AMRITSAR GROUP OF COLLEGES

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SYLLABUS

B. Tech. (CSE): 5th SEM

5th Semester	AGCS-21501: DESIGN AND ANALYSIS OF ALGORITHMS				
Internal Marks:	**40**		**L**	**T**	**P**
External Marks:	**60**		**3**	**1**	**0**
Total Marks:	**100**		**Credits**		**4**

Course Outcomes: After studying the course, students will be able to:	
CO-1	Understand the concept of asymptotic notations and analysis of algorithms.
CO-2	Gain practical experience in implementing searching and sorting techniques.
CO-3	Acquire skills to implement efficient algorithms using the dynamic programming strategy.
CO-4	Learn and apply various algorithmic techniques such as greedy method and backtracking strategy to solve computational problems.
CO-5	Develop the ability to understand techniques for traversing the graphs to finds shortest path.
CO-6	Classify the problems into class P or NP and explore efficient algorithms for the pattern matching.

Part	Content	CO
I	**Introduction:** Basics of algorithms, time and space complexity of an algorithm, comparing the performance of different algorithms for the same problem, asymptotic notations and order of growth.	CO-1
II	**Sorting and searching techniques:** Merge sort, quick sort and randomized quick sort using divide and conquer strategy, heap sort, counting sort, radix sort, bucket sortand bubble sort with analysis of their running times. Recurrence relations. Linear and binary search in an ordered array, hashing.	CO-2
	Dynamic programming: Matrix chain multiplication, longest common subsequence, 0/1 knapsack problem, fractional knapsack problem, travelling salesman problem.	CO-3
III	**Greedy Strategy:** Minimum spanning trees using Kruskal's and Prim's technique. **Backtracking:** 0/1 knapsack, N queens problem, graph colouring problem.	CO-4
	Graph Algorithms: Graph traversals: Breadth-first search (BFS) and depth-first search (DFS), applications of BFS and DFS, topological sorting, single sourceshortest paths in graphs: Dijkstra and Bellman-Ford, all pair shortest path in graphs: Floyd Warshall's algorithm.	CO-5
IV	**NP-Completeness:** Definition of class NP, NP-hard and NP-complete problems, proving a problem to be NP-complete using polynomial-time reductions, examples of NP complete problems: 3SAT, vertex cover problem, Hamiltonian cycle, clique problem. **Pattern matching algorithms:** Naive String Matcher, Knuth-Morris-Pratt algorithm, Rabin Karp algorithm.	CO-6

References:

- Introduction to Algorithms 2nd Edition by Thomas H. Cormen, Charles E. Leiserson, Ronald L.Rivest, and Clifford Stein, The MIT Press Cambridge , McGraw-Hill Book Company.
- Fundamentals of Computer Algorithms 2nd Edition by Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran, Silicon Press, 2008.
- The Design and Analysis of Algorithms by Nitin Upadhyay, S. K. Kataria& Sons, 2008.
- The Design and Analysis of Algorithms, 3rd Edition by Gajendra Sharma, Khanna Book Publishing Company, Delhi

5th Semester	AGCS-21502: SOFTWARE ENGINEERING				
Internal Marks:	**40**		**L**	**T**	**P**
External Marks:	**60**		**3**	**1**	**0**
Total Marks:	**100**		**Credits**		**4**

Course Outcomes: After studying the course, students will be able to:	
CO-1	Understand the processes and models involved in SDLC lifecycle.
CO-2	Understand software requirements specification and design.
CO-3	Understand the role of project management planning
CO-4	Implement different coding standards and software testing approaches such as unit testing and integration testing.
CO-5	Understand the basics software quality strategies.
CO-6	Understand the role of project risk management, ethical and professional issues.

Part	Content	CO
I	Evolution and impact of Software engineering, introduction to agile software development, software life cycle models: Waterfall, prototyping, Evolutionary, and Spiral models. Feasibility study.	CO-1
II	Functional and Non-functional requirements, Requirements gathering, Requirements analysis and specification. Function-oriented software design: DFD and Structure chart, Object modeling using UML, Object-oriented software development, user interface design, Basic issues in software design, modularity, cohesion, coupling and layering. Coding standards and Code review techniques and tools. Integrated development environments (IDEs).	CO2
	Software project management, Project planning and control, size and cost estimation, project scheduling using PERT and Gantt chart.	CO-3
III	Fundamentals of testing, White-box, and black-box testing, test case design techniques, bug tracking system, mutation testing,Automated build and deployment tools, Tool: Selenium. Static and dynamic analysis, verification and validation, Software reliability metrics, reliability growth modelling.	CO4
	Software quality assurance: quality concepts, quality control, quality assurance, SQA activities, Software reviews, Formal Technical Reviews, Review guidelines. Quality Assurance Standards: ISO 9000, 9001:2000, CMM, TQM and Six Sigma.	CO-5
IV	Introduction to SCM, Version Control and Change Management, Risk Mitigation, Monitoring and Management (RMM), Computer aided software engineering, software maintenance, Integrated Change Control, software reuse, Component-based software development. Software engineering ethics and professionalism- ethical and legal issues in software engineering, intellectual property rights and plagiarism, professional code of conduct, social and economic impact of software.	CO-6

References:

- Roger Pressman, "Software Engineering: A Practitioners Approach,(6th Edition), McGrawHill,1997.Sommerville," Software Engineering, 7th edition", Adison Wesley, 1996.
- Watts Humphrey," Managing software process", Pearson education, 2003.
- James F. Peters and Witold Pedrycz, "Software Engineering – An Engineering Approach", Wiley.
- Mouratidis and Giorgini. "Integrating Security and Software Engineering–Advances and Future",
- IGP. ISBN – 1-59904-148-0.
- Pankaj Jalote, "An integrated approach to Software Engineering", Springer/Narosa.

5th Semester	AGCS-21503: PROGRAMMING IN JAVA				
Internal Marks:	**40**		**L**	**T**	**P**
External Marks:	**60**		**3**	**0**	**0**
Total Marks:	**100**		**Credits**		**3**

Course Outcomes: After studying the course, students will be able to:	
CO-1	Understand Object-Oriented Programming constructs, Byte codes, basics of java console and programming concepts.
CO-2	Implementation of Classes, Inheritance, and Packages.
CO-3	Developing logic for problem solving with String handling and Exception handling.
CO-4	Developing simple java applications with JDBC connectivity.
CO-5	Understand and utilize Java Graphical User Interface in the program writing.
CO-6	Utilize the knowledge of Multithreading and Networking to develop java applications.

Part	Content	CO
I	**Overview of Java:** Object oriented programming concepts and features, Introduction to Java, Features of Java, Bytecode, Lexical Issues, Java class libraries. **Data types and Variables:** Integers, Floating-point types, Characters, Boolean, Java Literals-Integer, Floating Point, Boolean, Character, String, Variable, Data types, Type Conversion and Casting, Automatic type promotion in expressions. **Operators, Control Statements and arrays:** Arithmetic operators, Modulus Operator, Increment, Decrement, Bit wise operators, Relational operators, Boolean logical operators?: Operators, Operator precedence, Java's selection statements, Iteration statements, Jump statements, Arrays in Java (1 D and 2 D).	CO-1
	Introduction to Classes: Class fundamentals, Declaring object, reference variable, Introducing methods, Constructors, this keyword, Garbage collection, Finalize () method. **Methods and Classes:** Overloading methods and constructors, Using objects as parameters, Recursion. Access Control, Usage of static keyword, Nested Classes, Command-Line arguments, Variable-Length arguments **Inheritance:** Inheritance basics, using super, method overriding, dynamic method dispatch, Abstract Classes, Using final keyword. **Package and Interfaces-** Creating a package, Importing a package, Defining and Implementing interfaces, Extending interfaces.	CO-2
II	**String Handling:** The string constructors, String length, Special string operations-String Literal, Concatenation, Conversion Character extraction, String comparison, Searching Strings, Modifying string, Data conversion, Changing the case of characters, Joining Strings, StringBuffer class, StringBuffer constructors, StringBuffer methods- length(), capacity(), ensureCapacity(), setLength(), setLength(), charAt(), setCharAt(), getChars(), append(), insert(), reverse(), , delete(), deleteCharAt(), , replace(), substring(). **Exception Handling:** Exception handling fundamentals, Exception types, Uncaught exceptions Using try and catch, Multiple catch clauses, Nested try statements, Throw, Finally Java 's built-in exceptions, Creating your own exception, Chained Exceptions, Assertions.	CO-3
III	**Database Connectivity:** JDBC drivers, Driver Manager Class, Connection interface, Statement interface, ResultSet interface, Query Execution.	CO-4
	Event Handling: Delegation event model, Sources of Events, Event listener interfaces, Adapter classes.	CO-5

	Swings: Introduction to Swings, JFrame, JApplet, JPanel, Components in Swings, Layout managers- Springlayout, Boxlayout, JList, JScrollPane, Split Pane, JTabbedPane, JTree, JTable	
IV	**Multithreaded Programming:** The java thread model, The main thread, Creating threads, Using isalive() and join(), Thread priorities, Synchronization; Inter thread communications, Suspending,Resuming and Stopping threads. **Networking:** Networking basics, Java and Net, Datagram, TCP/IP Client Sockets, TCP/IP Server Sockets, URL, URL Connection.	CO-6

References:

- Herbert Schildt, The Complete Reference Java 2, McGraw-Hill.
- Joycee Farrell, Java For Beginners, Cengage Learning
- Deitel and Deitel, Java: How to Program, 6th Edition, Pearson Education

5th Semester	AGCS- 21505: ARTIFICIAL INTELLIGENCE				
Internal Marks:	**40**		**L**	**T**	**P**
External Marks:	**60**		**3**	**0**	**0**
Total Marks:	**100**		**Credits**		**3**

Course Outcomes: After studying the course, students will be able to:	
CO-1	Understand the basics of AI and its ethical considerations, problem solving and state space search.
CO-2	Understand, classify and implement various search techniques.
CO-3	Understand the game playing algorithm and the significance of planning in AI.
CO-4	Understand and represent knowledge using logic and structures
CO-5	Understand the concept of reasoning & inferencing and get acquainted to natural language processing.
CO-6	Understand the significance of neural network and learning in developing AI systems

Part	Content	CO
I	**Introduction to AI:** Definition of Intelligence, Definition, history and importance of Artificial Intelligence, components of AI, categories of AI machines, Turing Test, Recent AI Applications, **Ethical Considerations in AI:** Bias and fairness in AI system, privacy and security concerns, accountability and transparency, social and economic impact of AI **Problem Solving**: Definition & Characteristics of problem, formulating problems, problem solving agents, problem types, states and operators, state space search, Common Problems: Water Jug Problem, Travelling Salesman Problem, Tower of Hanoi, Tic Tac Toe, 8 puzzle game, Chess.	CO-1
	Heuristic Based Search: search strategies, uninformed search, Breadth first search, Depth first search, informed search, Heuristic Search, heuristic function, Generate and Test, Best first search, A* algorithm, Problem Reduction, AO* Search, Hill Climbing, Constraint Satisfaction Problem, Crypt arithmetic problems, Means End Analysis.	CO-2
II	**Game Playing** - Perfect decision game, imperfect decision game, components of game, game tree, evaluation function, MinMax Algorithm, problem in MinMax algorithm, alpha-beta pruning. **Planning** - Definition, linear & non linear planning, reactive & non reactive systems, Components of planning, planning in the blocks world, partial order planning, hierarchical planning, conditional planning, resource constraints, temporal constraints blocks world problem	CO-3
III	**Knowledge Representation using Propositional and First Order Predicate Logic:** syntax and semantics for propositional logic, syntax and semantics for FOPL, properties of WFFs, conversion to casual form, inference rules, resolution using proposition logic and predicate logic. **Knowledge Representation using Structures**: Semantic nets, frames, conceptual dependency, scripts.	CO-4
	Inferencing & Reasoning: logical reasoning, Modus Ponen Rule, Modus Tollen Rule, inductive reasoning, deductive reasoning, formal reasoning, analogical reasoning, mono tonic and non monotonic reasoning, probabilistic reasoning, forward chaining, backward chaining, resolution, unification. **Uncertainty** - Basic probability, Bayes rule, Belief networks, Markov Decision processes **Natural Language Processing:** Introduction to natural language processing, applications of NLP, techniques used in NLP, components of NLP: natural language generation and natural	CO-5

	language understanding, phases of NLP: lexical analysis, syntactic analysis, semantic analysis, disclosure integration, pragmatic analysis.	
IV	**Neural Networks and Learning:** Basics, comparison of human brain and machine, biological neuron, need of Artificial Neural Network(ANN), artificial neuron model, fundamentals of ANN, architectures of ANN: feed forward network, single layer and multilayer feed forward networks, recurrent network **Learning Methods in Neural Networks:** Introduction to Supervised learning, unsupervised learning and reinforcement learning.	CO-6

References:

- Stuart Russell and Peter Norvig. Artificial Intelligence – A Modern Approach, Pearson Education 2 Press, 2001.
- Kevin Knight, Elaine Rich, B. Nair, Artificial Intelligence, McGraw Hill, 2008.
- George F. Luger, Artificial Intelligence, Pearson Education, 2001.
- Nils J. Nilsson, Artificial Intelligence: A New Synthesis, Morgan Kauffman, 2002

5th Semester	AGCS-21506: DESIGN AND ANALYSIS OF ALGORITHMS LAB				
Internal Marks:	**30**		**L**	**T**	**P**
External Marks:	**20**		**0**	**0**	**2**
Total Marks:	**50**		**Credits**		**1**

Course Outcomes: After studying the course, students will be able to:	
CO-1	Understand the trade-offs involved in various sorting algorithm techniques.
CO-2	Improve efficiency of searching and sorting algorithms using Divide and Conquer strategy.
CO-3	Develop the ability to apply dynamic programming algorithms for various computational problems.
CO-4	Acquire skills to design minimum spanning tree using various techniques.
CO-5	Understand the implementation of string-matching Algorithms.
CO-6	Develop an application using various algorithms studied in the course.

Part	Experiment	CO
A	Code and analyse insertion sort for best and worst case as an implementation for algorithm analysis. Code and analyse heap sort, counting sort, radix sort.	CO-1
	Code and analyse quick sort as an implementation for divide and conquer strategy. Code and analyse randomized quick Sort as an implementation for divide and conquer strategy. Code and analyse merge sort as an implementation for divide and conquer strategy.	CO-2
	Code and analyse optimal matrix chain multiplication, LCS as an implementation for dynamic programming.	CO-3
	Code and analyse Kruskal's technique and Prim's technique for finding a minimum spanning tree as an implementation for greedy strategy.	CO-4
	Code and analyse for finding occurrences of pattern in a string using brute force approach and Rabin Karp approach.	CO-5
B	Implementation of sum of subsets problem, knapsack problem, travelling salesman problem, N queen problem, graph colouring problem, Dijkstra's algorithm, Bellman Ford algorithm ,Floyd Warshall's algorithm.	CO-6

5th Semester	AGCS-21507 SOFTWARE ENGINEERING LAB				
Internal Marks:	**30**		**L**	**T**	**P**
External Marks:	**20**		**0**	**0**	**2**
Total Marks:	**50**		**Credits**		**1**

Course Outcomes: After studying the course, students will be able to:	
CO-1	Use Openproj tool to track the progress of project.
CO-2	Prepare SRS document, design document.
CO-3	Implement different software designs using suitable software tools.
CO-4	Prepare test cases and implement different testing techniques
CO-5	Prepare Software configuration management and risk management related document.
CO-6	Make a mini project that demonstrates a concept, based on Software Engineering.

Part	Experiment	CO
A	Study and usage of OpenProj or similar software to draft a project plan Study and usage of OpenProj or similar software to track the progress of a project	CO-1
	Preparation of Software Requirement Specification Document, Design Documents and Testing Phase related documents for some problems	CO-2
	Study and usage of any Design phase CASE tool (UML Diagrams, DFD)	CO-3
	To perform unit testing and integration testing. To perform various white box and black box testing techniques. Testing of a web site.	CO-4
	Preparation of Software Configuration Management and Risk Management related documents	CO-5
B	To make a mini project that demonstrates a concept, based on the content of AGCS-21507(Software Engineering Lab)	CO-6

5th Semester	AGCS-21508: PROGRAMMING IN JAVA LAB				
Internal Marks:	**30**		**L**	**T**	**P**
External Marks:	**20**		**0**	**0**	**2**
Total Marks:	**50**		**Credits**		**1**

Course Outcomes: After studying the course, students will be able to:	
CO-1	Gain knowledge about Java Runtime Environment and basic concepts
CO-2	Applying Object-Oriented Programming (OOP) Concepts
CO-3	Implement String and Exception handling.
CO-4	Learn about developing Graphical User Interface and Java Database Connectivity
CO-5	Implement multithreading and networking in Java
CO-6	Develop an application deploying java concepts

Part	Content	CO
A	Installation of Java, Setting of Environment variables, Implementation of Java on different IDEs (Eclipse/NetBeans/VS Code) Executing basic java programs using decision statements and Loops, Type Casting , Operators and Arrays	CO-1
	Implementing the concept of Classes and Objects, Constructor, Constructor Overloading Usage of static keyword, Nested classes, Command Line Arguments Implementation of Inheritance, Method overriding, Abstract classes and Interfaces Creating own packages, Importing Packages	CO-2
	Exploring String and String Buffer class and their methods. Implement Exception handling and creating own exceptions	CO-3
	Implementation of Java Database Connectivity Exploring Swing components and its classes	CO-4
	Implement Multithreading Explore java.net package to develop networking based applications.	CO-5
B	**Students are required to build one application based on the concepts implemented inPart-A. The applications can be (but not limited to):** ➢ GUI based with Java Database Connectivity • Student Database Management System • Library Management System • Any other using similar concepts ➢ Develop Notepad/Paint ➢ Menu Driven GUI based application ➢ Chat based application	CO-6

5th Semester	AGCS- 21509: ARTIFICIAL INTELLIGENCE LAB				
Internal Marks:	**30**		**L**	**T**	**P**
External Marks:	**20**		**0**	**0**	**2**
Total Marks:	**50**		**Credits**		**1**

Course Outcomes: After studying the course, students will be able to:	
CO-1	Understand and implement uninformed search techniques.
CO-2	Understand and implement informed search techniques.
CO-3	Understand and solve crypt arithmetic problems.
CO-4	Understand and implement common AI problems.
CO-5	Understand and implement common gaming problems.
CO-6	Develop AI based applications.

Part	Experiment	CO
A	• To implement Breadth First Search • To implement Depth First Search	CO-1
	• To implement Best First Search • To implement A* algorithm • To implement Hill Climbing algorithm	CO-2
	• To solve the crypt arithmetic problems	CO-3
	• To implement Water Jug problem • To implement Tower of Hanoi	CO-4
	• To implement n-queens problem • To implement Tic Tac Toe • To implement 3x3 puzzle/8 puzzle problem	CO-5
B	To make a mini project on AI based application/ Gaming application	CO-6

5th Semester	AGAP-21502: ENGINEERING APTITUDE - II				
Internal Marks:	**50**		**L**	**T**	**P**
External Marks:	**0**		**0**	**1**	**0**
Total Marks:	**50**		**Credits**		**1**

Course Outcomes: After studying the course, students will be able to:	
CO-1	Learn and practice Aptitude questions based on *"Problems on Ages"* and improve their skills in order to face the interview, competitive exams.
CO-2	Understand the relationships among things or finite groups of things.
CO-3	Outline the various formulas for calculating area , volume and surface area.
CO-4	Use a calendar to determine a Date and Day.
CO-5	Use a time schedule to determine ending time of a given event.
CO-6	Find out missing part of an element by subsequent comparison.

Part	Content	CO
I	**Problem on Ages**: Shortcut method to simplify questions based on Age	CO-1
	Venn Diagrams: Applications of Sets	CO-2
II	**Area , volume and surface area** : Cuboid, Cube, Parallelepiped, Cylinder, Sphere	CO-3
III	**Calendar and Time** :To find odd days in an ordinary year, Leap year, Days of week related to odd days	CO-4
	Clocks : Hands of Clock, Angle Traced by Hands	CO-5
IV	**Chain Rule** :Direct Proportion **,**Indirect Proportion	CO-6

References:

- Quantitative Maths: Arihant Publishers.
- Objective Mathematics: R S Aggarwal.
- Quantitative Maths: TMH Publications

5th Semester	AGCS-21504A: THEORY OF COMPUTATIONS(PEC-1)				
Internal Marks:	**40**		**L**	**T**	**P**
External Marks:	**60**		**3**	**1**	**0**
Total Marks:	**100**		**Credits**		**4**

Course Outcomes: After studying the course, students will be able to:	
CO1	Understand the basic concepts, design of finite automata and their applications.
CO2	Illustrate the formal languages and grammar types.
CO3	Demonstrate the relationship between regular sets and regular grammar.
CO4	Understand the concepts of context-free languages and different types in normal forms.
CO5	Familiarize and design pushdown automata and Turing machines for performing tasks of moderate complexity.
CO6	Outline the formal properties and definition of LL (k) and LR (k) grammars, Decidability, Recursively Enumerable Languages and PCP.

Part	Content	CO
I	**Basics of Strings and Alphabets**- Definition of string, null string, substring, alphabet, power of alphabet, The Kleene Star, languages. **Finite Automata:** Applications of automata theory, Introduction to DFA, Transition Diagram, Transition table, String acceptability by FA, non-deterministic FA, Conversion of NDFA to DFA, Comparison of DFA and NDFA, Design of Finite Automata, Mealy and Moore Machines and its conversion, Comparison of Mealy and Moore Machine, Two-way Finite automata (2DFA), Definition of dead state and unreachable state, Minimization of FA.	CO-1
II	**Formal Languages:** Definition of grammar, Derivation & language generated by grammar, Chomsky Classification of Languages, Languages and automata.	CO-2
	Regular Sets & Regular Grammar: Definition of Regular expressions and regular sets, Identities of regular expressions, Proof of Arden's Theorem, Finite Automata and Regular Expressions - Transition Systems and Regular Expressions, Conversion of Non deterministic Systems to Deterministic Systems, Algebraic Method using Arden's Theorem, Construction of finite automata equivalent to a regular expression, Equivalence of two finite automata, Equivalence of two regular expressions, Regular sets and regular grammar – Construction of a Regular grammar generating T(M) for a given DFA, Construction of a transition system accepting L(G) for a given regular grammar.	CO-3
III	**Context Free Languages (CFG):** Definition of Derivation tree, Leftmost and Rightmost derivation, ambiguity in CFG, Simplification of CFG - Construction of Reduced Grammars, Elimination of Null Productions, Elimination of Unit Productions, Normal forms for CFG- Chomsky Normal Form, Greibach Normal Form, Kuroda Normal Form, Conversion of CFG to CNF, Conversion of CFG to GNF.	CO-4
	Pushdown Automata (PDA): Definition of PDA, Acceptance by PDA, Definition of PDA with two stacks, context free languages and PDA, Design of PDA. **Turing Machines (TM):** Turing Machine Model, Definition of TM, Types of TM, String acceptability by Turing Machine, Representation of Turing Machine, Design of Turing Machine.	CO-5
IV	**Recursive And Recursively Enumerable Languages:** Definition of LL(k) Grammar and LR(k) Grammar, Properties of LL(k) Grammar and LR(k) Grammar, Difference between LL(k) & LR(k) grammar, Decidability, Recursively Enumerable Languages and Post Correspondence Problem.	CO-6

References:

- K.L.P. Mishra and N. Chandrasekaran, "Theory of Computer Science, Third Edition", PHI Learning Private Limited, 2011.
- John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, "Introduction to Automata Theory", Languages and Computation, Pearson Education.
- M. Sipser, "Introduction to the Theory of Computation", Second Edition, Cengage Learning.
- K. V. N. Sunitha, N. Kalyani, "Formal Languages and Automata Theory", McGraw-Hill, 2010.

5th Semester	AGCS-21504B: COMPILER DESIGN(PEC-1)				
Internal Marks:	**40**		**L**	**T**	**P**
External Marks:	**60**		**3**	**1**	**0**
Total Marks:	**100**		**Credits**		**4**

Course Outcomes: After studying the course, students will be able to:	
CO-1	Demonstrate the knowledge of patterns, tokens & regular expressions for lexical analysis.
CO-2	Use lex & yacc tool for developing a scanner and parser.
CO-3	Understand syntax directed translation.
CO-4	Analyse different representations of intermediate code.
CO-5	Apply type checking in values.
CO-6	Understand code optimization and code generation.

Part	Content	CO
I	**Introduction to compilers:** Definition of compiler, interpreter and its differences, the phases of a compiler, role of lexical analyzer, regular expressions, finite automata, from regular expressions to finite automata, pass and phases of translation, bootstrapping, LEX-lexical analyzer generator. **Parsing:** Parsing, role of parser, context free grammar, derivations, parse trees, ambiguity, elimination of left recursion, left factoring, eliminating ambiguity from dangling-else grammar, classes of parsing, top down parsing - backtracking, recursive descent parsing, predictive parsers, LL(1) grammars.	CO-1
II	**Bottom-up parsing:** Definition of bottom-up parsing, handles, handle pruning, stack implementation of shift-reduce parsing, conflicts during shift-reduce parsing, LR grammars, LR parsers-simple LR, canonical LR(CLR) and Look Ahead LR (LALR) parsers, error recovery in parsing, parsing ambiguous grammars, YACC-automatic parser generator.	CO-2
	Syntax directed translation: Syntax directed definition, construction of syntax trees, S-attributed and L-attributed definitions, translation schemes, emitting a translation.	CO-3
III	**Intermediate code generation:** intermediate forms of source programs abstract syntax tree, polish notation and three address code, types of three address statements and its implementation, syntax directed translation into three-address code, translation of simple statements, Boolean expressions and flow-of-control statements.	CO-4
IV	**Type checking:** Definition of type checking, type expressions, type systems, static and dynamic checking of types, specification of a simple type checker, equivalence of type expressions, type conversions, overloading of functions and operators. **Run time environments:** Source language issues, Storage organization, storage-allocation strategies, access to non-local names, parameter passing, symbol tables and language facilities for dynamic storage allocation.	CO-5
	Code optimization: Organization of code optimizer, basic blocks and flow graphs, optimization of basic blocks, the principal sources of optimization, the directed acyclic graph (DAG) representation of basic block, global data flow analysis. **Code generation:** Machine dependent code generation, object code forms, the target machine, a simple code generator, register allocation and assignment, peephole optimization.	CO-6

References:

- Alfred V. Aho, Jeffrey D. Ullman (2001), Principles of compiler design, Indian student edition, Pearson Education, New Delhi, India.
- Andre W. Appel (2004), Modern Compiler Implementation C, Cambridge University Press, UK

.pl

5th Semester	AGCS-21504D: DISTRIBUTED SYSTEMS(PEC-1)				
Internal Marks:	**40**		**L**	**T**	**P**
External Marks:	**60**		**3**	**1**	**0**
Total Marks:	**100**		**Credits**		**4**

Course Outcomes: After studying the course, students will be able to:	
CO-1	Understand the foundations of a Distributed System.
CO-2	Implement inter process communication in distributed environment.
CO-3	Learn issues related to clock synchronization and need of global state in distributed systems.
CO-4	Manage transactions and concurrency control.
CO-5	Understand distributed deadlock and transaction recovery.
CO-6	Design and implement shared memory management in distributed computing.

Part	Content	CO
I	**Characterization of Distributed Systems:** Introduction, Examples of Distributed systems, Resource sharing and web, challenges, System models -Introduction, Architectural and Fundamental models. Networking and Internetworking.	CO-1
II	**Interprocess Communication, Distributed objects, and Remote Invocation:** Introduction, Communication between distributed objects, RPC, Events and notifications, Case study-Java RMI **Operating System Support**- Introduction, OS layer, Protection, Processes and Threads, Communication and Invocation, Operating system architecture.	CO-2
	Peer to Peer Systems: Introduction, Napster and its legacy, Peer to Peer middleware, Routing overlays, Overlay case studies, Tapestry, Application case studies: Squirrel and Ocean Store. **Time and Global States:** Introduction, Clocks, Events, and Process states, Synchronizing physical clocks, logical time and logical clocks, global states, distributed debugging. **Coordination and Agreement:** Introduction, Distributed mutual exclusion, Elections, Multicast communication, consensus, and related problems.	CO-3
III	**Distributed File Systems:** Introduction, File Service architecture. **Transactions and Concurrency Control:** Introduction, Transactions, Nested Transactions, Locks, Optimistic concurrency control, Timestamp ordering.	CO-4
IV	**Distributed Transactions:** Introduction, Flat and Nested Distributed Transactions, Atomic commit protocols, Concurrency control in distributed transactions, Distributed deadlocks, Transaction recovery.	CO-5
	Replication: Introduction, System model and group communication, Fault-tolerant services, Transactions with replicated data. Distributed shared memory, Design and Implementation issues, and Consistency models.	CO-6

References:

- Distributed Systems Concepts and Design, G Coulouris, J Dollimore and T Kindberg, Fourth Edition, Pearson Education.
- Distributed Systems, S.Ghosh, Chapman & Hall/CRC, Taylor & Francis Group, 2010.
- Distributed Systems – Principles and Paradigms, A.S. Tanenbaum and M.V. Steen, Pearson Education.
- Distributed Computing, Principles, Algorithms and Systems, Ajay D. Kshemakalyani and Mukesh Singhal, Cambridge, rp 2010