SYLLABUS FOR COMPUTER SCIENCE (MAJOR)

Under Single Major Single Minor (FYUGP)

(To be implemented from Session 2024-25)

SEM. V&VI

Proposed Syllabus for four years B.Sc. Computer Science (Major) Programme							
Year	**Semester**	**Paper Code**	**Paper**	**Credits**	**Periods/Week**	**Exam. Marks**	**Total Marks**
3rd Year	**V**	COMSMAJ509	Database Management System	3	3	60	80
		COMSMAJ3509L	Database Management System (Lab)	1	2	20	
		COMSMAJ510	Software Engineering	3	3	60	80
		COMSMAJ510T	Software Engineering (Tutorial)	1	1	20	
		COMSMAJ511	Data Communication and Computer Networks	3	3	60	80
		COMSMAJ511T	Data Communication and Computer Networks (Tutorial)	1	1	20	
		COMSMAJ512	Python Programming	3	3	60	80
		COMSMAJ512L	Python Programming (Lab)	1	2	20	
	VI	COMSMAJ613	Theory of Computation	3	3	60	80
		COMSMAJ613T	Theory of Computation (Tutorial)	1	1	20	
		COMSMAJ614	Design and Analysis of Algorithm	3	3	60	80
		COMSMAJ614T	Design and Analysis of Algorithm (Tutorial)	1	1	20	
		COMSMAJ615	Microprocessor	3	3	60	80
		COMSMAJ615L	Microprocessor (Lab)	1	2	20	
		COMSMAJ616	Computer Graphics	3	3	60	80
		COMSMAJ616L	Computer Graphics (Lab)	1	2	20	

NOTE:Tutorials should involve problem solving session/activity related to the subject taught.

3rd Year
Semester-V

Course-MAJOR Paper:	**Paper Code-COMSMAJ509 Database Management System**	**Credits-3**	**Lectures/Week-3**

Brief Course Description:

This course provides a comprehensive introduction to the principles, design, and implementation ofDatabaseManagementSystems(DBMS).Studentswilllearnthefundamentalconceptsofdatabaseorganization,datamodeling,andSQL(StructuredQueryLanguage).Thecourseemphasizesboththeoretical understanding and practical skills necessary for effective database design, implementation,andmanagement.

Prerequisite(s) and/or Note(s):

1. Basic understanding of:
    a. Computer systems
    b. Data structures and Databases
    c. Programming concepts
2. **Note(s):**The syllabus is subject to minor modifications depending on institutional or academic requirements.Students will be assessed only on the basis of topics covered in class during the semester.

Course Objectives

Knowledge Acquired:

Upon completion of the course, students will:

1. Understand the principles of relational database design.
2. Be able to implement and manage relational databases.
3. Gain practical knowledge of SQL for data querying and manipulation.
4. Learn about database optimization techniques.
5. Be prepared for roles in:
    - Database administration
    - Database development
    - Data-driven application development

Skills Gained:

1. Database design and schema creation
2. Proficiency in SQL for querying, updating, and managing data
3. Data modeling using Entity-Relationship (ER) diagrams
4. Query optimization for efficient data retrieval
5. Application of normalization techniques to database structures
6. Awareness of database security and access control measures

Competency Developed:

1. Database connectivity through APIs and tools
2. Implementing backup and recovery solutions
3. Performing database administration tasks
4. Understanding the basics of data warehousing
5. Introduction to data mining and knowledge discovery
6. Fundamentals of distributed databases and their management

Syllabus Overview

Unit 1: Introduction 10 Lectures

Definition of DBMS, file processing system Vs DBMS, Limitation of file processing system, databaseusers, Characteristics of database,database components,database system architecture- 3-layerand 2-layers, data independence, data dictionary. View of Data – Data Abstraction – Instances and Schemasdiagrams.DBA-DatabaseAdministrator-rolesandresponsibilitiesof aDBA

Unit 2: Database models 7 Lectures

A brief overview of three models- Hierarchical model, Network model and relationalmodel,comparisonofthree models.

Unit 3: DatabaseDesign,ER-Diagram 12 Lectures

ER-Model,Constraints,ER-Diagrams,different types of entities, attributes, relationships,E. F. Codd'srules,Keys-Primary,Candidate,Composite, Alternate, Foreign, Super, Relational Schemas, Normalization: Lossless decomposition, Lossydecomposition,Functionaldependencies,NormalForms- (Up toBCNF-1NF,2NF,3NF,BCNF).

Unit 4: TransactionProcessing 4 Lectures

ACIDproperties,Concurrencycontrol.

Unit 5: SQLandORACLE 12 Lectures

Oracle commands, Oracle datatypes, operators- Arithmetic, Logical, Range searching, Pattern Matching,Oracle functions- Aggregate, Numeric, String, Conversion, Date and Time, Oracle table-Dual, Groupingdata-Concept of grouping, Group by Clause, Having clause Language-DDL, DML, DCL, SQL Functions-queriesandsub-queriesinSQL, ConstrainsandindexesinSQL.

Suggested Readings

1. R.Elmasri,S.B.Navathe,FundamentalsofDatabaseSystems6thEdition, PearsonEducation, 2010.
2. R.Ramakrishanan, J.Gehrke, DatabaseManagementSystems3rdEdition, McGraw-Hill,2002.
3. A.Silberschatz,H.F. Korth,S.Sudarshan, DatabaseSystemConcepts6thEdition, McGrawHill,2010.
4. R.Elmasri,S.B.NavatheDatabaseSystemsModels,Languages,DesignandapplicationProgrammi ng,6thEdition,PearsonEducation,2013.
5. IvanBayross, Teach YourselfSQL &PL/SQL UsingORACLE8i& 9iwithSQLJ,1st Edition, BPBPublications,2003

Course-MAJOR Paper Code- COMSMAJ509L Credits-1 Lab hours/Week-2
Paper: Database Management System(Lab)

Studentsareadvicetodolaboratory/practicalpracticenot limitedto,butincludingthefollowingtypesofproblems.

1. Demonstratetablecreation
2. INSERTdataintothetable
3. SELECTtherecordfromthetable
4. SELECTallrowsandallcolumnsformthetable
5. Demonstratefilteringtabledata–Selectedcolumns andallrows,selectedrows andallcolumns,selectedcolumnsandselectedrows,Renametable.
6. SELECTallrecordsfromthetableeliminating duplicaterows.

7. Filteringtabledata
8. Sortingdatainatable
9. Creatingatablefromaanothertable
10. Insertingdataintoatablefromanothertable.
11. Demonstratedeleteoperations
12. Updatingthecontents ofatable
13. Modifyingthestructureof table
14. Destroyingtables
15. Findoutthetablecreatedbydifferentusers
16. Demonstratedifferenttypesof dataconstraints
17. DemonstratedifferentoperatorsinSQL-Arithmetic, logical,
18. Demonstratepatternmatching,rangesearching
19. Demonstrateoracletable'DUAL'
20. Demonstratedifferent oraclefunctions.

Course-MAJOR Paper:	**Paper Code-COMSMAJ510 Software Engineering**	**Credits-3**	**Lectures/Week-3**

Brief Course Description:

This course provides a comprehensive introduction to the principles, methodologies, and practices ofsoftware engineering. Students will learn the entire software development lifecycle, from requirementsgathering to design, implementation, testing, deployment, and maintenance.

Prerequisite(s) and/or Note(s):

1. Basic understanding of programming concepts
2. Basic familiarity with software development principles and terminology
3. **Note(s):**Course content may be revised during the term to align with academic priorities.Students will be evaluated only on the topics covered in class.

Course Objectives:

Knowledge Acquired:

Upon completion of the course, students will:

1. Understand and apply software engineering principles.
2. Effectively manage the Software Development Life Cycle (SDLC).
3. Contribute to the development of high-quality software systems.
4. Be prepared for roles in software development, project management, and related fields.
5. Gain knowledge in key software engineering areas, including:
    - SDLC phases
    - System design principles
    - Software testing techniques
    - Software configuration management

Skills Gained:

1. Application of SDLC models in real-world projects
2. Designing structured and modular software systems
3. Performing software testing and quality assurance
4. Managing software versions and configurations
5. Documenting and communicating software requirements and specifications

Competency Developed:

1. Strong programming skills relevant to software development
2. Enhanced problem-solving abilities

3. Application of algorithmic thinking to software design
4. Ability to participate in and contribute to software development projects
5. Working knowledge of software maintenance and evolution
6. Understanding the importance of team collaboration and project management in software engineering.

Syllabus Overview

Unit 1:	**Introduction**	**4 Lectures**

Definition of software and software engineering, the Evolving Role of Software, Software Characteristics,Softwarequality,ChangingNatureofSoftware,Softwareprocess.

Unit 2:	**Software Development Models**	**6 Lectures**

Software Development Life Cycle (SDLC), Types of Software Models- Waterfall Model, Iterative Model,SpiralModel,PrototypeModel.CapabilityMaturityModel Integration(CMMI).

Unit 3:	**SoftwareRequirementAnalysisandPlanning**	**10 Lectures**

Software Requirement Analysis and Modeling Techniques, Software Requirement Specification (SRS)-NeedforSRS,CharacteristicsandComponentsofSRS,Factfindingtechniques,CostEstimation-COCOMOModel,DataFlowDiagram(DFD)andEntity-RelationshipDiagram (ERD),

Unit 4:	**SoftwareProjectManagement-SPM**	**8 Lectures**

Define SPM, Needs of SPM, Software Risks, Risk Identification, RMMM Plan, Software Quality ControlandQualityAssurance,ISO(InternationalStandardsOrganization)9000Certification,Software Metrics.

Unit 5:	**Coding**	**4 Lectures**

Define Coding, Goals of Coding, characteristics of Programming Language, Programming style, CodingStandards,CodingGuidelines.

Unit 6:	**TestingStrategies&Tactics**	**13 Lectures**

Software Testing Fundamentals-Test Plan, Test Execution, Test Review, Inspection and Auditing, Testingprinciples, software Verification and validation, Types of software testing - Black-Box Testing, White-BoxTesting and their type, Other testing- Regression Testing, Mutation Testing, Acceptance testing, Alphatesting,Betatesting.

Suggested Readings

1. R.S.Pressman,SoftwareEngineering: APractitioner'sApproach(7thEdition),McGraw-Hill, 2009.
2. P.Jalote,AnIntegrated ApproachtoSoftwareEngineering(2ndEdition),NarosaPublishingHouse,2003.
3. K. K. AggarwalandY. Singh, SoftwareEngineering(2ndEdition),NewAgeInternationalPublishers,2008.
4. I.SommerVille, SoftwareEngineering(8thedition),AddisonWesley, 2006.
5. D.Bell, SoftwareEngineeringforStudents(4thEdition), Addison-Wesley, 2005.R.Mall,FundamentalsofSoftwareEngineering(2ndEdition),Prentice-HallofIndia,2004

Course-MAJOR Paper:	**Paper Code-COMSMAJ510T**	**Credits-1**	**Tutorial hours/Week-2**
	Software Engineering (Tutorial)		

Software Engineering tutorial as assigned and advised by teacher(s).

Course- MAJOR Paper Code-COMSMAJ511 Credits-2 Lectures/Week-2
Paper: Data Communication and Computer Networks

Prerequisite(s) and/or Note(s):

1. Basic Computer Literacy: Students should be familiar with fundamental computer operations
such as using the keyboard, mouse, files, folders, and basic software applications.
2. No Prior Networking Knowledge Required: This is an introductory course, designed for beginners.
3. Recommended for: Undergraduate students from Computer Science, IT, Electronics, or any discipline with interest in Information Technology.

Course Objectives:

1. To introduce students to the basic concepts and purpose of computer networks.
2. To explain the different types of networks, devices, transmission media, and communication protocols.
3. To provide a foundational understanding of network architectures such as the OSI and TCP/IP
models.
4. To familiarize students with basic IP addressing, configuration of small networks, and networking tools.
5. To develop practical skills for designing, setting up, and troubleshooting basic wired and wireless networks.

Knowledge Acquired:

By the end of this course, students will have gained theoretical and practical understanding of:
1. The role and components of computer networks in modern communication.
2. Types of networks (LAN, WAN, MAN), topologies, and data transmission media.
3. Hardware components like routers, switches, hubs, access points, NICs, and their uses.
4. Networking protocols such as TCP/IP, HTTP, FTP, SMTP, DNS, and DHCP.
5. Network security fundamentals including firewall basics, encryption, and safe internet practices.

Skills Gained:

1. Identify and differentiate between types of networks, hardware, and topologies.
2. Setup, configure, and troubleshoot a basic wired or wireless network.
3. Share resources like printers and files over a local network, Configure Wi-Fi routers and manage basic internet sharing settings.
4. Implement safe browsing and basic network security practices.

Competency Developed:

1. Understanding the foundational structure and operation of computer networks.
2. Performing hands-on tasks involved in configuring small network environments.
3. Communicating effectively using networking terminology and concepts.
4. Applying practical problem-solving techniques in network setup and troubleshooting.
5. Enhancing employability for roles such as IT support technician, helpdesk executive, or junior
network administrator.

Syllabus Overview

Unit 1: DataCommunicationFundamentalsandTechniques 10 Lectures

Introduction:DataCommunications,Networks,Categoriesofnetwork-LAN, MAN, WAN,ModesofCommunication-Simplex,HalfDuplex,FullDuplex,Networktopologies,InternetHistory,LayeredNetwork Architecture-OSI model andTCP/IP protocol suite. Data and Signals- Analog and Digital signals,ParallelandSerialtransmission,Transmissionimpairment-Attenuation,DistortionandNoise,Multiplexingtechniques- FDM,TDM, WDM,Transmission media - Guided: Twisted Pair, Coaxial Cable, Fiber Optic, Unguided -Wi-Fi, Bluetooth, Infrared, Satellite.

Unit 2: NetworksSwitchingTechniquesandAccessmechanisms 5 Lectures

Switching-Circuitswitching,Packetswitching;Messageswitching,ConnectionlessandConnection-orienteddatagramswitching.

Unit 3: DataLinkLayerFunctionsandProtocol 7 Lectures

Errordetectionanderrorcorrectiontechnique-HammingCode,CRC,Checksum.Flowcontrolmechanism-SlidingWindowprotocol, go-back-nARQ,stopandwaitARQ

Unit 4: MultipleAccessProtocolandNetworks 7 Lectures

Multiple Access Protocols: Radom Access - ALOHA, CSMA, CSMA/CD, CSMA/CA, Controlled Access -Reservation, Polling, Token Passing, Channelization - FDMA, TDMA, CDMA. Network devices- repeaters,hubs,switches, bridges,router andgateways;

Unit 5: NetworksLayer& Transport Layer FunctionsandProtocols 10 Lectures

Routing- Routingalgorithms,classesofIP, IPv4andIPv6 Addresses, IP Addressing schemes, subnetting (basic concepts);Transport services-Congestion control, Connection establishment and release- three- way handshaking,NetworkSecurity-Cryptography

Unit 6: Overview ofApplicationlayerprotocol 6 Lectures

OverviewofDNSprotocol,overviewofWWW, Important networking protocols: HTTP/HTTPS, FTP, SMTP, POP3, DNS and DHCP; Understanding firewalls and antivirus for network protection.

Suggested Readings

1. Behrouz A. Forouzan, “Data Communications and Networking”, 5th Edition, McGraw HillEducation,2013.
2. AndrewS.Tanenbaum,DavidJ.Wetherall,“ComputerNetworks”,5thEdition,PrenticeHall,2011.
3. Larry L. Peterson and Bruce S. Davie, “Computer Networks A System Approach”, 5th Edition,MKP,2012.
4. James F. Kurose, Keith W. Ross, “Computer Networking, A Top-Down Approach”, 5thEdition,Pearson,2012.
5. Dr.RakeshKumarMandal:Computer NetworksforStudents, FirstEdition,SPD,2018

Course-MAJOR Paper Code-COMSMAJ511T Credits-1 Lab hours/Week-2
Paper: Data Communication and Computer Networks (Tutorial)

Data Communication and Computer Networks tutorial as assigned and advised by teacher(s).

Course-MAJOR	Paper Code-COMSMAJ512	Credits-3	Lectures/Week-3
Paper:	Python Programming		

Prerequisite(s) and/or Note(s):

(1) High school mathematics.

(2) Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Course Objectives

Knowledge Acquired:

1. Fundamental Concepts: Students acquire knowledge of fundamental programming concepts such as variables, data types, loops, conditionals, and functions in Python.
2. Data Structures: They learn about essential data structures like lists, tuples, dictionaries, and sets, understanding their usage and implementation.

Skills Gained:

1. Coding Proficiency: Through hands-on practice and assignments, students develop coding proficiency in Python, enabling them to write clear, concise, and functional code.
2. Problem-Solving: They enhance their problem-solving skills by applying Python programming concepts to solve various computational problems and algorithms.
3. Debugging and Troubleshooting: Students acquire skills in debugging code and troubleshooting errors, learning how to identify and fix common programming mistakes effectively.

Competency Developed:

1. Logical Thinking: Python programming exercises require logical thinking and algorithmic problem-solving skills, helping students develop a logical mindset.
2. Attention to Detail: Writing code necessitates attention to detail to ensure accuracy and functionality. Students develop this competency through debugging and code review processes.
3. Collaboration and Documentation: Students learn to collaborate on coding projects using version control systems like Git and to document their code effectively, enhancing their ability to work in teams and communicate technical concepts clearly.

Syllabus Overview

Unit 1:	Introduction to Programming	6 Lectures

Concept of problem solving, Problem definition, Program design, Debugging, Types of errors in programming, Documentation. Flowcharts, algorithms, Types of programming paradigms – unstructured, procedure oriented, object oriented. Programming methodologies - top-down and bottom-up programming.

Unit 2:	Introduction to Python	12 Lectures

Structure of a Python Program, Elements of Python, Entering Expressions into the Interactive Shell, The Integer, Floating-Point, and String Data Types, String Concatenation and Replication, Storing Values in Variables**.**

Unit 3:	Flow control and Functions	12 Lectures

Boolean Values, Comparison Operators, Boolean Operators, Mixing Boolean and Comparison Operators, Elements of Flow Control, Program Execution, Flow Control Statements, Importing

Modules, Ending a Program Early with sys.exit(), def Statements with Parameters, Return Values and return Statements, The None Value, Keyword Arguments and print(), Local and Global Scope, The global Statement, Exception Handling.

Unit 4: List, Dictionary, String and Tuples 15 Lectures

String, String functions, Manipulating Strings, Lists: Creating Lists; Operations on Lists; Built-in Functions on Lists; Implementation of Stacks and Queues using Lists; Nested Lists. Dictionaries: Creating Dictionaries; Operations on Dictionaries; Built-in Functions on Dictionaries; Dictionary Methods; Populating and Traversing Dictionaries. Tuples and Sets: Creating Tuples; Operations on Tuples; Built-in Functions on Tuples; Tuple Methods; Creating Sets; Operations on Sets; Built-in Functions on Sets; Set Methods.

Suggested Readings:

1.Yashavant Kanetkar and Aditya Kanetkar , Let Us Python,BPB,3rd Edition.
2. Sheetal Taneja and Naveen Kumar, Python Programming- A modular approach with Graphics, Database,
3. Mobile and Web applications, Pearson, Sixteenth Impression-2023,
4. T. Budd, Exploring Python, TMH, 1st Ed, 2011
5. Python Tutorial/Documentation www.python.or 2015
6. Allen Downey, Jeffrey Elkner, Chris Meyers, How to think like a computer scientist: learning withPython , Freely available online.2012

Course-MAJOR Paper: **Paper Code- COMSMAJ512L Python Programming (Lab)** **Credits-1** **Lab hours/Week-2**

Studentsareadvisedto dolaboratory/practicalpractice notlimitedto,but includingthefollowing types of problems:

1. Write a menu driven program to convert the given temperature from Fahrenheit to Celsius and vice versa depending upon users' choice.
2. WAP to calculate total marks, percentage and grade of a student. Marks obtained in each of the three subjects are to be input by the user. Assign grades according to the following criteria:

Grade A: Percentage >=80
Grade B: Percentage>=70 and <80
Grade C: Percentage>=60 and <70
Grade D: Percentage>=40 and <60
Grade E: Percentage<40

3. Write a menu-driven program, using user-defined functions to find the area of rectangle, square, circle and triangle by accepting suitable input parameters from user.

4. WAP to display the first n terms of Fibonacci series.
5. WAP to find factorial of the given number.
6. WAP to implement the use of arrays in Python.
7. WAP to implement String Manipulation in python in Python.
8. WAP to find sum of the following series for n terms: 1 – 2/2! + 3/3! - - - - - n/n!
9. WAP to calculate the sum and product of two compatible matrices.
10. WAP to create Class and Objects in Python.
12. WAP to implement Data Hiding in Python.
13. WAP to implement constructor and destructor for a class in Python.

14. WAP to implement constructor and destructor in Python.
15. WAP to implement different types of inheritance in Python.
16. WAP to implement concept of Overriding in Python.
17. Write programs to create mathematical 3D objects using class.
    a. curve b. sphere c. cone d. arrow e. ring f. cylinder
18. WAP to Check if a Number is Positive, Negative or 0
19. WAP to Check leap year or not.
20. WAP to display calendar of the given month and year.

3rd Year
Semester-VI

Course-MAJOR	**Paper Code-COMSMAJ613**	**Credits-3**	**Lectures/Week-3**
Paper:	**Theory of Computation**		

BriefCourseDescription:

The Theory of Computation course explores into the foundational aspects of computer science, focusing on the study of formal languages, automata, and Turing machines. It explores the theoretical underpinnings of what can be computed and how efficiently computations can be performed. This course is crucial for understanding the limits of computation and the computational complexity of various problems.

Prerequisite(s)and/orNote(s):

Students interested in this course should have completed introductory courses in Discrete Mathematics and Algorithms. A solid grasp of mathematical logic and the ability to construct and understand mathematical proofs are essential for success in this course.

CourseObjectives:

The primary objectives of the Theory of Computation course are to impart a deep understanding of the basic concepts of computation, to familiarize students with various computational models, and to enable them to analyse the complexity of computational problems. By the end of the course, students will be well-versed in the theoretical aspects of computer science that are fundamental to advanced study and research in the field.

Knowledgeacquired:

1. Gain a comprehensive understanding, including finite automata, context-free grammars, and pushdown automata.
2. Learn about the hierarchy of formal languages and their corresponding automata models.
3. Study the theoretical framework of Turing machines and their role in computational theory.
4. Acquire knowledge about the boundaries of computation and the inherent limitations of computational processes.

Skillsgained:

Students will acquire skills to design and analyze various types of automata and grammars.
Develop problem-solving abilities related to computational complexity and algorithmic challenges.
Gain proficiency in constructing and proving theorems in the context of automata theory and formal languages.
These skills are essential for addressing complex problems in computer science and for conducting theoretical research.

CompetencyDeveloped:

1. Students will develop competence in identifying and solving intricate computational problems.
2. The subject will hone abstract thinking and formal reasoning abilities to apply theoretical concepts effectively.
3. Apply theoretical knowledge to real-world scenarios in fields such as algorithm design, software development, and computer science research.
4. This competency prepares students for advancing in diverse fields where complex problem-solving and theoretical understanding are crucial.

Syllabus Overview

Unit 1:	**Languages**	**8 Lectures**

Alphabets, string, language, Basic Operations on language, Chomsky hierarchy, Concatenation, Kleene Star

Unit 2:	**Finite Automata and Regular Languages**	**15 Lectures**

Regular Expressions, Transition Graphs, Deterministic and non-deterministic finite automata, NFA to DFA Conversion, Regular languages and their relationship with finite automata, Pumping lemma and closure properties of regular languages.

Unit 3:	**Context free languages**	**12 Lectures**

Context free grammars, parse trees, ambiguities in grammars and languages, Pushdown automata (Deterministic and Non-deterministic), Pumping Lemma, Properties of context free languages, normal forms.

Unit 4:	**Turing Machines and Models of Computations**	**10 Lectures**

Turing Machine as a model of computation, Universal Turing Machine, Language acceptability, recursively enumerable and recursive languages.

Suggested Readings:

1. Daniel I.A.Cohen, Introduction to computer theory, John Wiley,1996
2. Lewis & Papadimitriou, Elements of the theory of computation , PHI 1997.
3. Hoperoft, Aho, Ullman, Introduction to Automata theory, Language & Computation –3rd Edition, Pearson Education. 2006
4. P. Linz, An Introduction to Formal Language and Automata 4th edition Publication Jones Bartlett, 2006

Course-SEC Paper:	**Paper Code-COMSMAJ613T Theory of Computation (Tutorial)**	**Credits-1**	**Lab hours/Week-2**

Theory of Computation tutorial as assigned and advised by teacher(s).

3rd Year
Semester-VI

Course-MAJOR Paper Code-COMSMAJ614 Credits-3 Lectures/Week-3
Paper: Design and Analysis of Algorithm

BriefCourseDescription:

Design and Analysis of Algorithms is a core subject in computer science that focuses on developing efficient procedures for solving computational problems and assessing their performance. It involves crafting algorithms using techniques like divide and conquer, dynamic programming, and greedy methods, and evaluating their efficiency through time and space complexity using Big O notation. The subject also includes optimizing algorithms for better performance and exploring complexity classes, such as P and NP, to understand computational limits. Mastery of these concepts is essential for creating effective and efficient software and systems.

Prerequisite(s)and/orNote(s):

Students interested in this course should have had anexposure to introductory courses on programming and data structures. A basic of mathematical logic is essential for success in this course**.**

CourseObjectives:

The objective of learning Design and Analysis of Algorithms is to develop efficient methods for solving computational problems and to evaluate their performance using time and space complexity. It aims to equip students with skills to optimize algorithms for better speed and resource usage while understanding problem complexity and classification. This foundational knowledge is crucial for advancing in computer science and creating effective software solutions.

Knowledgeacquired:

1. Understanding how to measure and improve the efficiency of algorithms.
2. Proficiency in analyzing and comparing time and space complexity.
3. Skills to enhance algorithms for better performance and lower resource consumption.
4. Mastery of various algorithmic techniques such as divide and conquer, dynamic programming, and greedy methods.

Skillsgained:

1. Ability to conceptualize and develop step-by-step procedures for solving complex problems.
2. Proficiency in analyzing the time and space complexity of algorithms to gauge their efficiency.
3. Skills in refining algorithms to enhance their performance and minimize resource usage.
4. Mastery of various Problem-Solving Techniques, such as divide and conquer, dynamic programming, and greedy strategies.
5. Insight into the classification of problems and algorithms based on complexity classes like P, NP,etc.

CompetencyDeveloped:

1. Ability to evaluate and compare the efficiency of different algorithms using time and space complexity metrics.
2. Proficiency in refining and optimizing algorithms to improve performance and resource utilization.
3. Expertise in applying various algorithmic strategies and techniques to effectively solve diverse computational problems.
4. Enhanced capability to analyze and reason about the computational limits and trade-offs of different algorithmic approaches.

Syllabus Overview

Unit 1:	**Introduction**	**6 Lectures**

Definitions, Algorithms vs Programs, Basic AlgorithmDesignTechniques,CorrectnessofAlgorithm,Algorithm classification: Iterativetechniques,DivideandConquer,Dynamic Programming,GreedyAlgorithms, Algorithm analysis (formal and informal), calculation of running time of an algorithm, loop invariants.

Unit 2:	**Growth of Functions**	**7 Lectures**

Complexity of Algorithms, Best, Worst and Average case complexity, growth rate of functions, Asymptotic Analysis, Analyzing algorithm control structures.

Unit 3:	**Recurrences**	**8 Lectures**

Introduction, Substitution method, Iteration method, Master method (Master theorem), Simple problems using Master theorem.

Unit 4:	**Analysis of simple SortingandSearchingalgorithms**	**16 Lectures**

Elementary sorting techniques: Bubble Sort, Selection sort, Insertion Sort, Shell sort, Merge Sort, Advanced Sorting techniques - Heap Sort,QuickSort,SortinginLinearTime-BucketSort,RadixSortandCountSort,SearchingTechniques.

Unit 5:	**Graphs**	**8 Lectures**

GraphAlgorithms–BreadthFirst Search,DepthFirst SearchanditsApplications, Directed acyclic graphs, Single source shortest paths: Dijkstra's algorithm, MinimumSpanningTrees algorithms: Prim's algorithm, Kruskal's Algorithm.

Suggested Readings

1. T.H.Cormen,CharlesE.Leiserson,RonaldL.Rivest,CliffordSteinIntroductionto Algorithms,PHI,3rdEdition 2009
2. Sarabasse& A.V. Gelder Computer Algorithm – Introduction to Design and Analysis, Publisher Pearson 3rdEdition 1999
3. RajeshK.Shukla,AnalysisandDesignofAlgorithms:ABegineersApproach,Wiley
4. Udit Agarwal, Algorithms design and analysis, Dhanpat rai & Co.

Course-SEC Paper:	**Paper Code-COMSMAJ614T** **Design and Analysis of Algorithms (Tutorial)**	**Credits-1**	**Lab hours/Week-2**

Design and Analysis of Algorithms tutorial as assigned and advised by teacher(s).

3rd Year
Semester-VI

Course-MAJOR Paper:	**Paper Code-COMSMAJ615** **Microprocessor**	**Credits-3**	**Lectures/Week-3**

BriefCourseDescription:

The Microprocessor Fundamentals and Programming course provides a thorough introduction to the architecture, functioning, and programming of microprocessors. Students will explore the internal workings of microprocessors, learning how they interact with memory and peripherals to perform various tasks. This course emphasizes both theoretical concepts and practical programming skills, enabling students to develop a solid understanding of microprocessor-based systems.

Prerequisite(s)and/orNote(s):

Students should have a background in basic electronics and digital logic before enrolling in this course. Familiarity with assembly language or a high-level programming language is advantageous. This course is ideal for those pursuing studies in computer engineering, electrical engineering, or related fields.

CourseObjectives:

The primary objectives of the Microprocessor Fundamentals and Programming course are to provide students with a comprehensive understanding of microprocessor architecture and operation. The course aims to teach students how to program microprocessors in assembly language, interface with various hardware components, and design simple microprocessor-based systems. By the end of the course, students will be equipped with the knowledge and skills to develop and troubleshoot microprocessor applications.

Knowledgeacquired:

1. Students will gain detailed knowledge of the CPU, memory hierarchy, and input/output mechanisms.
2. They will learn about instruction sets, addressing modes, and the execution of instructions.
3. The course covers the principles of assembly language programming, enabling students to write and understand low-level code.
4. Students will learn about interfacing microprocessors with external devices, such as sensors and actuators.

Skillsgained:

1. Students will develop practical skills in programming microprocessors using assembly language.
2. They will learn how to write efficient code to control hardware, manage data, and perform arithmetic and logic operations.
3. Students will gain experience in debugging and optimizing assembly programs.
4. They will acquire skills in designing and implementing interfaces between the microprocessor and peripheral devices.

CompetencyDeveloped:

1. Students will have developed the competency to design, program, and troubleshoot microprocessor-based systems.
2. They will be capable of developing assembly language programs to solve complex problems and interface with various hardware components.
3. This competency will enable students to work in fields such as embedded systems, hardware design, and computer engineering.
4. The course provides a strong foundation for advanced studies in microprocessor technology and related areas.
5. Students will be prepared for careers in the rapidly evolving field of computing & electronics

Syllabus Overview

Unit 1:	**Microprocessor 8085 Architecture**	**15 Lectures**

Introduction to Microprocessor – Introduction to Microprocessors, Components of

Microprocessor: Registers, ALU, timing and control, CPU, I/O devices, clock, memory, bussed architecture, tri-state logic, Bus Structure, address bus, data bus and control bus. Block diagram of 8085, pin configuration of 8085, Architecture of Microprocessor 8085, Internal registers (8-bit & 16-bit), CPU, ALU, multiplexing and demultiplexing address/data bus, Instruction Register and Decoder, Timing and Control Unit, Interrupts and Serial I/O.

Unit 2: Instruction Set-I **10 Lectures**

Machine Language and Assembly Language, Addressing modes, types of instruction format, Data Transfer type instructions, Arithmetic and logical instructions, Branching instructions -looping, Timing diagram for opcode fetch, memory read, memory write, I/O read, I/O write.

Unit 3: Instruction Set-II and Programming **10 Lectures**

Special Instructions: Rotate instructions - stack and subroutine related instructions. Assembly Language Programs – Addition, Subtraction, Multiplication (8-bit), Division (8-bit), sorting- Ascending / Descending Order, Largest/Smallest (single byte), Addition of N numbers (single byte).

Unit 4: Memory/IO Interface and Interrupts **10 Lectures**

Memory Interface (Basics) – memory mapped I/O & I/O mapped I/O. Interrupts in 8085- Types, Generation of RST codes-Hardware, software interrupts and their function, Edge triggered and level triggered interrupts, Interrupt priority, Vectored and non-vectored interrupt -SIM and RIM instructions, Comparison of Microprocessor and Microcontroller.

Suggested Readings

1. Microprocessor Architecture, Programming and Application with the8085, Ramesh S. Gaonakar, Penram International Publishing, Mumbai, (2011).
2. Fundamental of Microprocessor 8085: Architecture Programming, and Interfacing, V. Vijayendran, Viswanathan, S., Printers & Publishers Pvt. Ltd (2009).
3. The 8051 Microcontroller, Architecture, Program and application, Kenneth J Ayala, Pen ram
4. Microprocessor Organisation and Architecture, Leventhal L.A , Prentice Hall India.
5. B. Ram, Fundamentals of microprocessors and microcomputers – Dhanpat Rai Publications, New Delhi
6. The 8080/85 Family: Design, Programming & Interfacing, John Uffenbeck, , PHI India.
7. A. K. Ray & K. M.Bhurchandani, Advance Microprocessor and Peripherals, 2ndEdition, Tata McGraw Hill, 2006
8. Mathur A.P., Introduction to Microprocessors. 3rd edn, Tata McGraw, New Delhi,
9. Muhammed Ali Mazidi, Janice Gillispie Mazidi – The 8051 Microcontroller and Embedded systems
10. Microprocessors & Microcontrollers by B. P. Singh, Galgotia publications Pvt. Ltd.

Course-SEC Paper:	**Paper Code-COMSMAJ615L Microprocessor (Lab)**	**Credits-1**	**Lab hours/Week-2**

Students are advised to do laboratory/practical practice not limited to, but includingthe following types of programs

ASSEMBLYLANGUAGE PROGRAMMINGin 8085

1. Writean assemblylanguageprogramto checkwhethera numberiseven orodd.
2. Writean assemblylanguageprogramto findthe numberofones andthenumberof zerosin an8-bit
   number.
3. Writean assemblylanguage programto findthesmaller oftwonumbers.
4. Writeanassemblylanguage programtomultiplytwo8-bit numbers.
5. Write an assembly language program to find the smallest among 10 integers stored in memory
   locationsstartingfrom2050H
6. Write an assembly language program to find the largest among 10 integers stored in memory
   locations startingfrom2050H
7. Writeanassemblylanguageprogramtosort10numbers usingbubble sort.
8. Write an assembly language program to generate the first 10 fibonacci series and store the result at
   memorylocationstartingfromFC50H
9. Writean assemblylanguage programto findthesumof the series 12+22 +32 +42 + …..+ 102
10. Write an assembly language program to find the perfect square of any number and if the number
    is not aperfectsquare,displayFFH
11. Writean assemblylanguage programforlinearsearch.
12. Writean assemblylanguage programto calculatethefollowingexpression usinga singleregister:
    Y=X2+2X+3XZ
13. Writeanassemblylanguageprogramtofindthesumofthe first 10evennaturalnumbers.
14. Writeanassemblylanguage programtofindthesumof thefirst n oddnaturalnumbers.
15. Writean assemblylanguage programto create anoddparitygenerator.
16. Writeanassemblylanguageprogramtocreate anevenparitygenerator.
17. Writean assemblylanguage programtofindthesumof five 8-bit numbers.
18. Writeanassemblylanguageprogramto convert decimal tobinary.
19. Writeanassemblylanguageprogramtoconvertoctal tobinary.
20. Writeanassemblylanguageprogramtoconvert hexadecimaltobinary.
21. Writeanassemblylanguageprogramto convert hexadecimal todecimal.
22. Writeanassemblylanguageprogramtocheckwhetheran8-bitnumber ispalindrome or not.
23. Writeanassemblylanguage programtodisplaythetruthtable forandANDgate.
24. Writeanassemblylanguageprogramtodisplaythetruthtable forand ORgate.
25. Writeanassemblylanguageprogramto displaythetruthtable forandXORgate.
26. Writeanassemblylanguageprogramtoperformnbyteadditionoftwonumbers.
27. Writeanassemblylanguage programtoimplementasimplesubroutinecall.
28. Writean assemblylanguageprogramto checkwhethera number isprimeornot

3rd Year
Semester-VI

Course-MAJOR Paper:	**Paper Code-COMSMAJ616 Computer Graphics**	**Credits-3**	**Lectures/Week-3**

BriefCourseDescription:

Computer Graphics is a field of study that explores the creation, manipulation, and representation of visual images using computers. The course covers foundational concepts such as raster graphics, vector graphics, and image processing techniques. Students learn about algorithms and techniques for rendering 2D and 3D graphics, including shading, lighting, and texture mapping. The course also delves into graphical user interface (GUI) design, animation principles, and virtual reality applications. Practical skills in programming graphics applications and using relevant software tools

are emphasized, preparing students for careers in game development, animation, simulation, and visual effects industries.

Prerequisite(s)and/orNote(s):

To enroll in this course, students should have a basic understanding of several key areas of mathematics and geometry. This course is suitable for beginners and those looking to expand their knowledge in graphics and designing.

CourseObjectives:

The computer graphics course aims to teach students fundamental principles and techniques for creating and manipulating graphical images using computers. It covers algorithms for rendering 2D and 3D graphics, including shading and texture mapping, while emphasizing practical programming skills. Students gain expertise applicable to fields like animation, virtual reality, user interfaces, and interactive simulations, preparing them for careers in game development, animation, and visual effects.

Knowledgeacquired:

1. Understanding of fundamental principles and techniques for creating and manipulating graphical images.
2. Knowledge of algorithms for rendering 2D and 3D graphics, including shading, lighting, and texture mapping.
3. Practical skills in programming graphical applications and implementing rendering algorithms.

Skillsgained:

1. Proficiency in programming graphical applications and implementing rendering algorithms.
2. Skills in applying rendering techniques such as shading, lighting, and texture mapping to create realistic images.
3. Competence in designing intuitive user interfaces and interactive graphical elements.
4. Capability to solve complex visual problems and effectively communicate concepts through computer-generated imagery.

CompetencyDeveloped:

1. Gain proficiency in using graphics software and programming languages to create and manipulate graphical images.
2. Develop skills in designing and integrating visual elements into applications, games, or simulations.
3. Apply problem-solving skills to solve challenges related to visual representation and user interaction.
4. Acquire a foundational understanding of graphics principles and their applications in various domains, such as animation, virtual reality, and user interfaces.

Syllabus Overview

Unit 1: Introduction 4 Lectures

Definition, Basic elements of Computer Graphics, Applications of Computer Graphics, Graphics pipeline, Types of Computer Graphics (Raster and Vector Graphics), Display resolution, aspect ratio and pixel concepts.

Unit 2: Graphics Hardware 7 Lectures

Architecture of Raster and Random scan display devices, input/output devices, Frame buffer and refresh rate, Display technologies (CRT, LCD, LED and OLED), Introduction to Graphics

Processing Unit (GPU), Interactive graphics systems.

Unit 3: Fundamental Techniques in Graphics 15 Lectures

Raster scan line, Circle and ellipse drawing, Thick primitives, Polygon filling, Line and polygon clipping algorithms, 2D and 3D Geometric Transformations, Composite transformations, Homogeneous coordinate system, 2D and 3D Viewing Transformations (Projections- Parallel and Perspective), Window-to-Viewport transformation, Vanishing points, Character generation and text display, Introduction to Anti-Aliasing.

Unit 4: Geometric Modelling 7 Lectures

Representing curves and surfaces, Parametric representation of curves, Bezier curves, B-Spline curves, Polygon mesh representation, Basic surface representation techniques.

Unit 5: Visible Surface Determination & Surface Rendering 12 Lectures

Hidden surface elimination, Back-face detection, Z-buffer method, Scan-line method, Painter's algorithm. Illumination and shading models, Basic color models (RGB, CMY and HSV), Texture mapping concepts, Computer Animation, Key frame animation, Introduction to Ray Tracing and Rendering.

Suggested Readings

1. J D Foley, A. Van Dan, Feiner, Hughes Computer Graphics Principles & Practice 2nd edition
Publication Addison Wesley 1990.
2. D. Hearn, Baker: Computer Graphics, Prentice Hall of India 2008.
3. D F Rogers Procedural Elements for Computer Graphics, McGraw Hill 1997.
4. D F Rogers, Adams Mathematical Elements for Computer Graphics, McGraw Hill 2nd edition
1989.
5. Salini Govil-Pai, Principles of Computer Graphics, University Press

Course-MAJOR Paper: **Paper Code-COMSMAJ616L** **Credits-1** **Lab hours/Week-2**
Computer Graphics (Lab)

Students are advised to do laboratory/practical practice not limited to, but including the following types of programs

1. Writeaprogramtoimplement Bresenham'slinedrawingalgorithm.
2. Writeaprogramtoimplementmid-pointcircledrawingalgorithm.
3. WriteaprogramtoclipalineusingCohenandSutherlandlineclippingalgorithm.
4. WriteaprogramtoclipapolygonusingSutherlandHodgemanalgorithm.
5. Writeaprogramto applyvarious2Dtransformationsona2D object (usehomogenouscoordinates).
6. Writeaprogramtoapplyvarious3Dtransformationsona3Dobject andthenapply parallel andperspectiveprojectionon it
7. WriteaprogramtodrawHermite/Beziercurve