MSc. Computer Science Syllabus Structure

Department of Computer Science & IT

MANGALDAI COLLEGE (AUTONOMOUS)

A: Structure 1 (Level 6.5): PG Curricular Structure with only Course Work

Semester	Type		Paper	Credit (T+P)
I	Discipline Core Course -1		Advanced Data Structure	3+1=4
	Discipline Core Course-2		Machine Learning Algorithms	3+1=4
	Discipline Core Course -3		Database Systems and Applications	3+1=4
	Discipline Core Course -4		Research Methodology	4
	One Discipline Specific Elective Course Can be taken	Elective – 1	Advanced Computer Architecture	4
		Elective – 2	UI/UX Engineering	3+1=4
		Elective – 3	Probability, Statistics & Data Analytics	4
				Total Credit: **20**
II	Discipline Core Course - 5		Advanced Operating System	3+1=4
	Discipline Core Course -6		Compiler Design	4
	Discipline Core Course -7		Seminar/Minor Project	4
	Two Discipline Specific Elective Course Can be taken	Elective – 1	Full Stack Development	3+1=4
		Elective – 2	Blockchain Technology	4
		Elective – 3	Mobile Application Development	3+1=4
		Elective – 4	Cyber Security & Ethical Hacking	3+1=4
		Elective – 5	Advanced Algorithms	4
		Elective – 6	Computer Graphics	3+1=4
				Total Credit: **20**
III	Discipline Core Course -8		Deep Learning	4
	Discipline Core Course -9		Cryptography and Network Security	4
	Discipline Core Course -10		Theory of Computation	4
	Two Discipline Specific Elective Course Can be taken	Elective – 1	Speech Processing	3+1=4
		Elective – 2	Digital Image Processing	3+1=4
		Elective – 3	Advanced Web Technology	3+1=4
		Elective – 4	Generative AI and Large Language Models	3+1=4
		Elective – 5	Data Science	3+1=4
				Total Credit: **20**
IV	Discipline Core Course -11		Optimization Techniques	4
	Project/Dissertation		Dissertation / Internship + Major Project	4
	Three Discipline Specific Elective Course Can be taken	Elective – 1	Distributed and Cloud Systems	4
		Elective – 2	Natural Language Processing	4
		Elective – 3	Big Data Analytics	4
		Elective – 4	Augmented Reality	3+1=4
		Elective – 5	Advanced Cyber Security	3+1=4
				Total Credit: **20**

Course Information		
Course Title	:	**Advanced Data Structure**
Course Code	:	**CSC-CC-201014**
Programme	:	Master of Science in Computer Science
Semester	:	1st Semester
Credit	:	3+1=4
Marks	:	**Component** / **Marks** Internal Assessment / 30 ESE / 45 Practical / 25
Course Objective	:	The course is designed to provide students with a systematic understanding of **advanced data structures, algorithm design techniques, and computational complexity**. On successful completion of the course, students will be able to: • **Understand and apply advanced data structures** such as AVL trees, Red-Black trees, B-trees, B+ trees, heaps, binomial heaps, and hashing techniques. • **Analyze and implement fundamental operations** such as insertion, deletion, searching, balancing, Union-Find, and priority queue operations. • **Understand lower-bound concepts** and analyze the fundamental limitations of algorithms for searching, sorting, selection, and related problems. • **Apply appropriate algorithm design techniques**, including Divide and Conquer, Dynamic Programming, Greedy Algorithms, and Backtracking, to solve computational problems. • **Analyze the time and space complexity** of algorithms and compare alternative approaches based on their efficiency. • **Apply data structures and algorithms to practical problems**, including Minimum Spanning Tree, Knapsack, Job Sequencing, sorting, and searching. • **Understand computational complexity classes** such as P, NP, and co-NP, and explain the concepts of polynomial-time reducibility, NP-Completeness, and NP-Hardness. • **Develop problem-solving and algorithmic thinking skills** required to design efficient solutions to complex computational problems.
Course Outcomes	:	After successful completion of the course, students will be able to: CO1 **: Understand and apply** advanced data structures such as balanced search trees, B-trees, B+ trees, heaps, and hashing. CO2 **Analyze the efficiency** of algorithms using time complexity and lower-bound concepts. CO3 **Implement and apply** priority queues and Union-Find data structures to solve computational problems. CO4 **Apply algorithm design techniques** such as Divide and Conquer, Dynamic Programming, Greedy Algorithms, and Backtracking to solve problems. CO5 **Analyze graph algorithms** such as Prim's and Kruskal's algorithms for solving Minimum Spanning Tree problems. CO6 **Understand computational complexity** and distinguish between P, NP, NP-Complete, and NP-Hard problems.

		CO7 **Develop efficient problem-solving skills** by selecting appropriate data structures and algorithmic techniques for a given problem.
Recommended Book & References	:	1. Cormen T. H., Leiserson C. E., Rivest R. L.; Introduction to Algorithms; Tata-McGraw Hill Publishers 2. Aho A., Hopcroft J. E., Ullman J. D.; Data Structures and Algorithms; Addison-Wesley 3. Horowitz, Sahani; Fundamentals of Data Structures in C/C++; Computer Science Press 4. Aho A., Hopcroft J. E., Ullman J. D.; Design and Analysis of Computer Algorithms ; Addison-Wesley

Course Content / Units				
Theory			**Class**	**Marks**
Unit 1	:	**Dictionary ADT**	10	10
		Search trees and balancing of search trees – **AVL tree and Red-Black tree**. Multi-way and external storage trees – **B-tree and B+ tree**. Insertion and deletion in the above data structures. **Hashing and collision resolution techniques.**		
Unit 2	:	**Lower Bound Theory**	20	20
		Lower bounds on simple problems like finding **maximum and minimum, maximum and second maximum, sorting and searching problems**. Medians and order statistics. **Selection arguments.**		
Unit 3	:	**Priority Queue ADT**	15	20
		Revision of heap (**max and min**) data structure and operations such as **insertion, deletion, decrease-key and increase-key**. Heap-based priority queue. **Binomial heap**, its operations and analysis.		
Unit 4	:	**Partition ADT**	20	20
		Union-Find algorithms through **weighted merge and path compression**. Operations of Union and Find and their applications in **Minimum Spanning Tree algorithms**.		
Unit 5	:	**Algorithm Design Techniques**	15	15
		Algorithm design techniques – **Divide and Conquer, Dynamic Programming, Greedy Algorithm and Backtracking**. Illustration of design techniques by application to problems such as: **sorting and searching, MST (Minimum Spanning Tree) problem and algorithms due to Prim and Kruskal, knapsack problem and job sequencing problem.**		
Unit 6		**Theory of NP Completeness**	10	15
		Formal language framework, complexity classes – **P, NP and co-NP**. **Polynomial-time reducibility, NP-Completeness and NP-Hard problems.**		

Practical / Lab Work	Class	Marks
List of Practical 1. Implement binary search tree using array by writing a program to: a. Create a binary search tree using array b. Print the prefix traversal of the BST c. Print the infix traversal of the BST d. Print the postfix traversal of the BST e. Search for an element in the BST 2. Implement binary search tree using linked list by writing a program to: a. Create a binary search tree using linked list b. Print the prefix traversal of the BST c. Print the infix traversal of the BST d. Print the postfix traversal of the BST e. Search for an element in the BST 3. Write a program to transform an entire unsorted array of size n into a heap, 4. Implement a Heap (max / min) using array and write programs to a. Insert an element b. Delete the root of a heap and use heapify operation c. Update a heap using decrease-key and increase-key operation. 5. Write a program to implement the Heapsort algorithm. 6. Implement a Red Black tree and write functions to a. search for an element b. insert an element c. delete an element 7. Implement an AVL tree and write functions to a. search for an element b. insert an element c. delete an element 8. Write a program to implement Prim's algorithm for MST using Heap structure 9. Write a program to implement Kruskal's algorithm for MST using Heap structure 10. Implement find-union operation using a suitable data structure.	30	

Course Information			
Course Title	:	**Machine Learning Algorithm**	
Course Code	:	**CSC-CC-201024**	
Programme	:	Master of Science in Computer Science	
Semester	:	1st Semester	
Credit	:	3+1=4	
Marks	:	**Component**	**Marks**
		Internal Assessment	30
		ESE	45
		Practical	25
Course Objective	:	At the end of the course, students will be able to: • Explain the fundamental concepts of machine learning. • Differentiate between supervised and unsupervised learning techniques. • Implement suitable machine learning algorithms for solving real-world problems using relevant datasets. • Evaluate the performance of machine learning models using appropriate metrics and interpretation techniques.	
Course Outcomes	:	Upon completion of the course, students will be able to: CO1 Understand what is machine learning and the major machine learning approaches CO2 Differentiate between supervised and unsupervised learning CO3 Understand linear and non- linear classification. CO4 Understand the role and importance of feature extraction CO5 Apply machine learning algorithms to real world problems.	
Recommended Book & References	:	1. J. Shavlik and T. Dietterich (Ed), Readings in Machine Learning, Morgan Kaufmann, 1990. 2. Chris Albon, Machine Learning with Python Cookbook, Pub: Shroff/O'Reilly, 2018. 3. Shai Shalev-Shwartz and Shai Ben-David; Understanding Machine Learning: From Theory to Algorithms. E-book; https://kdnuggets.com 4. NPTEL	

Course Content / Units				
Theory			**Class**	**Marks**
Unit 1	:	**Introduction to Machine Learning**	12	12
		Definition and scope of machine learning, History and Goals of ML. Types of machine learning: supervised, unsupervised, reinforcement learning, Applications of machine learning, Data Preprocessing, Dealing with heterogeneous data. Introduction to Python libraries for machine learning (NumPy, pandas, scikit-learn).		
Unit 2	:	**Linear Regression**	5	5
		Simple linear regression, Multiple linear regression, Gradient descent, Regularization techniques: L1 and L2 regularization		
Unit 3	:	**Supervised Learning and Classification**	7	7
		Logistic regression, Decision trees – Entropy, Splitting attribute and Splitting criteria. Support Vector Machines (SVM), k-nearest Neighbors (k-NN) algorithm.		
Unit 4	:	**Model Evaluation and Selection**	4	4
		Training and testing datasets, Cross-validation, Evaluation metrics: accuracy, precision, recall, F1-score, ROC curve, AUC.Concept of Ensemble and Ensemble techniques- Bagging, Boosting, Stacking and Voting.		
Unit 5	:	**Unsupervised Learning**	8	8
		Partitional Clustering - K-means and k-medoids algorithm. Hierarchical clustering – Agglomerative and Divisive. Density based clustering – DBScan algorithm.		
Unit 6		**Dimensionality Reduction**	6	6
		Feature selection, Feature extraction, Principal Component Analysis (PCA).		
Unit 7		**NLP Basic**	3	3
		Working on real-world datasets to show applications of machine learning in Natural Language Processing (NLP), Image processing, Recommender Systems, Healthcare etc.		

Practical / Lab Work	Class	Marks
List of Practical a) Installation and setup of Python and ML libraries b) Data exploration using NumPy and pandas c) Data preprocessing and handling missing values d) Data normalization and encoding techniques e) Simple linear regression implementation f) Multiple linear regression implementation g) Gradient descent implementation h) L1 and L2 regularization techniques i) Logistic regression classification j) Decision tree classifier implementation k) Support Vector Machine (SVM) implementation l) k-Nearest Neighbors (k-NN) classification m) Comparison of classification algorithms n) Model evaluation using accuracy, precision, recall, and F1-score o) Cross-validation and train-test split p) Ensemble learning using bagging and boosting q) ROC curve and AUC analysis r) K-means clustering implementation s) Hierarchical clustering implementation t) DBSCAN clustering implementation	30	

<table>
<tr><th colspan="4">Course Information</th></tr>
<tr><td>Course Title</td><td>:</td><td colspan="2">Database System and Applications</td></tr>
<tr><td>Course Code</td><td>:</td><td colspan="2">CSC-CC-201034</td></tr>
<tr><td>Programme</td><td>:</td><td colspan="2">Master of Science in Computer Science</td></tr>
<tr><td>Semester</td><td>:</td><td colspan="2">1st Semester</td></tr>
<tr><td>Credit</td><td>:</td><td colspan="2">4</td></tr>
<tr><td>Marks</td><td>:</td><td>Component</td><td>Marks</td></tr>
<tr><td></td><td></td><td>Internal Assessment</td><td>30</td></tr>
<tr><td></td><td></td><td>ESE</td><td>45</td></tr>
<tr><td></td><td></td><td>Practical</td><td>25</td></tr>
<tr><td>Course Objective</td><td>:</td><td colspan="2">At the end of the course, students will be able to:
Students will be able to recall key concepts and terminologies related to SQL, object databases, XML, distributed databases, and NoSQL systems.
Students will analyze the effectiveness and efficiency of distributed database architectures, query processing, and transaction management, as well as the advantages and limitations of different NoSQL systems.
Students will evaluate advanced database models and systems, such as active, temporal, spatial, and multimedia databases, and assess data mining techniques for extracting meaningful insights from large datasets.</td></tr>
<tr><td>Course Outcomes</td><td>:</td><td colspan="2">Upon completion of the course, students will be able to:
CO1: Recall and comprehend fundamental and advanced SQL concepts, including data definition, constraints, and complex queries.
CO2: Understand object-oriented, object-relational, and XML database concepts and technologies.
CO3: Analyze and evaluate distributed database systems and NoSQL technologies.
CO4: Create and assess advanced database models and systems, including active, temporal, spatial, multimedia, and deductive databases.</td></tr>
<tr><td>Recommended Book & References</td><td>:</td><td colspan="2">Elmasri Ramez and Navathe Shamkant; Fundamentals of Database System, 7edition, Pearson
R. Ramakrishnan, J. Gehrke, Database Management Systems, McGraw Hill, 2004</td></tr>
</table>

Course Content / Units				
Theory			**Class**	**Marks**
Unit 1	:	**SQL Revisited**	9	9
		SQL Data Definition and Data Types; Specifying Constraints in SQL; Basic Retrieval Queries in SQL; INSERT, DELETE, and UPDATE Statements in SQL; Additional Features of SQL; More Complex SQL Retrieval Queries; Specifying Constraints as Assertions and Actions as Triggers; Views (Virtual Tables) in SQL; Schema Change Statements in SQL		
Unit 2	:	**Object Database Concepts and XML**	12	12
		Overview of Object Database Concepts- Introduction, Object Identity, and Objects versus Literals, Object Identity, and Objects versus Literals, Encapsulation, Type Hierarchies and Inheritance ; Object Database Extensions to SQL; The ODMG Object Model and the Object Definition Language ODL; Object Database Conceptual Design; The Object Query Language OQL; Overview of the C++ Language Binding in the ODMG Standard; Structured, Semi structured, and Unstructured Data; XML Hierarchical (Tree) Data Model; XML Documents, DTD, and XML Schema; Storing and Extracting XML Documents from Databases; XML Languages; Extracting XML Documents from Relational Databases; XML/SQL: SQL Functions for Creating XML Data.		
Unit 3	:	**Distributed Databases**	6	6
		Distributed Database Concepts; Data Fragmentation, Replication, and Allocation Techniques for Distributed Database Design; Overview of Concurrency Control and Recovery in Distributed Databases; Overview of Transaction Management in Distributed Databases; Query Processing and Optimization in Distributed Databases; Types of Distributed Database Systems; Distributed Database Architectures; Distributed Catalog Management;		
Unit 4	:	**NOSQL Databases**	4	4
		Introduction to NOSQL Systems; The CAP Theorem; Document-Based NOSQL Systems and MongoDB; NOSQL Key-Value Stores; Column-Based or Wide Column NOSQL Systems; NOSQL Graph Databases and Neo4j		
Unit 5	:	**Advanced Database Models and Data mining**	12	12
		Active Database Concepts and Triggers; Temporal Database Concepts; Spatial Database Concepts; Multimedia Database Concepts; Introduction to Deductive Databases; Active Database Concepts and Triggers; Temporal Database Concepts; Spatial Database Concepts; Multimedia Database Concepts; Introduction to Deductive Databases; Trends in Information Retrieval; Overview of Data Mining Technology; Association Rules -Market-Basket Model, Support, and Confidence, Apriori Algorithm, ; Classification; Clustering; Applications of Data Mining;		

<table>
<tr><th colspan="4">Course Information</th></tr>
<tr><td>Course Title</td><td>:</td><td colspan="2">Research Methodology</td></tr>
<tr><td>Course Code</td><td>:</td><td colspan="2">CSC-CC-201044</td></tr>
<tr><td>Programme</td><td>:</td><td colspan="2">Master of Science in Computer Science</td></tr>
<tr><td>Semester</td><td>:</td><td colspan="2">1st Semester</td></tr>
<tr><td>Credit</td><td>:</td><td colspan="2">4</td></tr>
<tr><td>Marks</td><td>:</td><td>Component</td><td>Marks</td></tr>
<tr><td></td><td></td><td>Internal Assessment</td><td>40</td></tr>
<tr><td></td><td></td><td>ESE</td><td>60</td></tr>
<tr><td></td><td></td><td>Practical</td><td>Nil</td></tr>
<tr><td>Course Objective</td><td>:</td><td colspan="2">At the end of the course, students will be able to:
To introduce the basic concepts in research methodology in Computer Science.
To familiarize the issues inherent in selecting a research problem
To discuss the techniques and tools to be employed in completing a research project.
Enable the students to prepare report writing and framing Research proposals.</td></tr>
<tr><td>Course Outcomes</td><td>:</td><td colspan="2">At the end of the course, students will be able to:
CO1: Explain the basics in scientific research methodology and its various types
CO2: Apply different data collection and sampling designs, and also to perform statistical analysis of the collected data.
CO3: Develop skills in qualitative and quantitative data analysis and presentation.
CO4: Analyze different research articles/papers for writing critical reviews
CO5: Use ethical approaches in research to obtain maximum benefit and minimum risk to human kind in the research process</td></tr>
<tr><td>Recommended Book & References</td><td>:</td><td colspan="2">C. R. Kothari; (2004); Research Methodology-Methods and Techniques; New-age International Publishers
K. Prathapan; (2019); Research Methodology for Scientific Research; Wiley Publication</td></tr>
</table>

Course Content / Units				
Theory			**Class**	**Marks**
Unit 1	:	**Introduction to Computer Science Research**	5	5
		What is Research? Types of Research, Why Research, Significance & Status of Research in Computer Science.		
Unit 2	:	**Steps in Research**	15	15
		Major Journals & Publications in Computer Science, Major Research Areas of Computer Science, Identification, selection & formulation of research problems. Developing a research proposal, planning your research. How engineering and technological research differs from other scientific research.		
Unit 3	:	**Research Data**	10	10
		Data Collection: Methods of Data Collection, Theory of Sampling, Sampling techniques. Size of a sample.		
Unit 4	:	**Data Analysis and interpretation**	8	8
		Statistical analysis of data: Measures of central tendency, dispersion; Associations/Relations; Regression and Co-relation analysis; Hypothesis testing and tests of significance. Data processing software and statistical inference, Interpretation of results. Statistical Analysis and Data Visualization using a software tool		
Unit 5	:	**Simulation and tools**	8	8
		Concept of Simulation. Time and randomness in simulation, Application of simulations. How a simulation model works, tools for simulations.		
Unit 6		**Literature Survey**	7	7
		Finding out about your research area, Literature search strategy. Writing critical reviews. Identifying venues for publishing your research.		
Unit 7		**Plagiarism**		
		Concept and Importance of understanding what is plagiarism and what is not plagiarism. Methods and ways to detect and avoid plagiarism. Available tools and software for plagiarism check.		
UNIT-8		**Writing Papers and the Review Process**		
		Preparing and presenting your paper. The conference review process. Making use of the referees' reports. The journal review process, Group exercise in reviewing research papers. Tools for paper formatting and Reference Management. Use of LaTeX.		
UNIT-9		**Ethical Issues and Professional Conducts**		
		Ethics in general, Professional Ethics, Ethical Issues that Arise from Computer Technology, General Moral Imperatives, More Specific Professional Responsibilities, Organizational Leadership Imperatives.		

Practical Sessions:

Practical sessions will be held for the following

(i) Statistical Analysis and Data Visualization.

(ii) Explore available tools and software for plagiarism check.

(iii) Use of LaTeX in writing documents with equations, figures etc. and report generation.

Course Information			
Course Title	:	**Advanced Computer Architecture**	
Course Code	:	*CSC-EC-201014*	
Programme	:	Master of Science in Computer Science	
Semester	:	1st Semester	
Credit	:	4	
Marks	:	**Component**	**Marks**
		Internal Assessment	30
		ESE	45
		Practical	25
Course Objective	:	At the end of the course, students will be able to: • Understand the importance of multiprocessor and multicomputer. • Learn about advance architecture designs. • Understand the concepts of interconnected structures. • Learn about data flow computer architectures.	
Course Outcomes	:	At the end of the course, students will be able to: CO1: Explain the working of computer arithmetic. CO2: Demonstrate concepts of parallelism in hardware/software CO3: Apply assembly language to create instruction sets. CO4: Describe architectural features of advanced processors. CO5: Explain data flow in arithmetic algorithms.	
Recommended Book & References	:	• Govindarajalu, B. Computer Architecture and Organization, TMH publication. • Richard Y. Kain, Advanced Computer Architecture A systems Design Approach, PHI Publication • Stallings William, Computer Organization and Architecture Designing for Performance, Pearson Education • Kai Hwang, "Advanced Computer architecture Parallelism, scalability, Programmability", McGraw-Hill, N.Y, 2003 • Kai Hwang and Faye Briggs, "Computer Architecture and Parallel Processing", McGrawHill International Edition, 2000	

Course Content / Units				
Theory			**Class**	**Marks**
Unit 1	:	**Computer Arithmetic**	10	10
		Serial adder, Parallel adder, Ripple carry adder, Carry look-ahead adder, Multiplication of signed and unsigned numbers, Booth's algorithm, Division of integer, Floating point arithmetic.		
Unit 2	:	**Advanced Architectures**	15	15
		Classification: SISD, SIMD, MISD, MIMD, Bus data path:one, two and three bus, Scalar, vector, superscalar and pipelined processor, VLIW architecture, EPIC architecture, Pipelining: Instruction pipeline, pipeline bubbles, Hazards: resource conflicts, data dependency, branch difficulty, Vector computing: arithmetic pipeline, vector and scalar register, chaining, scatter gather operations, vector-register processor, Memory vector processor, Array processor.		
Unit 3	:	**Assembly Language**	10	10
		Machine Language, Register transfer language, Assembly Language, Assembler, Program Loops, Subroutines, Developing counters and Time delay routines, Input-Output Programming. Interfacing concepts.		
Unit 4	:	**8085 Microprocessor Architecture and Memory Interface**	8	8
		Pin Description, Operating Modes, Instruction Set of 8085: Instructionset, Data formats, Addressing modes, Opcode & Operands, Data and Storage, Word size, 8085 Instructions: Counter and Time delays, Stack, Subroutines, Call &Return statements, Interrupts in 8085: generation of RST codes, interrupt priority, SIM & RIM instructions.		
Unit 5	:	**Multiprocessor**	8	8
		Characteristics of Multiprocessors, Interconnection Structures, Interprocessor Arbitration, Interprocessor Communication and Synchronization, Cache Coherence, Multicore Processors		
Unit 6		**Dataflow Computers and VLSI Computations**	7	7
		Data flow architecture: static and dynamic, VLSI Computing Structures, Array Architectures, Mapping algorithms, Reconfigurable processor array, VLSI matrix arithmetic models and processors, Matrix Algorithms and Pipelines		

Course Information		
Course Title	:	**Advanced Operating System**
Course Code	:	**CSC-CC-202014**
Programme	:	Master of Science in Computer Science
Semester	:	2nd Semester
Credit	:	4
Marks	:	**Component** / **Marks** Internal Assessment / 30 ESE / 45 Practical / 25
Course Objective	:	After successful completion of this course, students will be able to: • Understand advanced operating system concepts and architectures. • Explore virtualization technologies and their applications. • Study real-time operating systems and their characteristics.
Course Outcomes	:	At the end of the course, students will be able to: **CO1** Identify different operating advanced systems and its types, functions, applications. **CO2** Explain process, threads and solve process scheduling problems of various advanced operating systems. **CO3** Describe hardware requirements, load balancing of advanced operating systems. **CO4** Compile case study reports of various real world advanced operating systems. **CO5** Compare advanced operating systems and utilize them according to their use cases.
Recommended Book & References	:	**1.** Modern Operating Systems, Andrew S. Tanenbaum, 4th edition, Pearson Education **2.** Operating System Concepts, Abraham Silberschatz, Peter B. Galvin, Greg Gagne,Wiley **3.** Distributed Systems Principles and Paradigms Tanenbaum, Steen 2nd edition, Pearson Education **4.** Real-Time Systems, Jane W. S. Liu, 1st edition, Pearson Education India

Course Content / Units				
Theory			**Class**	**Marks**
Unit 1	:	**Review of Operating system concepts**	2	2
		Process and threads: Process Control Block (PCB), process scheduling, Process synchronization: critical section problem, semaphore, process deadlock.		
Unit 2	:	**Multiprocessor systems**	12	12
		Introduction to multiprocessor systems, multiprocessor hardware: UMA (Uniform Memory Access), NUMA (Non Uniform Memory Access), Multicore Chips, Moore's law, Manycore chips. Multiprocessor Operating System Types: private OS, Master-Slave Multiprocessors, Symmetric Multiprocessors, Multiprocessor Synchronization: TSL (Test and Set Lock), Peterson's protocol, Multiprocessor scheduling: Space Sharing, Gang Scheduling.		
Unit 3	:	**Multicomputer systems**	13	13
		Introduction to multicomputer systems, multicomputer Hardware organization: interconnection Technology, Network Interfaces, Low-Level Communication Software, User-Level Communication Software: Send and Receive, Blocking versus Nonblocking Calls, Remote Procedure Call : client stub, server stub, marshalling, Load Balancing, processor allocation algorithms: Graph-Theoretic Deterministic Algorithm, Sender-Initiated Distributed Heuristic Algorithm, Receiver-Initiated Distributed Heuristic Algorithm.		
Unit 4	:	**Distributed system**	10	10
		Definition of a distributed system. Characteristics of distributed systems, distributed systems vs centralized systems. Examples of Distributed Systems in Real-world Applications, System models: Fundamental and Architectural model, System architectures- client-server architectures, Synchronization: Needs of clock synchronization, external and internal clock synchronization, Logical and vector clocks, Lamport's logical clock, Vector clocks, Causal Order of messages. Global state, Chandy Lamport snapshot algorithm		
Unit 5	:	**Virtualization and Containerization**	10	10
		Definition of virtualization, need of virtualization, examples of virtual machines, advantage and disadvantage of virtualization, hypervisor architecture : type 1 and type 2 hypervisor, guest OS, host OS, Introduction to Containerization Technologies (e.g., Docker, Kubernetes), Containers vs. Virtual Machines, benefits of containerization, use cases of containerization, container orchestration		
Unit 6		**Real-time operating systems**	10	10
		Definition and Characteristics of Real-Time Systems(RTOS), Differences Between General Purpose Operating Systems and RTOS, Importance of Timing Constraints in RTOS, Types of RTOS, Hard vs Soft RTOS, Classification of Real-Time Tasks: Periodic, Aperiodic, Sporadic tasks, Real-Time Scheduling Algorithms: Earliest Deadline First, Least Laxity First.		
Unit 7		**Case study on real-time operating systems**	3	3
		FreeRTOS, FreeRTOS features, Applications of FreeRTOS, FreeRTOS architecture, FreeRTOS supported architectures, FreeRTOS libraries.		

Course Information			
Course Title	:	**Complier Design**	
Course Code	:	**CSC-CC-202024**	
Programme	:	Master of Science in Computer Science	
Semester	:	2nd Semester	
Credit	:	4	
Marks	:	**Component**	**Marks**
		Internal Assessment	30
		ESE	45
		Practical	25
Course Objective	:	After successful completion of this course, students will be able to: • Use compiler construction tools and describes the Functionality of each stage of compilation process • Construct Grammars for Natural Languages and find the Syntactical Errors/Semantic errors during the compilations using parsing techniques • Analyze different representations of intermediate code. • Construct new compiler for new languages.	
Course Outcomes	:	At the end of the course, students will be able to: **CO1** Describe about compilers, its phases and working of compilers. **CO2** Computer Regular Expressions, NFA, DFA to recognized tokens. **CO3** Design different parsers such as Top-down parser, Bottom-up parsers like LR, SLR and LALR **CO4** Explain intermediate representation i.e. generation of three address codes, representation of it using DAG and devise code optimization solution for a given problem. **CO5** Develop Lexical analyzer using Lex and parser generator using Yacc.	
Recommended Book & References	:	1. Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman (2007), Compilers: Principles, Techniques and Tools, 2nd edition, Pearson Education, New Delhi, India. 2. Alfred V. Aho, Jeffrey D. Ullman (2001), Principles of compiler design, Indian student edition, Pearson Education, New Delhi, India. 3. Kenneth C. Louden (1997), Compiler Construction– Principles and Practice, 1st edition, PWS Publishing. 4. K. L. P Mishra, N. Chandrashekaran (2003), Theory of computer science-Automata Languages and computation, 2nd edition, Prentice Hall of India, New Delhi, India. 5. Andrew W. Appel (2004), Modern Compiler Implementation C, Cambridge University Press, UK.	

Course Content / Units				
Theory			**Class**	**Marks**
Unit 1	:	**Introduction to Compiler**	12	12
		Definition of compiler, Phases of a compiler, Lexical analysis, Role of lexical analyzer, Tokens, Patterns, Lexemes, Input buffering, Specification of tokens-strings and languages, operations on languages, regular expressions, regular definitions, Recognition of tokens, Lexical analyzer generator- Lex, Finite automata, From Regular expressions to automata.		
Unit 2	:	**Syntax Analysis**	16	16
		Parsing, Role of parser, Context free grammar, Parse tree and derivations, Ambiguity, Eliminating ambiguity from dangling-else grammar, Elimination of left recursion, Left factoring, Top down Parsing- Recursive descent parser, Predictive parser- LL (1) Grammar, construction of predictive parsing table. Bottom Up Parsing- Reductions, Handle pruning, Shift-Reduce parsing, Conflicts during shift reduce parsing, LR Parser-Items, Kernel items, Non-kernel items, closure of Item Sets, The function GOTO, LR (0) automaton, Construction of SLR parsing table, Basics of LALR parser, Automatic parser generator-YACC.		
Unit 3	:	**Syntax Directed Translation**	12	12
		Syntax directed definition- inherited and synthesized attributes, evaluating an SDD at the nodes of a parse tree, Evaluation orders of SDD's- dependency graphs, ordering the evaluation of attributes, S-attributed and L-attributed definitions, Applications of syntax-directed translation- construction of syntax trees, the structure of a Type, Syntax directed translation schemes- postfix translation schemes, SDT's with actions inside productions, eliminating left recursion from SDT's, Variants of syntax trees- directed acyclic graphs (DAG) for expressions, The value number method for constructing DAG's, Three address code- Quadruples, Triples and Indirect triples, Static single-assignment form, Types and Declarations, Translation of expressions, Type Checking, Basics of Control flow, Basics of Back patching.		
Unit 4	:	**Run Time Environments**	10	10
		Storage organization, Stack allocation of space, Access to non-local data on Stack, Basics of Heap management, Basics of garbage collection		
Unit 5	:	**Code Generation and optimization**	10	10
		Machine dependent code generation, Issues in design of code generator, The target language, Addresses in the target code, Basic blocks and flow graphs, Optimization of basic blocks- the DAG representation of Basic blocks, Finding local common sub-expression, dead code elimination, A simple code generator, Basics of Peephole optimization, The Principal Sources of Optimization, Introduction to Data-Flow Analysis.		

<table>
<tr><th colspan="4">Course Information</th></tr>
<tr><td>Course Title</td><td>:</td><td colspan="2">Cryptography and Network Security</td></tr>
<tr><td>Course Code</td><td>:</td><td colspan="2">CSC-CC-203024</td></tr>
<tr><td>Programme</td><td>:</td><td colspan="2">Master of Science in Computer Science</td></tr>
<tr><td>Semester</td><td>:</td><td colspan="2">3rd Semester</td></tr>
<tr><td>Credit</td><td>:</td><td colspan="2">4</td></tr>
<tr><td rowspan="4">Marks</td><td rowspan="4">:</td><td>Component</td><td>Marks</td></tr>
<tr><td>Internal Assessment</td><td>30</td></tr>
<tr><td>ESE</td><td>45</td></tr>
<tr><td>Practical</td><td>25</td></tr>
<tr><td>Course Objective</td><td>:</td><td colspan="2">After successful completion of this course, students will be able to:
Demonstrate a comprehensive understanding of fundamental concepts, principles, and technologies related to network security, including encryption techniques, authentication protocols, and security architectures.
Apply their knowledge and skills to analyze, design, and implement effective security measures to protect networked systems and data from unauthorized access, malicious attacks, and other security threats.</td></tr>
<tr><td>Course Outcomes</td><td>:</td><td colspan="2">At the end of this course, students will be able to:
CO1 Students will recall and summarize fundamental concepts of network security, including encryption techniques, authentication protocols, and security threats.
CO2 Students will demonstrate an understanding of various encryption algorithms and protocols used in network security, explaining their purposes, mechanisms, and limitations.
CO3 Students will comprehend the importance of key management, digital signatures, and authentication mechanisms in ensuring the confidentiality, integrity, and availability of networked systems.
CO4 Students will comprehend the importance of key management, digital signatures, and authentication mechanisms in ensuring the confidentiality, integrity, and availability of networked systems.
CO5 Students will evaluate the strengths and weaknesses of different approaches to network security, assessing their suitability for addressing specific security requirements and mitigating emerging threats.</td></tr>
<tr><td>Recommended Book & References</td><td>:</td><td colspan="2">W. Stallings, Cryptography & Network Security Principles & Practices (7th Edition), Pearson Education, 2017.68
Wade Trappe, Lawrence C Washington, “Introduction to Cryptography with coding theory”, Pearson.
D. R. Stinson, Cryptography: Theory & Practice (3rdEdition), CRC Press, 2006.</td></tr>
</table>

Course Content / Units				
Theory			**Class**	**Marks**
Unit 1	:	**INTRODUCTION**	8	8
		OSI Security Architecture -Security Attacks-Security Services-A Model for Network Security- Classical Encryption techniques – Cipher Principles – Symmetric Cipher Model; Substitution Techniques; Transposition Techniques; Steganography		
Unit 2	:	**Modern Block Ciphers**	7	7
		Data Encryption Standard – Block Cipher Design Principles and Modes of Operation - Evaluation criteria for AES – AES Cipher – Triple DES – Placement of Encryption Function – Traffic Confidentiality		
Unit 3	:	**PUBLIC KEY CRYPTOGRAPHY AND HASH FUNCTIONS**	15	15
		Public Key Cryptosystems, Applications, Requirements, Cryptanalysis, RSA Algorithm,Key Management - Diffie-Hellman key Exchange – Cryptographic Hash Functions: Applications of Cryptographic Hash Functions, Simple Hash Functions, Authentication requirements – Authentication functions – Message Authentication Codes-Message Authentication Requirements, Authentication Functions, Requirements of Message authentication codes, Security of MACs.		
Unit 4	:	**NETWORK SECURITY**	15	15
		Digital Signatures; Key Management and Distribution;Authentication Applications: Kerberos – X.509 Authentication Service – Electronic Mail Security – PGP – S/MIME - IP Security – Web Security; Intrusion detection – password management – Viruses and related Threats – Virus Counter measures – Firewall Design Principles – Trusted Systems.		

Course Information			
Course Title	:	**Theory of Computation**	
Course Code	:	**CSC-CC-203034**	
Programme	:	Master of Science in Computer Science	
Semester	:	3rd Semester	
Credit	:	4	
Marks	:	**Component**	**Marks**
		Internal Assessment	30
		ESE	45
		Practical	25
Course Objective	:	After completing this course, students • Understand the Mathematical model of a finite state machine. Know deterministic and non-deterministic versions of Finite automata. • Grasp the mathematical concepts of languages and grammar. • Know Pushdown Automata and the associated grammar/language. • Know the properties of Regular languages and Context free languages.	
Course Outcomes	:	At the end of this course, students will be able to: **CO1** Define the concept of finite automata and language accepted by it. **CO2** Discuss regular grammar and the properties of regular languages. **CO3** Explain the concept of PDA and language accepted by it. **CO4** Analyze the simplification of context free language and Normal forms of context free language. **CO5** Formulate the concept of pumping lemma for regular language and context free language	
Recommended Book & References	:	1. An introduction to Formal Languages and Automata, Peter Linz,Narosa. 2. Introduction to Automata Theory, Languages and Computation, Hopcroft, Motwani and Ullman, Pearson. 3. Theory of Computer Science (Automata, Languages and Computation),K. L. P. Mishra, N. Chandrasekaran; P. H.I.	

Course Content / Units				
Theory			**Class**	**Marks**
Unit 1	:	**Finite Automata**	10	10
		DFA, NFA, NFA with empty-moves, Equivalence of DFA and NFA, Reduction of the number of states in finite automata.		
Unit 2	:	**Regular Languages and Regular Grammar**	12	12
		Concept of languages and grammar, Regular expressions, Connection between regular expressions and regular languages, Regular grammars, Right and Left-Linear Grammars, Equivalence between Regular languages and Regular grammars.		
Unit 3	:	**Properties of Regular Languages**	13	13
		Closure under simple set operations- union, intersection, concatenation, complementation and star closure, Decision algorithms for emptiness, finiteness and infiniteness, equality, Proof of non-regularity using Pigeonhole principle and using pumping lemma for regular languages.		
Unit 4	:	**Context Free languages**	15	15
		Context-free grammars, leftmost and rightmost derivations, derivation trees, parsing and Ambiguity in grammars and languages, Simplification of Context free Grammars- removing useless productions, empty-productions and unit-productions. Normal forms- Chomsky and Greibach normal forms, Pumping Lemma for CFL, Using Pumping Lemma to show that certain languages are not Context free, properties and decision algorithms of CFL.		
Unit 5		**Pushdown Automata**	15	15
		Definition and language accepted (acceptance by empty stack and final state and their equivalence), Pushdown Automata and Context free languages. Deterministic PDA and Deterministic Context free Languages.		