

1. Learn to select appropriate algorithm for a given problem, integrate multiple algorithms
2. Solve a complex problem, design new algorithms, and implement them.

Course outcomes (CO):

1. Apply the basic, sorting and searching techniques to solve the problem components etc.
2. Analyze the concepts of path finding algorithms for flows and cuts, strings and greedy algorithms
3. Develop solutions for the back tracking algorithms and bit manipulations
4. Solve the number theory and knowledge of dynamic programming to the real time scenario.

Mapping of Course Outcomes with Program Outcomes:

CO	PO												PSO	
	1	2	3	4	5	6	7	8	9	10	11	12	1	2
1	2	3	1	3	-	-	-	-	-	-	-	1	2	-
2	3	2	1	3	-	-	-	-	-	-	-	1	3	-
3	3	3	2	3	-	-	-	-	-	-	-	1	3	-
4	3	3	3	3	-	-	-	-	-	-	-	1	2	-

LIST OF EXPERIMENTS

Basic Optimization Problems:

1. Maximum sub array sum **CO1**

Solve the Maximum sub array sum with different time complexities.

Divide and Conquer:

2. The Median of Two Sorted Arrays **CO1**

Program to find the median of two sorted arrays of same size and different size are discussed here. Firstly, let us see what is median of the array? Median is an element which divides the array into two parts - left and right. So the number of elements on the left side of the array will be equal or less than the number of elements on the right side. Now, let us consider the case of an array with odd number of elements. Array = [9,11,16,7,2] Sorted array = [2,7,9,11,16]. In this case, the median of this array is 9, since it divides the array into two parts: [2,7] and [11,16]. Further, let us consider the case of an array with even elements. Array = [1,2,3,4,5,6]. In such a case, we will take the average between the last element of the left part and the first element of the right part. In this case, the median equals = (3 + 4) / 2 = 3.5.

Input Format

The input should contain 3 lines.

1. First line of the input should contain two integer values which specify the number of elements in array1 and array2.
2. Second line of the input should contain the elements of the first array.
3. Third line of the input should contain the elements of the second array.

Constraints

All elements must be Integers

Output Format

The output should print only the median value.

Sample Input 1

```
5 6
-5 3 6 12 15
-12 -10 -6 -3 4 10
```

Sample Output 1

```
3
```

Sample Input 2

```
4 6
2 3 5 8
10 12 14 16 18 20
```

Sample Output 2

```
11
```

3. Division with Binary Search **CO1**

We can modify binary search algorithm to perform division of two numbers, by defining range [0, infinity] which serves as initial low and high for the binary search algorithm. Now we need to find a mid that satisfies x/y = mid or x*mid for given two numbers x and y. Based on the comparison result based on x and y*mid, we either update low, update high or return mid. 1. If y*mid almost equal to x, we return mid. 2. If y*mid is less than x, we update low to mid 3. If y*mid is more than x, we update high to mid We need to care about division by zero and sign of the result etc.

Input: one line of input should contain two numbers separated by space.

Output: should print division of the numbers as a result.

Input Format

Input: one line of input should contain two numbers separated by space. Input1: 22 7

Constraints

x,y < infinity

Output Format

Output: should print division of the numbers as a result.

Sample Input 1

22 7

Sample Output 1

3.14286

<u>Computational Problems:</u>

4. Find the Triplet **CO1**

Given an array of integers, find a triplet having maximum product in the array.

Input: First line of input should specify the number of elements in the array. Second line of input should specify each element separated by space.

Output: should print Triplets.

Test Cases:

Sample Input 1

5

-4 1 -8 9 6

Sample Input 1

-4 -8 9

Sample Input 2

5

1 7 2 -2 5

Sample Input 2

7 2 5

5. **3n+1** **CO1, CO4**

Consider the following algorithm to generate a sequence of numbers. Start with an integer n. If n is even, divide by 2. If n is odd, multiply by 3 and add 1. Repeat this process with the new value of n, terminating when n = 1. For example, the following sequence of numbers will be generated for n = 22: 22 11 34 17 52 26 13 40 20 10 5 16 84 2 1 It is conjectured (but not yet proven) that this algorithm will terminate at n = 1 for every integer n. Still, the conjecture holds for all integers up to at least 1, 000, 000. For an input n, the cycle-length of n is the number of numbers generated up to and including the 1. In the example above, the cycle length of 22 is 16. Given any two numbers i and j, you are to determine the maximum cycle length over all numbers between i and j, including both endpoints. The input will consist of a series of pairs of integers i and j, one pair of integers per line. All integers will be less than 1,000,000 and greater than 0. Output For each pair of input integers i and j, output i, j in the same order in which they appeared in the input and then the maximum cycle length for integers between and including i and j. These three numbers should be separated by one space, with all three numbers on one line and with one line of output for each line of input.

Sample Input	**Sample Output**
1 10	1 10 20
100 200	100 200 125
201 210	201 210 89
900 1000	900 1000 174

6. Euclid Problem **CO4**

From Euclid, it is known that for any positive integers A and B there exist such integers and Y that AX + BY = D, where D is the greatest common divisor of A and
B. The problem is to find the corresponding X, Y , and D for a given A and B.

Input The input will consist of a set of lines with the integer numbers A and B, separated with space (A, B < 1, 000, 000, 001).

Output For each input line the output line should consist of three integers X, Y , and D, separated with space. If there are several such X and Y , you should output that pair for which X ≤ Y and |X| + |Y | is minimal.

Sample Input 1

4 6
17 17

Sample Output 1

-1 1 2
0 1 17

7. Tug of War CO3

Tug of war is a contest of brute strength, where two teams of people pull in opposite directions on a rope. The team that succeeds in pulling the rope in their direction is declared the winner. A tug of war is being arranged for the office picnic. The picnickers must be fairly divided into two teams. Every person must be on one team or the other, thenumber of people on the two teams must not differ by more than one, and the total weightof the people on each team should be as nearly equal as possible. Input

The input begins with a single positive integer on a line by itself indicating the number oftest cases following, each described below and followed by a blank line.

The first line of each case contains n, the number of people at the picnic. Each of the next n lines gives the weight of a person at the picnic, where each weight is an integer between 1 and 450. There are at most 100 people at the picnic. Finally, there is a blank line between each two consecutive inputs.

Output For each test case, your output will consist of a single line containing two numbers: the total weight of the people on one team, and the total weight of the people onthe other team. If these numbers differ, give the smaller number first. The output of each two consecutive cases will be separated by a blank line.

Sample Input 1

```
1
3
100
90
200
```

Sample Output 1

```
               1900
```

String Matching Problems:

8.Matching –String with wild card –pattern. CO2

Check the given string is matches with pattern containing wild card characters („*‟ and „?‟), where the „*‟ can match to any number of characters including zero characters and „?‟ can match to any single character in the given input string. Check if the given input string is matches with given input pattern or not.

Input: Input should contain two lines.

First line of input should contain input string. Second line of input should contain pattern string.

Output: The output should print either „0‟ or „1‟.

‘1’ in the output indicates that the given string is matches with the given pattern.

‘0’ in the output indicates that the given string is not matched with the given pattern.

Sample Input 1: abcabcccda?c*d

Sample Output1:1

Sample Input 2: abcabc ccd a?c*c

Sample Output2 : 0

Grid Based Problems :

9.Rotten Oranges **CO2**

Given a grid of dimension **nxm** where each cell in the grid can have values 0, 1 or 2 which has the following meaning:

0 : Empty cell

1 : Cells have fresh oranges

2 : Cells have rotten oranges

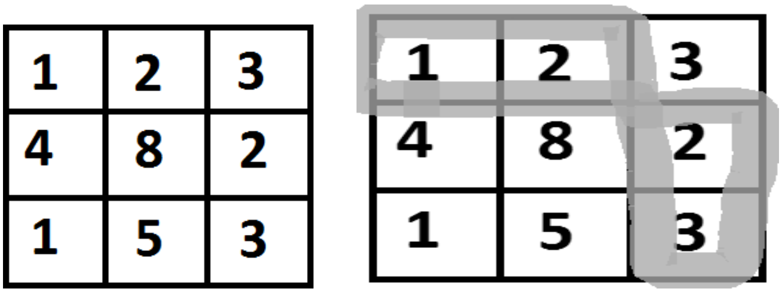
We have to determine what is the minimum time required to rot all oranges. A rotten orange atindex [i,j] can rot other fresh orange at indexes [i-1,j], [i+1,j], [i,j-1], [i,j+1]
(**up**, **down**, **left** and **right**) in unit time.

Example 1:

Input: grid = {{0,1,2},{0,1,2},{2,1,1}}
Output: 1
Explanation: The grid is-
0 1 2
0 1 2
2 1 1
Oranges at positions (0,2), (1,2), (2,0)
will rot oranges at (0,1), (1,1), (2,2) and
(2,1) in unit time.

10. Minimum-Cost –Path **CO2**

Find a path in an n×n grid from the upper-left corner to the lower-right corner such that we only move down and right and diagonally lower cells from a given cell, i.e., from a given cell (i, j), cells (i+1, j), (i, j+1) and (i+1, j+1) can be traversed. Assume that all costs are positive integers. Each square contains a number, and the path should be constructed sothat the sum of numbers along the path is as small as possible.



1	2	3
4	8	2
1	5	3

1	2	3
4	8	2
1	5	3

The path is (0, 0) –> (0, 1) –> (1, 2) –> (2, 2). The cost of the path is 8 (1 + 2 + 2 + 3)

Sample Input 1

1 2 3
4 8 2
1 5 3

Sample Output 1

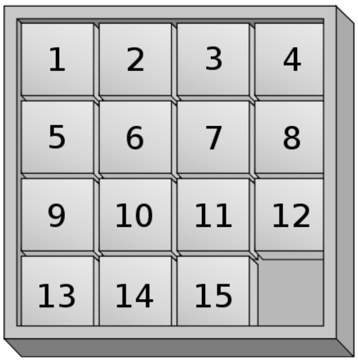
8

11. 15-Puzzle Problem CO3

The 15-puzzle is a very popular game: you have certainly seen it even if you don’t know it by that name. It is constructed with 15 sliding tiles, each with a different number from 1 to 15, with all tiles packed into a 4 by 4 frame with one tile missing. The object of the puzzle is to arrange the e tiles so that they are ordered as below:

The only legal operation is to exchange the missing tile with one of the 2, 3, or 4 tiles it shares an edge with. Consider the following sequence of moves:

We denote moves by the neighbor of the missing tile is swapped with it. Legal values are “R,”“L,” “U,” and “D” for right, left, up, and down, based on the movements of the hole. Given an initial configuration of a 15-puzzle you must determine a sequence of steps that take you to the final state. Each solvable 15-puzzle input requires at most 45 steps to be solved with our judge solution; you are limited to using at most 50 steps to solve the puzzle.



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

Input The first line of the input contains an integer n indicating the number of puzzle set inputs. The next 4n lines contain n puzzles at four lines per puzzle. Zero denotes the missing tile.

Output For each input set you must produce one line of output. If the given initial configuration is not solvable, print the line “This puzzle is not solvable.” If the puzzle is solvable, then print the move sequence as described above to solve the puzzle.

Sample Input 1

```
2
2 3 4 0
1 5 7 8
9 6 10 12
13 14 11 15
13 1 2 4
5 0 3 7
9 6 10 12
15 8 11 14
```

Sample Output 1

```
LLLDRDRDR
This puzzle is not solvable
```

Bitwise Operators:

12. Find first set bit CO3

Problem Statement: Given an integer an **N.** The task is to return the position of **first set bit found from the right side** in the binary representation of the number.

Note: If there is no set bit in the integer N, then return 0 from the function.

Test case 1

Input: N = 18

Output: 2

Test case 2 Input**:**

N = 12

Output**: 3**

Expected Time Complexity: O(log N).

Constraints:

0 <= N <= 10^8

13. Coin Problem CO4

Given a value V. You have to make change for V cents, given that you have infinite supply of each of C{ C1, C2, .. , Cm} valued coins. Find the minimum number of coins to make the change and print the coins that appear in an optimal solution.

Input:

The first line of input contains an integer T denoting the number of test cases.

The first line of each test case is V and N, V is the value of cents and N is the number of coins.

The second line of each test case contains N input C[i], value of available coins.

Output:

Print the coins appear in an optimal solution and in a newline print the minimum number of coins to make the change and, if not possible print "-1".

Constraints:

1 ≤ T ≤ 100

1 ≤ V ≤ 10^6

1 ≤ N ≤ 10^6

1 ≤ C[i] ≤ 10^6

Sample Input 1

1

7 2

2 1

Sample Input 1

2 2 2 1

4

Explanation :

Testcase 1: We can use coin with value 2 three times, and coin with value 1 one times tochange a total of 7.

14.spirally traversing a matrix CO4

Given a matrix of size r*c. Traverse the matrix in spiral form.

Test case 1:

Input:

```
r = 4, c = 4
matrix[][] = {{1, 2, 3, 4},
        {5, 6, 7, 8},
        {9, 10, 11, 12},
        {13, 14, 15,16}}
```

Output:

```
1 2 3 4 8 12 16 15 14 13 9 5 6 7 11 10
```

TEXT BOOKS:

1. "Competitive programming 3" by Steven Halim, Felix Halim, Handbook for ACM ICPC and IOI contestants.
2. "Competitive Programmer's Handbook" by Antti Laaksonen, Code Submission Evaluation System.

REFERENCE BOOKS:

1. "Data Structures and Algorithms Made Easy" by Narasimha Karumanchi, Career Monk.
2. "Programming challenges" by Steven S. Skiena Miguel A. Revilla, The Programming Contest Training Manual, Springer.

SOFTWARE ENGINEERING LAB (Professional Core)	
23CS4219	**Credits: 1.5**
Instruction: 3 periods / week	Sessional Marks: 50
End Exam: 3 Hours	End Exam Marks: 50

Course Objectives:

- To provide working knowledge of UML
- To provide working knowledge of the technologies essentially for incorporating in the project.
- To expertise for testing and document software.
- To present the project in a professional manner

Course Outcomes (CO):

By the end of the course, the student will be able to:	
CO-1	Design DFD, UML Diagrams for the specified software project
CO-2	Write the Software Requirements Document for a specified project
CO-3	Discuss about the project implementation among the team members and improve their professional skills
CO-4	Identify the Testing Strategies and design test suits for the given Test Scenarios.

Mapping of Course Outcomes with Program Outcomes:

Mapping		**PO**												**PSO**	
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
CO	**1**	-	3	3	2	3	2	-	1	1	1	2	2	3	3
	2	-	2	1	1	-	2	1	1	1	2	1	-	2	-
	3	-	2	3	2	-	2	-	2	1	-	1	-	1	-
	4	-	-	-	1	-	-	-	-	3	2	-	1	-	-

SYLLABUS

List of Experiments

Exp No.	Name Of the Experiment	CO
1	System Modelling – DFD Diagrams	2,4
2	UML- UseCase Scenario, UseCase Diagram	1,4
3	Class Diagram	1,4
4	Interaction Diagrams: Sequence and Collaboration Diagrams	1,4
5	State Chart Diagram	1,4
6	Flow of Events and Activity Diagrams	1,4
7	Component and Deployment Diagrams	1,4
8	Software Requirements Specification Document	2,4
9	Design Test Cases	3,4
10	Test Report & Error Report	2,3,4
11	Project - Case Study – Identify Testing Strategies and represent A Case Study using all UML diagrams	1,2,3,4
12	PPT Presentation of their project	1,2,3,4

TEXT BOOKS:

1. Roger S Pressman,Software Engineering: A Practitioner’s approach,Tata McGraw Hill Education,9th edition,2023
2. Pankaj Jalote, An integrated Approach to Software Engineering,2010
3. Timothy C. Lethbridge, "Object Oriented Software Engineering” (Practical Software Development using UML and Java" Tata McGraw-Hill,2nd Edition,2019

REFERENCE BOOKS:

1. Ivar Jacobson, “Object Oriented Software Engineering”, Pearson, 2009.
2. Rumbaugh et. al, “Object Oriented Modeling and Design”, Pearson.
3. Bertrand Meyer, Object-Oriented Software Construction, Prentice Hall, 2nd edition, 1998.
4. Edwards Yourdon, Carl Argila,”Case Studies in object oriented analysis and design” Prentice Hall.

DATA ANALYTICS AND VISUALIZATION (Professional Skill Course)	
23CS9213	**Credits: 1**
Instruction:2 Periods/week	Sessional Marks: 50
EndExam:3 Hours	End Exam Marks:50

Pre-requisites: Basics of statistics and python programming

Course Objectives:

The course should enable the students to:

1. Develop proficiency in handling structured data efficiently using powerful libraries for data manipulation and analysis.
2. Learn techniques to clean, pre-process, and transform raw data into a structured format suitable for analysis.
3. Master the fundamentals of data visualization using various plotting libraries to effectively represent data insights.
4. Explore advanced visualization techniques to communicate complex data trends and geographical distributions intuitively.

Course Outcomes:

Students will be able to

1. Apply various functions using Numpy and pandas on data/information
2. Apply data wrangling tools to format, handle outliers, remove duplicates, and normalize and standardize data
3. Apply various functions of Matplotlib and Seaborn on the complex data
4. Develop advanced data visualization techniques using 2D and 3D plots for the real-time data

Mapping of Course Outcomes with Program Outcomes:

CO	**PO**												**PSO**	
	1	2	3	4	5	6	7	8	9	10	11	12	1	2
1	2	2	3	2	2	-	-	-	-	-	-	-	3	2
2	2	2	2	3	-	3	-	-	3	-	-	-	2	-
3	2	2	3	3	-	3	-	-	3	-	-	-	2	-
4	2	2	-	3	-	2	2	-	-	3	-	3	2	-

S.No	Lab Experiments	CO’s
1	**NumPy:** Creating ndarray, data types, array attributes, indexing, slicing. **Sample Programs:** • Write a Python program to create a one-dimensional, two-dimensional, and three-dimensional NumPy array. Print their shapes and sizes. • .Write a Python program to create a 2D NumPy array and demonstrate indexing and slicing. **Pandas: :** Series, data frame, how to read write CSV and Excel files, indexing, adding columns, aggregations, handling missing data, group by and merging. **Sample Programs:** • Write a Python program to create a Pandas Series and DataFrame from a dictionary • Write a Python program to group data by a column and merge two DataFrames.	**CO-1**
2	**Data Wrangling:** Introduction, Need of data cleanup, data clean up basics. Data wrangling tools with – formatting, outliers, duplicates, Normalizing and standardizing data. **Sample Programs:** • Write a Python program to load a dataset and identify issues such as missing values, incorrect data types, and inconsistent formatting. • Write a Python program to normalize and standardize numerical data in a dataset.	**CO-2**
3	**Matplotlib:** Working with axes, working with legends, line plot, scatter plot, bar plot, box plot, histogram, pie chart and subplots. **Seaborn:** Explore seaborn, customizing seaborn plots, colour palette, multiple plots. **Sample Programs:** • Write a Python program to create a line plot with labeled axes, title, and legend. • Write a Python program to create multiple subplots in a single figure using Matplotlib. • Write a Python program to apply different Seaborn color palettes to a boxplot.	**CO-3**
4	**Data visualization:** Crosstab, scatter plot, box plot, tree map/heat map, bump chart, histograms, motion charts, waterfall charts, waffle charts, geospatial data using choropleth maps. **Sample Programs:** • Write a Python program to create a crosstab of categorical data and visualize the relationship between two numerical variables using a scatter plot. • Write a Python program to create a choropleth map using geopandas and folium to visualize data based on geographic regions.	**CO-4**

TEXT BOOKS:

1. Introduction to Data mining by Vipin Kumar, Pang-Ning Tan, Michael Steinback, Anuj Karpatne;2018
2. Ashutosh Nandeshwar, Tableau Data Visualization Cookbook, 1e, Packt Publishing.
3. Introduction to statistics by PkGiri and Banerjee, Academic publishers
4. Python for Data Analysis by Wes McKinney, 2nd Edition, O'REILLY

REFERENCE BOOKS:

1. Fabio Nelli, Python Data Analytics, First edition.
2. Data Wrangling with Python: Tips and Tools to Make Your Life Easier, Jacqueline Kazil and Katharine Jarmul, O'Reilly
3. Interactive Data Visualization: Foundations, Techniques, and Applications, Ward, Grinstein Keim, Natick A. K. Peters Ltd.