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)

2022-23

Academic Regulations (R20-CSE) Curriculum & Syllabi

(III Year I Semester)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

VISION

Our vision is to emerge as a world class Computer Science and Engineering department through excellent teaching and strong research environment that responds swiftly to the challenges of changing computer science technology and addresses technological needs of thestakeholders.

MISSION

To enable our students to master the fundamental principles of computing and to develop in them the skills needed to solve practical problems using contemporary computer-based technologies and practices to cultivate a community of professionals who will serve the public as resources on state-of-the-art computing science and information technology.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO-1	Employability	Work as Competent Computer Engineer either globally or locally by engaging in professional practice in a variety of roles with ability to serve as a team or individual.
PEO-2	Higher studies	Prepared to pursue masters or research programmes in computer science or otherdisciplines.
PEO-3	Entrepreneurship	Become successful Entrepreneurs who demonstrate strong technical and leadership skills to bring out innovative designs/products that alsoaddresses social issues.
PEO-4	Lifelong learning andethics	Adapt to rapidly changing technology in engineering domains through continuous learningand practice code of ethics.

PROGRAM SPECIFIC OUTCOMES (PSOs)

1	Programming and software Development skills: Ability to acquire programming efficiency to analyze, design and develop optimal solutions, apply standard practices in software project development to deliver quality software product.
2	Computer Science Specific Skills: Ability to formulate, simulate and use knowledge in various domains like data engineering, image processing and information and network security, artificial intelligence etc., and provide solutions to new ideas and innovations

PROGRAM OUTCOMES (POs)

Graduate Attribute1:	Engineering Knowledge
PO-1	Apply the knowledge of basic engineering sciences, humanities, core engineering and computing concept in modeling and designing computer based systems.
Graduate Attribute2:	Problem Analysis
PO-2	Identify, analyze the problems in different domains and define the requirements appropriate to the solution.
Graduate Attribute3:	Design/Development of Solution
PO-3	Design, implement & test a computer-based system, component or process that meet functional constraints such as public health and safety, cultural, societal and environmental considerations.
Graduate Attribute4:	Conduct Investigations of Complex Problems
PO-4	Apply computing knowledge to conduct experiments and solve complex problems, to analyze and interpret the results obtained within specified timeframe and financial constraints consistently.
Graduate Attribute5:	Modern Tool Usage
PO-5	Apply or create modern techniques and tools to solve engineering problems that demonstrate cognition of limitations involved in design choices.
Graduate Attribute6:	The Engineer and Society
PO-6	Apply contextual reason and assess the local and global impact of professional engineering practices on individuals, organizations and society.
Graduate Attribute7:	Environment and Sustainability
PO-7	Assess the impact of engineering practices on societal and environmental sustainability.
Graduate Attribute8:	Ethics
PO-8	Apply professional ethical practices and transform into good responsible citizens with social concern.
Graduate Attribute9:	Individual and Team Work
PO-9	Acquire capacity to understand and solve problems pertainingto various fields of engineering and be able to function

	effectively as an individual and as a member or leader in ateam.
Graduate Attribute10:	Communication
PO-10	Communicate effectively with range of audiences in both oral and written forms through technical papers, seminars, presentations, assignments, project reports etc.
Graduate Attribute11:	Project Management and Finance
PO-11	Apply the knowledge of engineering, management and financial principles to develop and critically assess projectsand their outcomes in multidisciplinary areas.
Graduate Attribute12:	Life-long Learning
PO-12	Recognize the need and prepare oneself for lifelong selflearning to be abreast with rapidly changing technology.

Third Year First Semester Course Structure

CODE	SUBJECT NAME	Category	Scheme of instruction			Sessional marks	Semester end Exam marks	Total Marks	Credits
			L	T	P				
CSE 311	OPEN ELECTIVE-I* (Essentials of Python as EC)	OE	3	0	0	40	60	100	3
CSE 312	PROFESSIONAL ELECTIVE -I	PE	3	0	0	40	60	100	3
CSE 313	COMPETITIVE PROGRAMMING	SOC	2	1	0	40	60	100	3
CSE 314	COMPILER DESIGN	PC	2	1	0	40	60	100	3
CSE 315	DATA BASE MANAGEMENT SYSTEMS	PC	3	0	0	40	60	100	3
CSE 316	DESIGN & ANALYSIS OF ALGORITHMS	**PC**	2	1	0	40	60	100	3
CSE 317	DATA BASE MANAGEMENT SYSTEMS LAB	**PC**	0	0	3	50	50	100	1.5
CSE 318	COMPETITIVE PROGRAMMING LAB	**SOC**	0	0	3	50	50	100	1.5
CSE 319	QA-I &SOFT SKILLS	**HS**	0	0	3	100	0	100	1.5
CSE31A	SUMMER INTERNSHIP-INDUSTRY-1	**PR**	0	0	0	100	0	100	2
Total			**15**	**3**	**9**	**540**	**460**	**1000**	**24.5**

PROFESSIONAL ELECTIVES		OPEN ELECTIVES	
PE1	1. CSE 312(A)Smart Systems Design & Programming 2. CSE312(B)Advanced Data Structures 3. CSE312(C)No SQL Data Bases 4. CSE 312(D)Artificial Intelligence	OE 1	CSE 311 Essentials of Python (as an emerging course)

L - Lecture (clock hours) T - Tutorial (clock hours) P - Practical (clock hours)
OE – Open Elective PE - Professional Elective SOC - Skill Oriented Course
PC – Professional course HS – Humanity Sciences PR - Project

ESSENTIALS OF PYTHON (Open Elective)	
Code: CSE 311	**Credits: 3**
Instruction: 3 L	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Pre-requisites: Basic Knowledge of Programming Fundamentals

Course Objectives:

The course should enable the students to:

- Describe the core syntax and semantics of Python programming language.
- Illustrate the process of structuring the data using lists, dictionaries, tuples, strings and sets.
- Discover the need for working with the functions, modules and packages.
- Infer the Object-oriented Programming concepts in Python.
- Familiarize the advanced concepts like regular expressions, Indicate the use of and built-in functions to navigate the file system.

Course Outcomes (CO):

By the end of the course, the student will be able to:	
CO-1	Interpret the fundamental Python syntax and semantics and able to solve, test and debugpython programs
CO-2	Fluency in the use of Python control flow statements and determine the methods tocreate and manipulate Python programs by utilizing the data structures like lists,
CO-3	Express proficiency in the handling of functions, modules and packages.
CO-4	Articulate the Object-Oriented Programming concepts such as encapsulation, inheritanceand polymorphism as used in Python.
CO-5	List the usage and application of regular expressions. Identify the commonly used operations involving file systems.

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	2	3	-	-	-	-	-	-	-	-	-	1
2	2	2	2	2	-	-	-	-	-	-	-	-	-	1
3	3	2	1	2	-	-	-	-	-	-	-	-	-	-
4	4	2	3	3	3	-	-	-	-	-	-	-	-	1
5	5	3	2	3	-	-	-	-	-	-	-	-	-	1

SYLLABUS

UNIT-I: **8 Periods**

Introduction: Installation, Features, Applications, Keywords and Identifiers, Statement, Indentation, Comments, Variables, Constants, Literals, Data Types, Type Conversion, I/O, Operators, Namespace and Scope.

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

1. Analyze fundamental advantages of python over the other programming languages.
2. Solve, test and debug basic problems using python script.

UNIT-II: **12 Periods**

Flow control & Collections: If, If...else, if...elif...else, Nested if, for loop, while loop, Break, Continue and Pass. Numbers, Decimal, Fractions, Mathematics, List, Tuple, String, Set and Dictionary. Data types manipulations (create, Index, Negative indexing, Slicing, change or add elements, delete or remove elements, Methods, Comprehension, Membership Test, Iteration, Operations and Built in Functions)

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

1. Implement Flow control statements required real world problems.
2. Manipulate python programs by using the python data structures like lists, dictionaries, tuples, strings and sets.

UNIT-III: **12 Periods**

Functions: Function, Function argument, Recursion, Anonymous / Lambda functions, Global, Local and Nonlocal variables, Global keyword, Decorators, Modules and Packages. Exception Handling in Python, what is an Exception? Syntax for Exception Handing, Handling Single Exception, Handling Multiple Exceptions.

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

1. Resolve real world problems using python functions.
2. Familiarize the usage of Modules and packages to enhance the problem solving and usage of Exceptions.

UNIT-IV: **10 periods**

Object oriented programming: Introduction to OOPs, Class, Object, Constructors, Methods, Inheritance, Method Overriding, Multiple Inheritance, Operator overloading, Encapsulation and Polymorphism.

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

1. Design object-oriented programs with Python classes.
2. Usage of inheritance, encapsulation, inheritance, and polymorphism for reusability.

UNIT-V: **10 Periods**

Advanced topics: The RegEx package, Pattern using RegEx (Metacharacters, Special sequences, Sets), RegEx methods: (findall, finditer, match, search, split, sub, subn), Match object: (group, start, end, span, match.re & match. String and rawstring RegEX). Files (Open, Read, Write, Close) and File Methods, Python Directory: (Current, Changing, directory, list,New, Renaming, Removing).

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

1. Interpret the advantages of advanced concepts like regular expressions.
2. Identify the commonly used operation involved in files for I/O processing.

Textbooks:

1. Python Programming: Using problem solving approach: by Reema Thareja, Oxford University press.
2. Python Programming : A Modern Approach by Vamsi Kurama, Pearson

Reference books:

1. How To Think Like A Computer Scientist, Learning With Python, by Allen Downey, Jeffrey Elnker and Chris Meyers
2. A Beginners Guide to Python 3 Programming by John Hunt, Springer

SMART SYSTEM DESIGN & PROGRAMMING	
CSE 312(A)	**CREDITS: 3**
Instruction: 3 L	**Sessional Marks: 40**
End Exam: 3 Hours	**End Exam Marks: 60**

Pre-requisites:

- Basic knowledge of Microprocessor & Interfacing, Computer Organization, Digital logiccircuits
- Students must have knowledge of the C programming language.

Course Objectives:

The course should enable the students to:

- To learn the design and programming of microcontrollers.
- To learn the basics of ARM processors.
- To learn to program using ARM assembly language.
- To familiarize the students with Arduino kit and Raspberry Pi to implement small scaleembedded system applications.

Course Outcomes (CO):

By the end of the course, student will be able to	
CO-1	Describe the Embedded system fundamentals, design, and memory management.
CO-2	Write programs in ARM based assembly level language.
CO-3	Design Embedded system applications.
CO-4	Test and debug embedded system applications.
CO-5	Develop applications on Arduino and Raspberry Pi kits.

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	1	1	1	-	-	-	1	2	-	1	-	-
2	3	2	3	2	2	-	-	-	1	2	-	1	3	-
3	3	3	3	3	2	1	1	-	3	2	2	-	3	-
4	2	3	3	3	2	2	2	-	3	2	-	-	3	-
5	3	2	3	2	2	1	2	-	2	2	2	-	2	-

SYLLABUS

UNIT-I **10 Periods**

Introduction to Embedded Systems: Application domain of embedded systems, Desirable features and general features, Figures of merit, classification of MCUs.
Hardware Point of View: Microcontroller Unit, Memory for embedded systems.
Examples: Mobile phone, Automotive electronics, RFID, WISENET, Robotics, Biomedical applications, Brain machine interface.

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

1. Know about the Embedded systems and their classifications.
2. Identify the different micro controller units used in verity of hardware units.

UNIT-II **10 Periods**

Hardware Software Co-design and Embedded Product Development Lifecycle Management: Hardware Software Co-design, Modeling of systems, Embedded product development lifecycle management, Lifecycle models.
Embedded Design: A Systems Perspective – A typical example, Product design, The design process, Testing, Bulk manufacturing.

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

1. Extract information about the embedded product development lifecycle through different models.
2. Describe the design process of embedded products and manufacturing.

UNIT –III **12 Periods**

ARM Architecture and Assembly Language Programming – History, Architecture, Interrupt vector table, Programming, ARM Assembly language, ARM instruction set, Conditional execution, Arithmetic, logical & compare instructions, Multiplication, Division, Starting ALP, General structure of an Assembly Language Line, Writing ALP, Branch instructions, Loading Constants, Load and Store instructions.

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

1. Develop the simple ARM assembly language programs using instruction set.
2. Analyze the programs developed by using branching and comparing.

UNIT-IV **12 Periods**

Introduction to Arduino: What Is Physical Computing? The Arduino Way, The Arduino Platform, Really Getting Started with Arduino. Advanced Input and Output. Troubleshooting.
Case study: Automatic Garden-Irrigation System.

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

1. Describes how to compute with Arduino platform and troubleshooting.
2. Demonstrate the Adriano systems with some real time applications.

UNIT-V

11 Periods

Introducing the Raspberry Pi: The History of Raspberry Pi, Exploring the Pi Board, Hardware Requirements of the Pi, The Pi Operating System, Connecting the Peripherals, Configuring the Pi, Getting Started with Python, Accessing the GPIO Pins, Using the GPIO Library in python, Connecting the Temperature/Humidity Sensor, Setting Up the Motion Sensor.

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

1. Recognize and explores Raspberry Pi boards and its functionality when connecting with peripherals.
2. Summarize accessing the GPIO pins and it library in python and analyze different sensors.

Textbooks:

1. Embedded Systems: An Integrated Approach, by Das, Lyla B, Pearson Education.
2. Learn Raspberry Pi Programming with Python, by Donat, Wolfram, Apress,

Reference books:

1. Assembly Language: Fundamentals and Techniques, by Hohl, William, and Christopher Hinds. *ARM* Crc Press.
2. Monk, Simon, Raspberry Pi cookbook: Software and hardware problems and solutions, O'Reilly Media, Inc.,

ADVANCED DATA STRUCTURES	
CSE 312(B)	**Credits: 3**
Instruction: 3 L	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:

- Understand the variety of advanced data structures (skip lists, hash tables, priority queues, balanced search trees, graphs).
- Give the advantages and dis-advantages of each of the advanced data structure.
- Learn how to apply algorithm design techniques and data structures to solve problems.
- Learn different external sorting techniques and analyze their efficiency.

Course Outcomes (CO):

By the end of the course, the student will be able to:	
CO-1	Implement the ADTs like linear list, skip list and hash tables and their operations
CO-2	Design and apply binary heaps, leftist heaps, Binomial queues and sorting for solving the real-world scenarios
CO-3	Describe the methods to balance a binary search tree using Rotation methods, Color changing methods, splitting, and merging of nodes.
CO-4	Apply Algorithms for solving problems by using string matching techniques
CO-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	-	-	2	-
2	2	3	3	-	-	-	-	-	-	1	-	-	2	-
3	2	3	2	-	-	-	-	-	-	1	-	-	2	-
4	2	3	2	-	-	-	-	-	-	1	-	-	2	-
5	2	3	2	-	-	-	-	-	-	1	-	-	2	-

SYLLABUS

UNIT-I: **12 Periods**

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

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

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

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

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, TheBoyer-Moore Algorithm.

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

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- unweighted shortest path, graphs with negative edge cost (Bellman–Ford), acyclic graphs, Network flow problems, Applications of BFS, DFS,

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

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++, by Sartaj Sahni,University Press
2. Data Structures and Algorithms in C++, by Adam Drozdek, Cengage Learning

Reference books:

1. Data Structures: A Pseudocode Approachwith C, by Richard F. Gilberg, Behrouz A. Forouzan, Cengage Learning.
2. Data Structures and Algorithm Analysis in C++, by Mark Allen Weiss, Pearson Education.

NoSQL DATABASES	
CSE 312(C)	**Credits:3**
Instruction: 3 L	Sessional Marks: 40
End Exam: 3 Hours	Ena Exam Marks: 60

Pre-requisites:

Knowledge on Relational Database management systems.

Course Objectives:

- Distinguish and describing how NoSQL databases differ from relational databases fromtheoretical perspective.
- Explore the origins of NoSQL databases and the characteristics.
- Demonstrate competency in selecting a particular NoSQL database for specific use cases.
- Demonstrate Document databases with MongoDB.

Course Outcomes (CO):

By the end of the course, the student will be able to:	
CO-1	Compare and contrast the uses of relational RDBMSs and NoSQL systems for different types of data and applications.
CO-2	Differentiate various data models.
CO-3	Recognize Key value Databases and document databases.
CO-4	Create a sample database using NoSql.
CO-5	Apply the Query concepts in MongoDB database

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	1	-	2	-	-	-	-	-	-	-	1	2
2	1	2	1	-	2	-	-	-	-	-	-	-	1	2
3	1	2	1	2	2	-	-	-	-	-	-	-	1	2
4	2	2	1	2	2	-	-	-	-	-	-	2	2	2
5	2	2	2	2	3	-	-	-	-	-	-	2	2	2

SYLLABUS

UNIT-I: **12 Periods**

Why NoSQL: The value of relational databases, Impedance mismatch, Application and integrationdatabases, Attack of the cluster.

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

1. Recall Relational databases and security aspects in Realtime
2. Identify working with multiple databases.

UNIT-II: **12 Periods**

Aggregate Data Models: Aggregates - Example of Relations and Aggregates – Consequences of Aggregate Orientation, Key -Value and Document Data Models, Column-Family Stores

More Details on Data Models: Relationships, Graph Databases, Schema less Databases, Materialized Views - Modeling forData Access.

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

1. Explain internal operations on Database.
2. Analyze how to view data from database in different ways.

UNIT –III: **12 Periods**

Distribution Models: Single Server, Shading, Master-Slave Replication, Peer-to-Peer Replication, CombiningShading and Replication

Consistency: Update Consistency, Read Consistency, Relaxing Consistency, The CAP Theorem, RelaxingDurability.

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

1. Analyze how multiple clients can interact with database server.
2. Apply updating values dynamically in database.

UNIT-IV: **12 Periods**

Key-Value Databases:

What Is a Key, Value Store, Key-Value Store Features, Consistency, Transactions, Query Features, Structure of Data, Scaling, Suitable Use Cases, Storing Session Information, User Profiles, Preferences, Shopping Cart Data, When Not to Use, Relationships among Data, Multioperation Transactions, Query by Data, Operations by Sets.

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

1. Use the way how Operations used in Real-time applications.
2. Solve working with the database transactions.

UNIT-V: **12 Periods**

Document Databases: What Is a Document Database? Features, Consistency, Transactions, Availability, Query Features, Scaling, Suitable Use Cases, Event Logging, Content Management Systems, Blogging Platforms, Web Analytics or Real-Time Analytics, E-Commerce Applications, When Not to Use, Complex Transactions Spanning Different Operations, Queries against Varying Aggregate Structure.

Introduction to MongoDB: Introduction to MongoDB, The Data Model, Working with Data, GridFS.

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

1. Integrate database server like MongoDB to cloud apps.
2. Test Cloud data storage.

Textbooks:

1. NoSQL Distilled, A Brief Guide to the EmergingWorld of Polyglot Persistence, Pramod J.Sadalag and Martin Fowler, Addison Wesley.
2. The definitive guide toMongoDB", "A complete guide to dealing with big data using MongoDB, by David Hows, Eelco Plugge, Peter Membrey , and Tim Hawkins, Apress.

Reference books:

1. Professional NoSQL, Wiley, by Shashank Tiwari, Wrox Press.
2. Getting Started with NoSQL, by Gaurav Vaish, Packt Publishing.

ARTIFICIAL INTELLIGENCE **(Professional Elective)**	
Code: CSE 312(D)	**Credits: 3**
Instruction: 3 L	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Pre-requisites: Probability and Discrete Mathematics, Data Structures and Algorithms, Formallanguages, and Automata Theory.

Course Objectives:

The course should enable the students to:

- To discuss about the Basic principles, techniques, and applications of artificial intelligence.
- To analyze and apply the insights into knowledge representation, problem-solving, andlearning methods in Science and Engineering domains.

Course Outcomes (CO):

By the end of the course, the student will be able to:	
CO-1	Illustrate foundations of Artificial Intelligence (AI) and its applications, Problem Types, Characteristics and Search Space Representations.
CO-2	Apply Search Techniques (Brute-Force, Heuristic and Game Playing) of Artificial Intelligence and solving AI problems by applying suitable searching methods.
CO-3	Discuss Knowledge Representation, Approaches and Types and apply to process through different reasoning approaches such as Propositional and Predicate Calculus.
CO-4	Explain different types of Learning in Artificial Intelligence and apply planning in real-time world examples.
CO-5	Discuss about Uncertainty and its importance and classify the various approaches of the Expert systems using case studies.

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	2	1	2	-	2	2	2	1	1	-	2	1	2
2	3	3	2	3	2	2	2	2	2	2	1	2	2	2
3	2	2	1	2	1	1	2	2	2	2	-	2	1	2
4	3	3	2	3	1	2	2	2	3	2	2	2	2	2
5	2	2	2	2	1	2	2	2	2	1	1	2	1	2

SYLLABUS

UNIT I: **10 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. Discuss foundations of AI and its agents, environment, and rationality and also aboutproblem solving agents.
2. Explain different types of AI problems as Search space representation, characteristics andtechniques.

UNIT II: **10 Periods**

Searching: Searching- Searching for solutions, uniformed search strategies – Breadth first search, depth firstSearch, 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). Game Playing - Adversarial search, Games, mini-max algorithm, optimal decisions in multiplayer games, Problem in Game playing, Alpha-Beta pruning, Evaluation functions.

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

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

UNIT III: **10 Periods**

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

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

1. Discuss representation of Knowledge and reasoning, issues, approaches and applications.
2. Apply Propositional and Predicate logic through Knowledge Representation and casestudies.

UNIT 4: **10 Periods**

Learning: Introduction, Types of Learning, Supervised Learning, Unsupervised Learning, Reinforcement Learning, Applications of Learning, Case-Based Reasoning
Planning: Overview, An Example Domain: The Blocks World, Components of a PlanningSystem, Goal Stack Planning, Hierarchical Planning.

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

1. Discuss different Learning approaches of AI to solve real world problems.
2. Discuss and Apply Planning on real-world problems such as Block-World, Wearing ashoe, etc.

UNIT 5: **10 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 elementin expert systems how expert systems works, problem areas addressed by expert systems, expert systems success factors, types of expert systems and Applications.

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

1. Discuss about Uncertainty using Probabilistic and Statistical reasoning approaches
2. Illustrate foundations of Expert Systems and how to apply in real-time Scenario.

Textbooks:

1. "Artificial Intelligence", by Elaine Rich, Kevin Knight, Shivashankar B. Nair, McGraw Hill.
2. "Artificial Intelligence, Structures, Strategies for Complex Problem Solving", by George F Luger, Addison Wesley.

Reference books:

1. "Artificial Intelligence: Foundations ofComputational Agent", by David L Poole, Alan K. Mackworth, Cambridge University Press.
2. "Artificial Intelligence: A Modern Approach, Prentice Hall series of Artificial Intelligence.

COMPETITIVE PROGRAMMING	
Code: CSE 313	**Credits:3**
Instruction: 2 Periods & 1 Tutorial/Week	**Sessional Marks: 40**
End Exam: 3 Hours	**End Exam Marks: 60**

Pre-requisites:

Problem solving techniques and Data Structures

Course Objectives:

The course should enable the students to:

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

Course Outcomes (CO):

By the end of the course Students will be able to	
CO-1	Identify type of the problem and Apply sorting, searching techniques to solve problems.
CO-2	Perform bit manipulation and string manipulation operations.
CO-3	Compute the given complex problem using mathematical theorem and graph algorithms.
CO-4	Solve given problems using dynamic programming and Greedy Approach.
CO-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	-

SYLLABUS

UNIT I **10 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, Heap Sort, Bucket Sort, Ternary Search.

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

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

UNIT II **10 Periods**

Bit Manipulation: Bit representation, bit operations, representing sets
Strings: String terminology, Trie structures, String hashing, Z-algorithm

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

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

UNIT III **10 Periods**

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

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

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

UNIT IV **10 Periods**

Dynamic programming: Coin problem, longest increasing subsequence, Paths in a grid,Edit distance, Counting Tilings
Greedy Algorithms: Coin problem, Tasks and deadlines, Minimizing sums, Data Compression

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

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

UNIT V **10 Periods**

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

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

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

Textbooks:

1. Competitve 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.

COMPILER DESIGN	
Code: CSE 314	**Credits: 3**
Instruction: 2 Periods & 1 Tutorial /week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Pre-requisites:

Basic structure of programming language and formal languages.

Course Objectives:

The course should enable the students to:

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

Course Outcomes (CO):

By the end of the course, the student will be able to:	
CO-1	Analyze the various phases and Tools of a Lexical Analyzer
CO-2	Apply Top-down parsing methods for various grammars
CO-3	Identify the differences in the functioning of various bottom-up parsers
CO-4	Differentiate different intermediate code generation techniques and apply a variety of optimization techniques to improve the code.
CO-5	Compare different code generation techniques, and how symbol table, run time storage and error detecting are managed.

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	-	-

SYLLABUS

UNIT-I: **12 Periods**

Introduction: Programs related to compilers. Translation process. Major data structures. Other issues in compiler structure. Boot strapping and porting.
Lexical analysis: The role of Lexical Analyzer, Input Buffering, Specification of Tokens, Recognition of Tokens, The Lexical-Analyzer Generator Lex.

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

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

UNIT-II: **12 Periods**

Design of Parsers: Top-down Parsers, Backtracking, Recursive Descent Parser, Left recursion,
Left factoring Ambiguity, LL (1) grammar, non-recursive descent parser (Predictive Parser).

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

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 Parsingmethods 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, CLRparsers. LALR parser, parser Generator - YACC.

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

1. Identify different Bottom-up Parsers and able to implement Syntax analysis phase byusing bottom-up parsers.
2. Explain the working procedure of syntax analysis phase with help of Bottom up Parsingmethods using Different Bottom up Parsers

UNIT-IV: **12 Periods**

Syntax Directed Translation: Syntax Directed Definitions and Translation. Intermediate code
generation, Three-Address Code, DAG, Types and Declarations, Translation of Expressions, Type Checking, Control Flow.
Machine Independent Optimizations: The Principal Sources of Optimizations, Introduction to data flow analysis, Foundation of data flow analysis.

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

1. Explain different Semantic analysis techniques.
2. Explain how intermediate code can be represented and different machine independent code optimization techniques.

UNIT-V: **12 Periods**

Code Generation: Problems, Machine model, A simple code generator, Machine dependent
code Optimization**,** Register allocation and assignment, Code generation from DAG,
Peephole optimization.
Storage Organization and Error Recovery: Symbol tables, Run-time storage administration, Stack,
Heap Management, Garbage Collection, Error detecting and Reporting in various Phases.

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

1. Explain how code generation and machine dependent techniques
2. Explain the working of symbol table, detection, and recovery in Compilation process.

Text Book:

1. Principles of Compiler Design, by Aho, D. Ullman, Pearson Education
2. Advanced Compiler Design and Implementation, by Steven S. Muchnick Morgan Kaufmann Publishers Inc.

Reference Books:

1. Compiler Construction: Principles and Practice, Cengage Learning. Lex &Yacc, by Kenneth C Louden, John R Levine, Oreilly Publishers.
2. Engineering a Compiler, by Keith D Cooper & Linda Tarezon, Morgan Kafman, Second edition. Lex &Yacc, John R Levine, Tony Mason, Doug Brown, Shroff Publishers.

DATA BASE MANAGEMENT SYSTEMS	
CSE 315	**Credits: 3**
Instruction: 3 Periods/week	Sessional Marks: 40
End Exam: 3 Hours	End Exam Marks: 60

Pre-requisites:
Basic Knowledge about Set Theory, Discrete Mathematics, Relations and Functions, Data Structures, and algorithms

Course Objectives:
The course should enable the students to:

- Understand basic database concepts, including the structure and operation of the relational data model.
- Understand logical database design principles, including E-R diagrams and database normalization
- To learn the basics of SQL and construct queries using SQL
- Understand the concept of database transaction and concurrency control, backup and recovery, data object locking and protocols.

Course Outcomes (CO):

By the end of the course, the student will be able to:	
CO-1	Describe basic concepts of database systems and principles of transaction processing, concurrency techniques and recovery of database.
CO-2	Apply Conceptual and logical database design principles, including E-R diagrams
CO-3	Compose SQL queries to perform operations on database. (Create, Retrieve, Update, Delete)
CO-4	Construct relational algebra expressions for queries
CO-5	Analyze and apply schema Refinement, database normalization principles.

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	-

SYLLABUS

UNIT-I: **12 Periods**

Introduction to DBMS: Overview of DBMS, File system versus a DBMS, Advantages of a DBMS, Three Schema architecture of DBMS, Data Models, Database Languages, Transaction Management, Structure of a DBMS, Client/Server Architecture, Database Administrator and Users.

Entity-Relationship Model: Design Issues, ER Modeling concepts, Cardinality constraints, Weak-entity types, Subclasses and inheritance, Specialization and Generalization, Conceptual Database Design with the ER Model.

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

1. To design and build a simple database system and demonstrate competence with the fundamental tasks involved with modeling, designing, and implementing a DBMS.
2. Describe the fundamental elements of relational database management systems and design ER-models to represent simple database application scenarios

UNIT-II: **10 Periods**

Relational Model: Structure of Relational Databases, Basics of Relational Model, Integrity Constraints, Logical Database Design, Introduction to Views, Destroying/ Altering Tables and Views, Relational Algebra, Relational Calculus.

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

1. Design entity relationship and convert entity relationship diagrams into RDBMS and formulate SQL queries on the respect data into RDBMS and formulate SQL queries on the data.
2. Recall Relational Algebra concepts, and use it to translate queries to Relational Algebra statements and vice versa.

UNIT-III: **12 Periods**

SQL: Concept of DDL, DML, DCL, set operations, Nested queries, Aggregate Functions, NullValues, Referential Integrity Constraints, assertions, views, Embedded SQL, Cursors Stored procedures and triggers, ODBC and JDBC, Triggers and Active Database, designing active databases.

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

1. Perform PL/SQL programming using concept of Cursor Management, Error Handling, Package and Triggers.
2. Use the basics of SQL and construct queries using SQL in database creation andinteraction

UNIT-IV: **12 Periods**

Database Design: Schema Refinement, Functional Dependencies, Reasoning about Functional
Dependencies, Normalization using functional dependencies, Decomposition, Boyce-Codd Normal Form, 3NF, Normalization using multi-valued dependencies, 4NF, 5NF

Security: Access Control, Discretionary Access Control - Grant and Revoke on Views and IntegrityConstraints, Mandatory Access Control.

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

1. Apply various Normalization techniques, functional Dependency and Functional Decomposition to improve the database design.
2. Implement typical security techniques in real time applications.

UNIT-V: **12 Periods**

Transaction Management: The ACID Properties, Transactions & Schedules, Concurrent Execution of Transactions, Lock- Based Concurrency Control.

Concurrency Control: 2PL, Serializability and Recoverability, Introduction to Lock Management, Lock Conversions, Dealing with Deadlocks, Specialized Locking Techniques, Concurrency Control without Locking.

Crash Recovery: Introduction to ARIES, The Log, Other Recovery-Related Structures, TheWrite-Ahead Log Protocol, Check pointing, recovering from a System Crash, Media Recovery.

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

1. Execute various SQL queries related to Transaction Processing & Locking using theconcept of Concurrency control.
2. Analyze the crash recovery techniques of database systems and apply transaction processing mechanisms in relational databases.

Textbooks:

1. Database Management Systems, by 4th Edition, McGraw- Hill 2003 by Raghu Ramakrishnan, Johannes Gehrke.
2. Database System Concepts, by A.Silberschatz.H.Korth , McGraw-Hill

References Books:

1. Introduction to Database Systems, by Bipin Desai, Galgotia Publications.
2. Fundamentals of Database System, by RamezElmasri, Shamkant B. Navathe. Addison-Wesley.

DESIGN AND ANALYSIS OF ALGORITHM	
CSE 316	**Credits: 3**
Instruction: 3 L	**Sessional Marks: 40**
End Exam: 3 Hours	**End Exam Marks: 60**

Pre-requisites:

Basic programming, problem solving and data structures

Course Objectives:

The course should enable the students to:

- Student will understand the basic design concepts (e.g., pseudo code, specifications, top-Down design).
- Student will learn the different algorithm design strategies (procedural knowledge).
- Student can acquire the knowledge to solve the complexities of different problems.
- Student will be able to choose appropriate design strategies for different problems.

Course Outcomes (CO):

By the end of the course, the student will be able to:	
CO-1	Discuss the algorithm design concepts (e.g., pseudo code, specifications, top-down design).
CO-2	Explore different algorithm design strategies.
CO-3	Apply the algorithms and design strategies to solve problems
CO-4	Analyze the complexities of various problems in different domains
CO-5	Illustrate the notions of P, NP, NP-hard problems, and its challenges.

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	-	-	-	-	-	-	2	-	-	-	1	-
2	2	3	-	-	-	-	-	-	2	-	-	-	1	-
3	2	3	2	2	-	-	-	-	2	-	-	-	1	-
4	2	3	2	2	-	-	-	-	2	-	-	-	1	-
5	1	2	-	2	-	-	-	-	-	-	-	-	-	-

SYLLABUS

UNIT- I **12 Periods**

Introduction:
Introduction, Steps for algorithmic problem solving, Important Problem Types, Asymptotic Notations and Efficiency Classes, Mathematical Analysis for recursive Algorithms and Non-recursive Algorithms, Empirical Analysis, Algorithm Visualization.

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

1. Discuss the correctness of algorithms using inductive proofs and invariants.
2. Analyze worst-case running times of algorithms using asymptotic analysis.

UNIT-II: **12 Periods**

Brute Force: Selection and Bubble sort, Sequential Search, Closest- Pair, Convex Hull Problem.
Exhaustive Search: Travelling Salesman problem, Knapsack problem, Assignment Problem.
Divide-and-Conquer: General Method, Binary Search, Merge sort, Quick sort, Stassen's MatrixMultiplication, Multiplication of large integers.

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

1. Describe the divide-and-conquer paradigm and explain when an algorithmic design situation calls for it.
2. Apply algorithms that employ this paradigm.

UNIT-III: **12 Periods**

Transform and conquer: Pre-sorting, Gauss Elimination, Balanced Trees – 2-3 Trees, Heap sort, Horner's rule and binary exponentiation, Problem reduction – Least Common Multiple, Counting paths in a graph.
Dynamic Programming: All Pair Shortest Path Problem, Optimal Binary Search Trees, 0/1 Knapsack Problem and Memory Functions.

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

1. Describe the dynamic-programming paradigm and explain when an algorithmic design situation calls for it.
2. Apply algorithms that employ this paradigm. Synthesize dynamic programming algorithms and analyze them.

UNIT-IV: **12 Periods**

Greedy Technique: Minimum Spanning Tree Algorithm, Single Source Shortest Path Problem, Huffman Trees.
Space and Time Trade-offs: Sorting by computing, Input Enhancement in String Matching-Horspool's Algorithm, Boyer-Moore Algorithm
Backtracking: N-queen problem, Sum of subsets problem, Graph Coloring, Hamiltonian Cycle Problem.

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

1. Discuss the greedy paradigm and explain when an algorithmic design situation calls forit.
2. Apply algorithms that employ this paradigm. Synthesize greedy algorithms and analyzethem.

UNIT-V: **12 Periods**

Branch and Bound: General method, applications - Travelling salesman problem, 0/1 knapsack problem- Assignment Problem.
NP-HARD & NP-COMPLETE PROBLEMS: Lower Bound Arguments, P, NP, NP - Hard and NP Complete classes, Challenges in numerical algorithms.

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

1. Explain what competitive analysis is and to which situations it applies.
2. Perform competitive analysis.

Textbooks:

1. " Introduction to Design & Analysis of Algorithms ", by Anany Levitin, Pearson Education.
2. " Fundamentals of Computer Algorithms", by Ellis Horowitz, S. Sahni et.al, Galgotia.

Reference Books:

1. " Introduction to Algorithms", by Thomas H. Corman, Charles E. Leiserson, Ronald R. Rivest & Clifford Stein, Prentice Hall of India.
2. " The Design and Analysis of computer Algorithms", by Aho, Hopcroft & Ullman, Pearson Education.

DATABASE MANAGEMENT SYSTEMS LAB	
CSE 317	**Credits: 1.5**
Instruction**:** 3 P	Sessional Marks: 50
End Exam: 3 Hours	End Exam Marks: 50

Pre-requisites:

Elementary knowledge about computers including some experience using UNIX or Windows. Knowledge about data structures and algorithms, corresponding to the basic course on Data Structures and Algorithms.

Course Objectives:

- To understand the basics of SQL and construct queries using SQL.
- To learn connectivity between web pages, OLAP, OLTP.

Course Outcomes(CO):

By the end of the course, the student will be able to:	
CO1	Make use of basic SQL queries to solve simple problems.
CO2	Solve complex queries using nested queries and joins.
CO3	Construct triggers, views, and stored procedures for different scenarios
CO4	Apply the principles of ER model and normalization for schema refinement in logical database design.

.

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	2	1	2	1	-	-	1	-	-	-	2	2
2	2	3	2	2	2	1	-	-	1	-	-	-	2	2
3	2	3	2	2	2	1	-	-	1	-	-	-	2	2
4	2	3	3	2	2	1	-	-	3	-	-	-	3	2

SYLLABUS

List of Experiments:

Experiment Name	Mapping
1. SQL DDL ,DML Statements	CO1
2. SQL Constraints.	CO1
3. Inbuilt functions in RDBMS.	CO2
4. Aggregate functions	CO2
5. Nested Queries & Join Queries.	CO2
6. Creation and dropping of Views.	CO3
7. Creating Triggers.	CO3
8. Stored Procedures.	CO3
9. Developing a sample application which includes all database design steps like requirements analysis, logical database design, normalization, developing user interface to access database from the application.	CO4

Sample Applications:

1. Development of an Online Course Portal for a campus
2. Book Bank Management System
3. Car Rental Management System
4. Exam/academic system for College Management
5. Real estate Management system
6. University Management System
7. Database manager for a Magazine agency or a newspaper agency
8. Ticket booking for performances
9. Inventory Control System
10. Students management System

REFERENCE BOOKS:

1. Raghu Ramakrishnan, Johannes *Gehrke "Database Management Systems*", 3rd Edition,2003, McGraw- Hill,
2. A.Silberschatz.H.Korth, *"Database System Concepts"* , 6th Edition, 2010,McGraw-Hill

COMPETITIVE PROGRAMMING LAB	
CSE 318	**Credits: 1.5**
Instruction:	Sessional Marks: 50
End Exam: 3 Hours	End Exam Marks: 50

Prerequisites: C and data Structures

Course objectives:

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

Course outcomes(CO):

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

1. Maximum sub array sum **CO1**

Solve the Maximum sub array sum with different time complexities.

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, sinceit divides the array into two parts: [2,7] and [11,16]. Further, let us consider the case of anarray 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.

I. First line of the input should contain two integer values which specify the number ofelements in array1 and array2.

II. Second line of the input should contain the elements of the first array.

III. 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 highor return mid. 1. If y*mid almost equal to x, we return mid. 2. If y*mid is less than x, weupdate low to mid 3. If y*mid is more than x, we update high to mid We need to care aboutdivision by zero and sign of the result etc. Input: one line of input should contain twonumbers 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

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 aninteger 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 aninput 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 oneline 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. 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 matchto 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

7. Rotten Oranges **CO2**

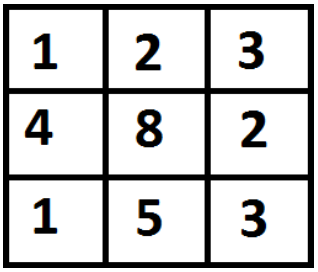
Given a grid of dimension **nxm** where each cell in the grid can have values 0, 1 or 2 which hasthe following meaning:
0 : Empty cell
1 : Cells have fresh oranges
2 : Cells have rotten oranges
We have to determine what is the minimum time required to rot all oranges. A rotten orange atindex [i,j] can 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.

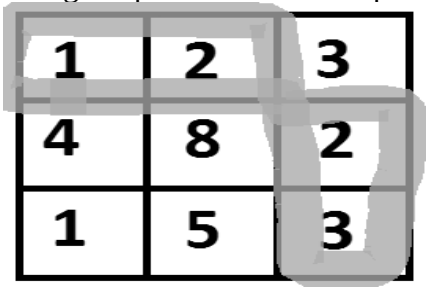
Example 1:

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.

8. 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.





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

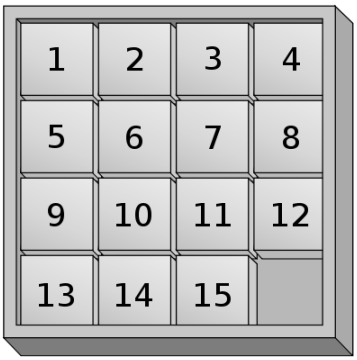
Sample Output 1
8

9. 15-Puzzle Problem **CO3**

The 15-puzzle is a very popular game: you have certainly seen it even if you don‟t know it bythat 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 sharesan 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 aninitial configuration of a 15-puzzle you must determine a sequence of steps that take you tothe 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.



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

10. 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

190 200

11. 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:

$0 <= N <= 10^8$

12. Euclid Problem **CO4**

From Euclid, it is known that for any positive integers A and B there exist such integersX 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, separatedwith 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

13. Coin Problem **CO4**

Given a value V. You have to make change for V cents, given that you have infinite supplyof 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 ofcoins.

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 numberof 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

Test case 2:

Input:
r = 3, c = 4
matrix[][] = {{1, 2, 3, 4},
{5, 6, 7, 8},
{9, 10, 11, 12}}
Output:
1 2 3 4 8 12 11 10 9 5 6 7

Text Books:

1. "Competitve 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 ProgrammingContest Training Manual, Springer.

QUANTITATIVE APTITUDE – I & SOFT SKILLS	
Code: CIV 319 - HS	**Credits: 1.5**
Instruction: 3 P	Sessional Marks :100

Prerequisites: -

Course objectives:

- To prepare the students on various principles related to numerical computations.
- To explain concepts related to numerical estimation.
- To illustrate and explain the fundamentals related to geometry and mensuration

Course Outcomes (CO):

By the end of the course, the student will be able to:	
1.	Solve problems related to numerical computations in company specific and other competitive tests.
2.	Able to recall and use the concepts to solve problems numerical estimation with respect to company specific and competitive tests.
3.	Apply basic principles related to geometry and mensuration & solve questions in company specific and competitive tests.

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	2	-	-	-	-	-	-	1	1	-	-	3	1
2	3	2	-	-	-	-	-	-	1	1	-	-	3	1
3	3	2	-	-	-	-	-	-	1	1	-	-	3	1

SYLLABUS

Section –A (Quantitative Aptitude –I)

UNIT I **6 Periods**

Numerical computation: Applications based on Numbers, Chain Rule, Ratio Proportion

UNIT II **6 Periods**

Numerical estimation – I: Applications Based on Time and work, Time, and Distance

UNIT III **4 Periods**

Numerical estimation – II: Applications based on Percentages, Profit Loss, and Discount, Simple interest and CompoundInterest, Partnerships, Shares, and dividends

UNIT IV **4 Periods**

Data interpretation: Data interpretation related to Averages, Mixtures, and allegations, Bar charts, Pie charts, Venndiagrams

UNIT V **4 Periods**

Application to industry in Geometry and Mensuration

Textbooks:

1. Quantitative Aptitude for Competitive Examinations - Quantitative Aptitude by rs agrawal (English, Aggarwal R. S.)-s chand Publications
2. A Modern Approach to Verbal & Non-Verbal Reasoning (R.S. Aggarwal) S Chand Publication

References books:

1. Quantitative Aptitude - For Competitive Examinations-U.Mohan Rao SCITECH publications
2. Quantitative Aptitude by Arun Sharma, McGrawhill publications