ACADEMIC SCHEME

FOR

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

INCLUDING

CURRICULUM STRUCTURE

AND

DETAILED SYLLABUS

FOR

SEMESTERS III TO VIII

BATCH 2023 AND ONWARDS

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ISLAMIC UNIVERSITY OF SCIENCE AND TECHNOLOGY, KASHMIR

SEMESTER-WISE CREDIT DISTRIBUTION

S. No.	Semester	Total Credits
01	I	18
02	II	20
03	III	19
04	IV	18
05	V	19
06	VI	21
07	VII	19
08	VIII	18
09	Open Electives	08
10	MOOCs	01
Total:		161

NOTE

1. **Credit Requirements for BTech CSE**:
   a. To fulfill the programme requirements for the *Bachelor of Technology in Computer Science and Engineering* at the Islamic University of Science and Technology Kashmir, students must earn a total of at least ***160*** credits.
   b. This must include ***8*** credits from *Open Elective Courses* (starting from the 3rd semester) and ***1*** credit from *Massive Open Online Courses (MOOCs)*.

2. **Eligibility for BTech CSE with Honors**:
   Students can obtain a *BTech CSE with Honors* ***either*** by completing an additional ***18*** credits by taking Honors Degree courses offered by the department of Computer Science and Engineering ***or*** by earning an additional ***20*** credits through MOOCs.

3. **Eligibility for BTech CSE with Minors**:
   To earn a *BTech CSE with Minors*, the students of the BTech CSE programme must complete an additional ***18*** credits by taking Minor Degree courses offered by other engineering departments of the University.

4. **Eligibility for BTech ECE/EE/CE/ME/FT with Minors in AI & ML:**
   To earn a *BTech ECE/EE/CE/ME/FT with Minors in Artificial Intelligence and Machine Learning*, the students of the concerned department must complete an additional ***18*** credits by taking Minor Degree courses offered by the University's department of Computer Science and Engineering.

5. **MOOCs Credits**:
   Students can earn the required MOOCs credits at any point during the programme. However, prior approval from the Head of the Department is mandatory before taking a MOOCs.

Other conditions may also apply in accordance with the University rules.

CURRICULUM STRUCTURE

Semester III

S.No.	Course Code	Course Title	L	P	Credits
1	CSE210C	Data Structures	3	0	3
2	CSE211C	Object Oriented Programming	3	0	3
3	CSE212C	Database Management System	3	0	3
4	CSE213C	Discrete Mathematics	3	0	3
5	CSE214C	Digital Electronics and Logic Design	3	0	3
6	CSE215C	Data Structures Lab	0	2	1
7	CSE216C	Object Oriented Programming Lab	0	2	1
8	CSE217C	Database Management System Lab	0	2	1
9	CE218C	Digital Electronics and Logic Design Lab	0	2	1
10	CSE210A	Seminar	0	0	-
11		Minors / Honors Degree Course			3
Total:			15	8	19/22

Semester IV

S.No.	Course Code	Course Title	L	P	Credits
1	CSE260C	Design and Analysis of Algorithms	3	0	3
2	CSE261C	Computer Architecture and Organization	3	0	3
3	CSE262C	Web Technologies	3	0	3
4	CSE263C	Software Engineering	3	0	3
5	CSE264C	Data Communication	3	0	3
6	CSE265C	Design and Analysis of Algorithms Lab	0	2	1
7	CSE266C	Web Technologies Lab	0	2	1
8	CSE267C	Project-I	0	2	1
9		Minors / Honors Degree Course			3
Total:			15	6	18/21

Semester V

S.No.	Course Code	Course Title	L	P	Credits
1	CSE310C	Operating System	3	0	3
2	CSE311C	Computer Networks	3	0	3
3	CSE312C	Python Programming	3	0	3
4	CSE313C	Microprocessors, Peripherals and Interfacing	3	0	3
5	CSE314C	Probability and Statistics	3	0	3
6	CSE315C	Operating System Lab	0	2	1
7	CSE316C	Computer Networks Lab	0	2	1
8	CSE317C	Python Programming Lab	0	2	1
9	CSE318C	Microprocessors, Peripherals and Interfacing Lab	0	2	1
10		Minors / Honors Degree Course			3
Total:			15	8	19/22

Semester VI

S.No.	Course Code	Course Title	L	P	Credits
1	CSE360C	Theory of Computation	3	0	3
2	CSE361C	Artificial Intelligence and Machine Learning	3	0	3
3	CSE362C	Embedded Systems and Internet of Things	3	0	3
4	CSE363C	Java Programming	3	0	3
5		Elective–I	3	0	3
6	CSE364C	Artificial Intelligence and Machine Learning Lab	0	2	1
7	CSE365C	Embedded Systems and Internet of Things Lab	0	2	1
8	CSE366C	Java Programming Lab	0	2	1
9	CSE360A	Aptitude and Reasoning	2	0	-
10	CSE367C	Project-II	0	6	3
11		Minors / Honors Degree Course			3
Total:			17	12	21/24

Semester VII

S.No.	Course Code	Course Title	L	P	Credits
1	CSE410C	Compiler Design	3	0	3
2	CSE411C	Network Security	3	0	3
3	CSE412C	Industrial Management and Entrepreneurship Development	3	0	3
4		Elective–II	3	0	3
5		Elective–III	3	0	3
6	CSE413C	Project-III	0	8	4
7		Minors / Honors Degree Course			3
Total:			**15**	**8**	**19/22**

Semester VIII

S.No.	Course Code	Course Title	L	P	Credits
1		Elective–IV/ MOOCs	3	0	3
2		Elective–V / MOOCs	3	0	3
3	CSE460C	Internship / Industrial Training	0	0	2
4	CSE461C	Project-IV	0	20	10
5		Minors / Honors Degree Course			3
Total:			**6**	**20**	**18/21**

LIST OF DISCIPLINE CENTRIC ELECTIVES

S.No.	Course Code	Course Title	L	P	Credits	Type	Prerequisite
1	CSE001E	Advanced Computer Architecture	3	0	3	Employability	Computer Architecture and Organization
2	CSE002E	Advanced Java	2	2	3	Skill	Java
3	CSE003E	Advanced JavaScript	2	2	3	Skill	Core JavaScript
4	CSE004E	Agile Software Development	3	0	3	Employability	Software Engineering
5	CSE005E	Big Data	3	2	4	Employability	Database Management System
6	CSE006E	Computational Ethics Design	3	0	3		
7	CSE007E	Computer Graphics	3	2	4		Computer Programming
8	CSE008E	Computer Vision	3	0	3	Employability	Data Structures
9	CSE009E	Data Visualization	2	2	3	Employability	
10	CSE010E	Deep Learning	3	2	4	Employability	
11	CSE011E	Distributed Computing	3	0	3		Operating System
12	CSE012E	Evolutionary Computing	3	0	3		Computer Programming
13	CSE013E	Modeling and Simulation	3	0	3	Employability	Computer Programming
14	CSE014E	Multimedia Technologies	3	0	3		
15	CSE015E	Natural Language Processing	3	2	4		Computer Programming
16	CSE016E	Open Source Technologies	3	0	3		
17	CSE017E	Optimization Techniques	3	0	3		
18	CSE018E	Reinforcement Learning	3	2	4		Machine Learning
19	CSE019E	Ruby on Rails	2	2	3	Employability	
20	CSE020E	Soft Computing	3	0	3		Computer Programming
21	CSE021E	Software Project Management	3	0	3	Skill	Software Engineering
22	CSE022E	Software Testing and Quality Assurance	3	0	3	Employability	Software Engineering
23	CSE023E	System Software	3	0	3		

LIST OF GENERIC ELECTIVES

S.No.	Course Code	Course Title	L	P	Credits	Type	Prerequisite
1	CSE001G	Applied ML for Embedded IoT devices	3	2	4		
2	CSE002G	Bioinformatics	3	0	3		Computer Programming
3	CSE003G	Blockchain	3	0	3	Employability	Computer Network/ Network Security
4	CSE004G	C# and .Net Programming	2	2	3	Skill	Computer Programming
5	CSE005G	Cloud Computing	3	0	3	Employability	Computer Networks
6	CSE006G	Cyber Physical Systems	3	0	3		
7	CSE007G	Data Mining	3	2	4	Employability	
8	CSE008G	Digital Image Processing	3	0	3		Computer Programming
9	CSE009G	Ethical Hacking	3	2	4	Skill	Computer Network
10	CSE010G	Go Language	2	2	3	Employability	
11	CSE011G	High Performance Computing	3	0	3		Computer Architecture and Organization
12	CSE012G	Programming in R	2	2	3	Skill	
13	CSE013G	Robotics	3	2	4	Skill	Embedded Systems
14	CSE014G	Simulation Technologies	2	2	3	Skill	
15	CSE015G	Wireless Communication	3	0	3		Computer Networks/ Data Communication

DETAILED SYLLABUS

SEMESTER III

Course Code: **CSE210C**	**Data Structures**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Define, understand, describe and implement linear data structures like stack, queues, linked lists and nonlinear data structures like trees and graphs.
- Design and trace the algorithms for various operations on different data structures studied.
- Understand and implement various searching and sorting techniques
- Write programs in C to simulate operations and applications of data structures learnt

Unit – I **(9)**

Introduction to data structures, classification of Data Structures, Primitive vs. Non-Primitive data structures, Linear vs Non-Linear data structures, Primitive Data Structures Operations, Recursion Function and its Examples. String Representation and operations, Arrays representation, implementation and limitations**.**

Unit – II **(10)**

Linked List: Linked List and its comparison with array implementation, Singly, Doubly and Circular linked list, their Implementation and Comparison.
Stacks: Static and Dynamic Implementation. Applications of Stacks. Prefix Postfix and Infix Expressions. Infix to postfix conversion.
Queues: Static and Dynamic Implementation. Applications of Queues, Types of Queues, Array Implementation of Circular Queues.

Unit – III **(9)**

Searching: Sequential and Binary Search on Array-based Ordered Lists, Binary Trees, their Implementation and Traversal, Binary Search Trees: Searching, Insertion and Deletion of Nodes, Height Balanced Trees and Concept of AVL Trees, Concept and purpose of B Trees and B+ Trees.

Unit – IV **(9)**

Graphs and their Representations, Graph Traversal Techniques: Breadth First Search (BFS) and Depth First Search (DFS), Applications of BFS and DFS, Minimum Spanning Trees (MST), Prim's and Kruskal's algorithms for MST, Connected Components, Dijkstra's Algorithm for Single Source Shortest Paths.

Unit – V **(8)**

Sorting Techniques: Insertion Sort, Selection Sort, Merge Sort, Quick Sort, Heap Sort, Shell Sort, Radix Sort, Bucket sort. Concept of Hash Functions, Hash-tables and Hashing with Chaining. File Structure: Sequential Files, Indexed Files, Direct Files.

Textbooks:

1. Shaum's outlines "*Data Structures with C*" Seymour Lipschutz, Tata McGraw Hill Education.
2. Langsam Augenstein Tenenbaum "*Data Structures using C and C++"*

Reference Books:

1. Data Structures and Algorithms by Narsimha Kamarachi.
2. Tremblay and Sorenson, "*An Introduction to Data Structures with Applications*",
3. McGraw hill, Kongakusha.
4. Horowitz Sahni Mehta, "*Fundamentals of Data structures*", SBCS Publication
5. Data Structures by Rajni Jindal.

Online Resources:

1. https://nptel.ac.in/courses/106102064

Course Code: **CSE211C**	**Object Oriented Programming**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To understand the principles of object-oriented programming with C++,
- To identify and practice the object-oriented programming concepts and techniques,
- To solve real-world problems through object-oriented approach,
- To understand functions, classes, data and objects in C++,
- To understand the concepts of operator overloading, virtual functions, polymorphism and inheritance with the understanding of early and late binding, and
- Understanding advanced features of C++ specifically strings and templates.

Unit – I **(7)**

Introduction: Basic features and concepts of Object-Oriented Programming (OOP), Benefits, Languages and Applications of OOPs.
Tokens, Expressions and Control Structures: Tokens, Keywords, Identifiers and Constants, Basic Data types, User-defined Data types, Derived Data Types, Memory Management Operators, Manipulators, Expressions, Operator Overloading, Control Structures.

Unit – II **(10)**

Classes and Objects: Specifying a class, defining member functions, private member functions, array within a class, memory allocation for objects, arrays of objects, objects as function arguments, returning objects, pointers to members, local classes, Nested Class, this pointer.
Constructors and Destructors: Constructors, Parameterized Constructors, Constructors with Default arguments, Dynamic Initialization of objects, Dynamic Constructors and Destructors, Recursive Constructor, Constructor and Destructor with Static Members

Unit – III **(10)**

Functions in C++: Main function, function prototyping, call by reference, Returning more values by Reference, inline functions, default functions, function overloading. Static data members, static function members.
Overloading and Type Conversion: Definition and Rules of overloading Operators, Overloading Binary and Unary Operators. Overloading operators using friends

Unit – IV **(9)**

Polymorphism and Inheritance: Polymorphism, Compile time polymorphism – function overloading and operator overloading. Pointers, Pointers to Objects and derived classes. Run time polymorphism – function overriding.
Inheritance – Types of inheritance, constructors and destructors in inheritance. Constraints of multiple inheritance. Abstract base class – virtual and pure virtual functions.

Unit – V **(10)**

Exception handling: Introduction, Principles of Exception Handling, The Keywords Try, Throw and Catch, Exception Handling Mechanism, Multiple Catch Statements, Catching Multiple Exceptions, Re-throwing Exception, Specifying Exception, Exceptions in Constructor and Destructors, Controlling Uncaught Exceptions.
Templates: Introduction, Function templates, Class templates, STL – Container, Algorithm, Iterator – vector, list, stack, map.

Textbooks:

1. Object-Oriented Programming in C++, Robert Lafore, Pearson Education India.
2. C++ Programming Language, Bjarne Stroustrup, Addison-Wesley.

Reference Books:

1. Effective Modern C++, Scott Meyers, Shroff/O'Reilly.
2. C++ Primer, Stanley Lippman, Addison-Wesley.
3. Object-Oriented Programming with C++, E Balagurusamy, Tata McGraw Hill.
4. The Complete reference – C++, Herbert Shield

Course Code: **CSE212C**	**Database Management System**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Identify the basic concepts and various data model used in Database design.
- Apply relational database theory and be able to describe relational algebra expression, tuple and domain relation expression for queries and SQL for implementing the queries.
- Recognize and identify the use of normalization and functional dependency.
- Apply and relate the concept of transaction, concurrency control in database.
- Apply recovery, indexing and hashing technique in database.

Unit – I **(6)**

Introduction: Introduction to database management, Characteristics of the Database, Database Systems, Data Models, data abstraction, and system structure, Purpose of database system, uses of database approach, database applications, Views of data, Database languages, Database system – Concepts and architecture, Database users and administrator, database types.

Unit – II **(10)**

Data models: definition and types, Entity-Relationship Model (E-R Model), E-R diagrams, entity set, relationship sets, mapping, cardinalities. Extended ER model. Translating ER Model into Relational Model. Introduction to relational databases, The relational model -Keys. Relational algebra, Tuple relational calculus.

Unit – III **(10)**

Database design: Relational database design, Functional dependencies, Non-loss decomposition, First, Second, Third Normal Forms, Dependency Preservation, Boyce/Codd Normal Form, Multi-Valued Dependencies and higher normal Forms.

Unit – IV **(10)**

Transactions: Transaction Concepts, ACID Properties, System Recovery, Concurrency, need for concurrency, locking protocols, Timestamp based Protocol, Deadlock, Serializability.

Unit – V **(9)**

Implementation techniques: Failures and their classification, recovery and atomicity, recovery algorithms, File organization, indexing (e.g., B and B+ trees).

Textbooks:

1. R. and Navathe, S.B., "Fundamentals of Database Systems", Pearson Education.

Reference Book:

1. Database system Concept by Silberschatz and Korth.
2. Abraham, H. and Sudershan, S., "Database System Concepts", McGraw-Hill.Elmasri.
3. Ramakrishnan, R. and Gekhre, J., "Database Management Systems", Tata McGraw-Hill.

Course Code: **CSE213C**	**Discrete Mathematics**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To gain an understanding of the fundamental concepts in discrete mathematics.
- Understand the notion of mathematical thinking, mathematical proofs, and algorithmic thinking
- To become familiar with various areas of discrete mathematics, including set theory, combinatorics, graph theory, discrete structures, and advanced topics.
- To develop the skills to apply discrete mathematics to solve problems in computer science and related fields.
- To prepare for interdisciplinary research that combines mathematics, computer science, and other disciplines to solve problems and advance the field of discrete mathematics.

Unit – I **(9)**

Propositional Logic, Applications of Propositional Logic, Propositional Equivalences, Predicates and Quantifiers, First order logic, Nested Quantifiers, Rules of Inference

Unit – II **(9)**

Sets, Venn Diagrams, power set, Cartesian product, Set Notation, Set operations, Set Identities, Computer representation of sets, Boolean functions, Identities of Boolean Algebra, Duality, Abstract Definition of a Boolean Algebra, Sum of Products and Product of sums Expansion, Functional Completeness, NAND and NOR implementation.

Unit – III **(9)**

Functions, relations, reflexive, symmetric, antisymmetric, transitive, composition, Representing relations, Equivalence relations, partial orders and lattices, Monoids, Groups

Unit – IV **(9)**

Graph Terminology, Graphs: connectivity, matching, coloring, Handshaking Lemma, Konigsberg seven bridge problem, Euler graphs, Euler's theorem, Hamiltonian path and circuits, Graph coloring, chromatic number, isomorphism and Homomorphism of graphs, Trees terminology, properties of trees, Application of graphs and trees.

Unit – V **(9)**

Counting, recurrence relations, generating functions, Sum and product rule, Principle of Inclusion Exclusion. Pigeonhole Principle

Textbooks:

1. C. L. Liu: Elements of Discrete Mathematics, Tata Mc-Graw Hill.
2. Kolman, Busby and Ross: Discrete Mathematical Structures,.PHI
3. Narsingh Deo: Graph Theory with Applications to Engineering and Computer Sciences, PHI.
4. Murry R. Spiegel: Discrete Mathematics (Schaums Outline series), Tata McGraw Hill

Reference Books:

1. Kenneth H. Rosen: Discrete Mathematics and its applications,5th Ed. Tata McGrawHill
2. K.R Parthasarty: Basic Graph Theory, Tata Mc-Graw Hill

Course Code: **CSE214C**	**Digital Electronics and Logic Design**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To introduce the basic concepts of digital systems and the use of Boolean algebra in logic analysis and design.
- Understand the principles and methodology of digital logic design at the gate and switch level, including both combinational and sequential logic elements.
- To introduce basic tools of logic design and provide hands-on experience designing digital circuits and components through simple logic circuits to hardware description language
- Apply Boolean algebra and other techniques to express and simplify logic expressions.
- Analyze and design combinational and sequential digital systems.
- Use different techniques, among them a hardware description language and a programming language, to design digital systems.

Unit – I **(8)**

Introduction

Number Systems - Decimal, Binary, Hexadecimal, Octal Number systems and their Conversions, Arithmetic operations, subtraction using 1's and 2's complement, Binary coded decimal, Excess-3 Codes, Gray Codes, Binary weighted code.
Introduction to Boolean algebra and Boolean theorems, Minimization of Boolean Expressions by using theorems, Different types of Logic Gates and implementation of logic circuits using logic gates, Binary Arithmetic.

Unit – II **(10)**

Simplification of Boolean Expressions / Logic Gates

Introduction to minterms, maxterms, sum of product and product of sum representation of Boolean function, Simplification techniques and minimization by K-map method and Tabular (Q-M) method, NAND and NOR implementation.

Unit – III **(10)**

Combinational Logic Circuits

Design of various combinational circuits: Half/Full Adder and Subtractor, Ripple carry adder, Carry look ahead adder, Binary Adder/Subtractor, BCD adder, Binary Multiplier, Magnitude comparator, Multiplexers, De-Multiplexers, Decoders, Encoders.

Unit – IV **(9)**

Sequential Logic Circuits and Digital IC Families

Introduction to latches and flip flops, truth table and excitation table of flip flops, conversions of flip flops, design of synchronous and asynchronous counters, design of various types of shift registers.
Digital IC families: DTL, TTL, ECL, MOS, CMOS and their interfacing.

Unit – V **(10)**

Digital Circuit Design and Semiconductor Memories

Introduction to Moore and Mealy systems, state diagrams and tables, state reduction, design and analysis of Moore and Mealy systems.
Semiconductors Memories like ROM and RAM, Introduction to programmable logic design: PLA, PAL, ADC and DAC.

Textbooks:

1. Digital Logic and Computer Design, M. Morris Mano, Pearson Education India.
2. Digital Systems: Principles and Applications, Ronald J. Tocci, Neal Widmer, Greg Moss, Pearson Education India.

Reference Books:

1. Digital Design: With an Introduction to the Verilog HDL, M. Morris Mano, Michael D. Ciletti, Pearson.

Course Code: **CSE215C**	**Data Structures Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- Define, understand, describe and implement linear data structures like stack, queues, linked lists and nonlinear data structures like trees and graphs.
- Design and trace the algorithms for various operations on different data structures studied.
- Understand and implement various searching and sorting techniques
- Write programs in C to simulate operations and applications of data structures learnt.

List of Experiments

1. Program on arrays.
2. Implementation of String Manipulation.
3. Program that uses functions to perform the following operations on singly linked list:
   (i) Creation (ii) Insertion (iii) Deletion (iv) Traversal.
4. Program that uses functions to perform the following operations on doubly linked list:
   (i) Creation (ii) Insertion (iii) Deletion (iv) Traversal.
5. Program that uses functions to perform the following operations on circular linked List:
   (i) Creation (ii) Insertion (iii) Deletion (iv) Traversal.
6. Program that implements stack (its operations) using:
   (i) Arrays (ii) Linked list (Pointers).
7. Program to Implement array-based circular queue.
8. Program that implements the following sorting:
   (i) Bubble sort (ii) Selection sort (iii) Quick sort
   (iv) Insertion sort (v) Merge sort (vi) Heap sort.
9. Program to perform the following operations:
   (a) Insert an element into a binary search tree.
   (b) Delete an element from a binary search tree.
   (c) Search for a key element in a binary search tree.
10. Program to perform the following operations:
    (a) Insert an element into an AVL tree.
    (b) Delete an element from an AVL tree.
    (c) Search for a key element in an AVL tree.
11. Basic programs on implementation of graphs.

Course Code: **CSE216C**	**Object Oriented Programming Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- To identify and practice the object-oriented programming concepts and techniques,
- To solve real-world problems through object-oriented approach,
- Apply C++ features to program design and implementation.,
- Use C++ to demonstrate practical experience in developing object-oriented solutions,
- Analyze a problem description and design and build object-oriented software using good coding practices and techniques, and
- Use common software patterns in object-oriented design and recognize their applicability to other software development contexts.

List of Experiments

1. Design a class to represent a bank account. Include the following.
Data Members
- Name of the depositor
- Account number
- Type of account
- Balance amount in the account

Methods
- To assign initial values
- To deposit an amount
- To withdraw an amount after checking balance
- To display the name and balance

Incorporate a constructor to provide initial values.

2. Assume that a bank maintains two kinds of account for its customers, one called saving account and the other current account. The saving account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if it falls below a specified level, a service charge is imposed. Create a class *Account* that stores customer name, account number, and type of account. From this derive the classes *Curr-acct* and *Sav-acct* to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks.
- Accept deposit from a customer and update the balance
- Display the balance
- Compute and deposit interest
- Permit withdrawal and update the balance
- Check for the minimum balance, impose penalty, if necessary and update the balance.

Do not use any constructors. Use methods to initialize the class members.

3. Develop a program which will read a string and rewrite it in the alphabetical order. For example, the word *STRING* should be written as *GINRST.*

4. Design a generic function for finding the largest of three numbers.

5. An election is contested by 5 candidates. The candidates are numbered 1 to 5 and the voting is done by marking the candidate number on the ballot paper. Write a C++ program to read the ballots and count the votes cast for each candidate using an array. In case, a number read is outside the range 1 to 5, the ballot should be considered as a 'spoilt ballot' and the program should also count the number of spoilt ballots.

6. Create a class by name *date* with the member data as *day*, *month* and *year*.
Perform the following:
- Overload all relational operators $<, <=, >, >=, ==, =$
- Overload ++ operator to increment a date by one day
- Overload + to add given number of days to find the next date
- Provide the necessary function to use the statement like days=dt; where days is an int variable and dt is an object of date class. The statement is intended to assign the number of days elapsed in the current year of the date to the variable days. Note that this is a case of conversion from derived type to basic type.

7. Design a class template by name Vector and perform the following:
- Find the smallest of the element in the Vector
- Search for an element in the Vector
- Find the average of the element in the array.

8. Exception Handling

Course Code: **CSE217C**	**Database Management System Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- Transform an information model into a relational database schema and to use a data definition language to implement the schema using a DBMS.
- Analyze the database using queries to retrieve records.
- Formulate queries using DDL, DML, DCL and TCL commands.
- Develop application programs using PL/SQL.
- Analyze front end tools to design forms, menus, etc. and establish back-end connectivity Develop solutions using database concepts for real-time requirements.

List of Experiments

1. Introduction to SQL, RDBMS: Visualizing the architecture of RDBMS, Different data types and its implementation.
2. SQL commands: Implementation of Creating and managing SQL tables, DDL (Data definition language): Implementation of Create, Alter, drop, rename, truncate, comment.
3. Implementing Operators in SQL: Comparison operators, Arithmetic operators.
4. Implementing Operators in SQL: Relational operators, Logical operators (AND, NOT, OR), Special operators (BETWEEN, IS NULL, EXISTS, IN, LIKE).
5. Implementing Aggregate Functions in SQL: SQL functions: (COUNT, MIN, SUM, MAX, AVERAGE, LIKE).
6. Manipulating Data in tables: Data manipulation Language: (Implementing Select, Insert, Update, Delete, merge.), Retrieval of data from the table: Implementing queries on single table.
7. Implementation of constraints: Not null, Primary Key, Unique, Check, Foreign key)
8. Combining Tables and execution of queries on such tables: Perform Join, inner join, outer join, natural join and subtypes of each.
9. Combining the tables using operations like Union, Intersect, minus and executing the queries.

MongoDB:

10. Introduction to NoSQL Databases.
11. MongoDB Ecosystem.
12. CRUD Operations:
    a) Create
    b) Read
    c) Update
    d) Delete

Course Code: **CSE218C**	**Digital Electronics and Logic Design Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- How digital values of analog signals are represented in different logic families.
- Description of truth tables for digital logic circuits.
- Gate level implementation of a number of circuits like multiplexers etc.
- Evaluation of number of constraints (parameters) of combinational and sequential logic circuits.
- To implement the logic circuits in the design of VLSI / IC circuits using hardware descriptive languages like VHDL, VERILOG etc.

List of Experiments

1. To verify the truth table of AND, OR and NOT gates.
2. Design NAND, NOR, XOR and X-NOR gates and also verify their truth tables.
3. Design Half Adder and verify its truth table.
4. Design Full Adder and verify its truth table.
5. Design a Binary Adder.
6. Design Half subtractor and verify its truth table
7. Design Full subtractor and verify its truth table.
8. Design multiplexer and demultiplexer using 2-input NAND gates.
9. Design Encoders and Decoders.
10. Realization of Flip-Flops (SR, JK, T, D flip-flops).

Course Code: **CSE210A**	**Seminar**	Credits: - **Compulsory Audit**

Description:

In the third semester, students have to take compulsory audit course titled ***Seminar***. In this course they have to present a PowerPoint Presentation on a pre-approved topic from the department. The topic of the seminar should be chosen by keeping in view the ***current technological trends and advancements*** in computer science and engineering. A departmental committee will evaluate the individual students' performance on the basis of their presentation and knowledge about the topic chosen for seminar.

SEMESTER IV

Course Code: **CSE260C**	**Design and Analysis of Algorithms**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Understand and use asymptotic notations to analyze the performance of algorithms
- Understand and analyze the design of algorithms using Brute force, Divide and Conquer, Dynamic
- Programming, Greedy technique, Backtracking, Branch and Bound techniques.
- Compare and contrast various search and sorting techniques.
- Apply the various algorithms to solve problems and analyze their efficiency.

Unit – I **(10)**

Introduction: Algorithm Design paradigms- motivation, Concept of algorithmic efficiency, Run time analysis of algorithms, Asymptotic Notations.
Divide and Conquer: Structure of divide and conquer algorithms: examples, Binary search, Merge Sort, Quick sort, Analysis of divide and conquer run time. Recurrence Relations, Master Theorem for solving Recurrence Relations.

Unit – II **(8)**

Greedy method: Overview of the greedy paradigm, examples of exact optimization solutions (minimum cost spanning tree), approximate solution (Knapsack problem), Huffman coding, Single source shortest path.

Unit – III **(9)**

Dynamic Programming: Overview, difference between dynamic programming and divide and conquer, applications: Shortest Path in Multistage Graph, Non-fractional (0/1) Knapsack problem, Matrix Chain Multiplication, Travelling salesman problem, Longest common sequence.

Unit – IV **(8)**

Graph searching and traversal: Overview, traversal methods, depth first and breadth first search. Dijkstra's and Bellman-Ford Algorithm for finding Single source shortest paths. All pair shortest paths and matrix multiplication, Floyd-Warshall algorithm for all pair shortest paths.

Unit – V **(10)**

Back Tracking: Overview, 8-queen problem and Knapsack problem.
Branch and Bound: LC searching, bounding, FIFO branch and bound, Applications: 0/1 Knapsack problem, Travelling salesman problem.
Computational complexity: Complexity measures, Polynomial vs non-polynomial time complexity; NP hard and NP complete classes, examples.

Textbooks:

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest, Clifford Stein, "*Introduction to Algorithms*", 2nd Ed., PHI, 2004.

Reference Books:

2. Ellis Horowitz and Sartaz Sahani, "*Computer Algorithms*", Galgotia Publications, 1999.
3. V. Aho, J. E. Hopcroft, J. D. Ullman, "*The Design and Analysis of Computer Algorithms*", Addition Wesley, 1998.
4. D. E. Knuth, "*The Art of Computer Programming*", 2nd Ed., Addison Wesley, 1998.

Course Code: **CSE261C**	**Computer Architecture and Organization**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Describe how different functional units in a computer system operate, interact and communicate.
- Describe the detailed architecture of the central processing unit, control unit, input-output unit and memory unit.
- Describe the representation, arithmetic and computation of data at machine level.
- Describe how various memory units and input-output devices are accessed in a computer system.
- Describe how the throughput of a computer system can be increased using pipelining.

Unit – I (8)

Register Transfer and Micro-operations: Defining computer architecture and computer organization, Register Transfer Language, Register Transfer, Bus and Memory Transfers, Bus system construction, Arithmetic Micro-operations, Logic Micro-operations, Shift Micro-operations, Arithmetic Logic Shift Unit.

Unit – II (8)

Basic Computer Organization and Control Unit Design: Instruction Codes, Computer Registers, Computer Instructions, Hardwired Control, Timing and Control, Instruction Cycle, Memory Reference Instructions, Input-Output and Interrupt, Complete Computer Description, Design of Basic Computer, Design of Accumulator Logic, Data-path, Microprogrammed Control: Control Memory, Address Sequencing, Micro Program Example, Design of Control Unit.

Unit – III (9)

Central Processing Unit: Introduction, General Register Organization, Stack Organization, Machine Instructions, Instruction Formats, Addressing Modes, Data Transfer and Manipulation, Program Control, Reduced Instruction Set Computer.
Computer Arithmetic: Addition and Subtraction, Multiplication Algorithms, Division Algorithms, Floating-Point Arithmetic Operations, Decimal Arithmetic Unit, Decimal Arithmetic Operations.

Unit – IV (10)

Input-Output Organization: Peripheral Devices, Input-Output Interface, Asynchronous Data Transfer, Modes of Transfer, Priority Interrupt, Direct Memory Access (DMA), Input-Output Processor, Serial Communication.
Pipelining: Pipelining Basics, Arithmetic Pipeline, Instruction Pipeline, Pipeline Hazards.

Unit – V (10)

Memory Organization: Memory Hierarchy, Main Memory, Auxiliary Memory, Associative Memory, Cache Memory, Virtual Memory, Memory Management Hardware.
Introduction to Multiprocessors.

Textbook:

1. Mano, M. Morris. *Computer system architecture*. 3rd Edition, Prentice-Hall.

Reference Books:

1. Hayes, John P. *Computer architecture and organization*. McGraw-Hill.
2. Stallings, William. *Computer organization and architecture: designing for performance*. Pearson.
3. Hamacher, V. Carl, Zvonko G. Vranesic, Safwat G. Zaky, Zvonko Vransic, and Safwat Zakay. *Computer organization*. McGraw-Hill.

Online Resources:

1. https://youtube.com/playlist?list=PLdl2B3KkY5upvcVCwXImlzDhop9Xhua9M&si=0xqCH_iNXn6urxM1
2. https://archive.nptel.ac.in/courses/106/105/106105163/
3. https://onlinecourses.nptel.ac.in/noc23_cs67/preview

Course Code: **CSE262C**	**Web Technologies**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To understand the basic concepts of web programming and internet,
- To understand how the client-server model of Internet programming works,
- To understand interactive web applications,
- To learn the latest technologies, including JavaScript, Node, and Web3 development,
- To understand the latest frontend and backend web development technologies.

Unit – I (5)

Introduction to Web System
Internet Overview, WWW, Web Protocols, Web Browsers and Web Servers, Web System Architecture. URL, Domain Name, Client and Server-side Scripting. MVC, Implementing Model-View-Controller.

Unit – II (10)

HTML and CSS
HTML5 Basics: Formatting, Colours, Images, Links, Tables, Lists, Layout, Forms, Canvas, Media.
CSS3 Basics: Selectors, Box Model, Backgrounds and Borders, Text Effects, Advanced Features.

Unit – III (10)

JavaScript and jQuery
JavaScript Basics: Functions, Arrays, DOM, Built-in Objects, Regular Expressions, Event handling, JavaScript Form Validation.
jQuery Basics: jQuery with Websites, Selecting Elements with jQuery, Manipulating Styles, Text, and Attributes with jQuery, Adding Event Listeners with jQuery, Adding and Removing Elements with jQuery, Website Animations with jQuery.

Unit – IV (9)

Backend Web Development (Node.js)
Node.js Basics: Node JS Modules, Functions, Buffer, Module, Modules Types, Core Modules, Local Modules, Modules Exports, The Node REPL (Read Evaluation Print Loops), The NPM Package Manager.

Unit – V (9)

MongoDB and Mongoose:
MongoDB: Introduction to NoSQL Databases, MongoDB Ecosystem, Replica Sets and Clusters, Advantages of MongoDB Databases, MongoDB Query Languages, MongoDB CRUD Operations: Create, Read, Update, and Delete.
Mongoose: Introduction to Mongoose, reading from the database, Data Validation, Updating and Deleting Data, Establishing Relationships and Embedding Documents using Mongoose.

Textbooks:

1. *"HTML, CSS, and JavaScript All in One*," Julie C. Meloni and Jennifer Kyrnin, Pearson Education India.
2. *"Node.js Web Development,"* David Herron, Packt Publishing Limited.
3. *"MongoDB the Definitive Guide,"* Shannon Bradshaw (Author), Eoin Brazil, and Kristina Chodorow, Shroff/O'Reilly.

Reference Books:

1. *"JavaScript and jQuery: Interactive Front-End Web Development,"* Jon Duckett, Wiley.
2. *"Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React, and Node,"* Scott Meyers, Shroff/O'Reilly
3. *"MERN Quick Start Guide,"* Eddy Wilson Iriarte Koroliova, Packt Publishing Limited.

Online Resources:

1. https://www.udemy.com/course/complete-web-development-course/
2. https://www.udemy.com/course/mongodb-the-complete-developers-guide/
3. https://www.coursera.org/learn/web-development

Course Code: **CSE263C**	**Software Engineering**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Understanding and applying the software engineering lifecycle models by demonstrating competence in communication, planning, analysis, design, construction, and deployment
- Translate a requirements specification into an implementable design, following a structured and organized process.
- Defining the basic concepts and importance of Software project planning concepts like cost estimation, and scheduling.
- Applying different testing and debugging techniques and analyzing their effectiveness.
- Defining software maintenance concepts, software quality and reliability on the basis of international quality standards.

Unit – I **(9)**

Introduction: Software engineering discipline-Evolution and Impact, program vs software product. Software Crisis, Software engineering a layered technology – processes, methods and tools. Software life cycle models: Waterfall, Prototype, Evolutionary and Spiral models, Overview of object oriented software development and Agile software.

Unit – II **(9)**

Software Requirement Analysis and Specifications: Problem Analysis, Data Flow Diagrams, Data Dictionaries, Entity-Relationship diagrams, Software Requirement and Specifications. Software Project Planning, Project Manager Responsibilities, Cost estimation, static, single and multivariate models, COCOMO model. Risk Management, Software Configuration Management.

Unit – III **(9)**

Software Design: Cohesion and Coupling, Classification of Cohesiveness and Coupling, Function Oriented Design, Object Oriented Design. Software Reliability: Introduction, Failure and Faults, Reliability Models: Basic Model, Logarithmic Poisson Model, Calendar time Component.

Unit – IV **(9)**

Software Testing: Software process, Functional testing: Boundary value analysis, Equivalence class testing, Decision table testing, Cause effect graphing, Structural testing: Path testing, Data flow and mutation testing, Unit testing, Integration and System testing, Black box testing, White- box testing, Debugging, Testing Tools and Standards, Using testing tools in lab.

Unit – V **(9)**

Software Quality, Software Quality Management System, ISO 9000, 9001, Capability Maturity Model, Personal Software Process, Six Sigma, Software Maintenance, Reverse Engineering, Software Reengineering, Configuration Management.

Textbooks:

1. "*Software Engineering A. Practitioner's Approach*", by Pressman, MGH.
2. "*Fundamental of Software Engineering",* by Rajib Mall, PHI.

Reference Books:

1. *"Software Engineering",* by James F. Peters, Wiley.
2. *"Software Project Management from Concept to Development",* by Kieron Conway, Dreamtech Press.
3. "*Software Engineering*", by Sommerville, Pearson Education.
4. "*Software Engineering",* by Jawadekar, TMH.

Course Code: **CSE264C**	**Data Communication**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Understand the model of data communication
- Understand the working of various Transmission Media.
- Understand various signal conversion techniques.
- Understand various error detection and correction techniques.
- Understand multiplexing techniques

Unit – I **(8)**

Introduction to Data Communications and Networking, Communications Model, Data Communications, Networks, The Internet, Data Transmission: Concepts and Terminology, Analog and Digital Data Transmission, Transmission Impairments, Channel Capacity.

Unit – II **(8)**

Guided Transmission Media, Wireless Transmission, Wireless Propagation, Line-of-Sight Transmission, Data transmission: simplex, half duplex and full duplex, Asynchronous and Synchronous Transmission.

Unit – III **(10)**

Digital Data and Digital Signals, Digital Data and Analog Signals, Analog Data and Digital Signals, Analog Data and Analog Signals.

Unit – IV **(9)**

Types of Errors, Error Detection, Error Correction, Line Configurations.
Flow Control, Error Control, High-Level Data Link Control (HDLC)

Unit – V **(10)**

Multiplexing Techniques: Frequency Division Multiplexing, Synchronous Time-Division Multiplexing Statistical Time-Division Multiplexing, Asymmetric Digital Subscriber Line
Spread Spectrum, Frequency Hopping Spread Spectrum, Direct Sequence Spread Spectrum, Code-Division Multiple Access

Textbooks:

1. William Stallings: Data and Computer Communications, 9th Ed, PHI
2. Data Communications and Networking: Behrouz A. Forouzan

Reference Books:

1. Andrew Tanenbaum, “Computer Networks” PHI
2. Sklar, “Digital Communications fundamentals and Applications”
3. Keizer, “Local Area Networks” McGraw Hill

Course Code: **CSE265C**	**Design and Analysis of Algorithms Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- Understand and use asymptotic notations to analyze the performance of algorithms
- Understand and analyze the design of algorithms using Brute force, Divide and Conquer, Dynamic
- Programming, Greedy technique, Backtracking, Branch and Bound techniques.
- Compare and contrast various search and sorting techniques.
- Apply the various algorithms to solve problems and analyze their efficiency

List of Experiments

1. Simple Experiments on time and space complexity of a program
2. Sort a set of elements using the Quick sort, Merge sort
3. Implement Knapsack Problem using Greedy Algorithm
4. Implement Huffman Codes using Greedy Algorithm
5. Implement 0/1 Knapsack problem using Dynamic Programming.
6. Implement Matrix Chain Multiplication using Dynamic Programming
7. Implement Traveling Salesman Problem using Dynamic Programming
8. Implement Longest common subsequence using Dynamic Programming
9. Find Minimum Cost Spanning Tree of a given undirected graph using Kruskal's algorithm.
10. Print all the nodes reachable from a given starting node in a digraph using BFS method
11. Check whether a given graph is connected or not using DFS method.
12. Find Minimum Cost Spanning Tree of a given undirected graph using Prim's algorithm
13. Implement All-Pairs Shortest Paths Problem using Floyd's algorithm.
14. Implement N Queen's problem using Backtracking

Course Code: **CSE266C**	**Web Technologies Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

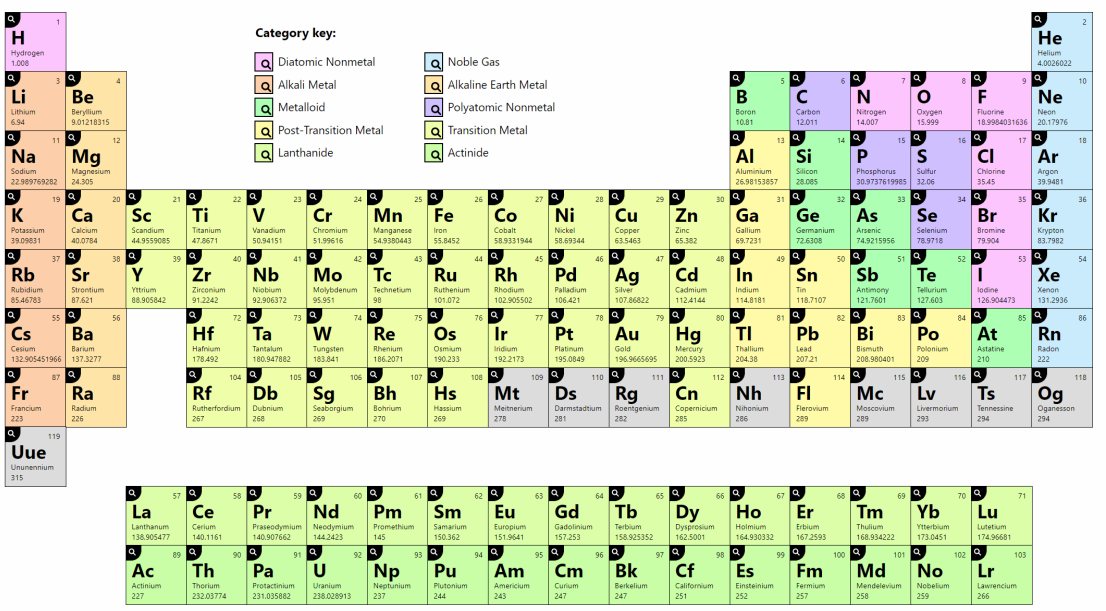
- To develop web-based application using suitable client side and server-side web technologies,
- To develop solution to complex problems using appropriate method, technologies, frameworks, web services and content management,
- To learn why server-side JavaScript is useful,
- To learn how Node.js is architected to allow high scalability with asynchronous code,
- To develop basic web applications with Node.js.

List of Experiments

1. Design an English alphabet chart such that on clicking the alphabet the appropriate example must be displayed using HTML client-side image mapping.

2. Develop and demonstrate the usage of inline, internal and external style sheet using CSS3.

3. Design the online periodic table of elements as shown below using HTML and CSS.



4. Validate the ISBN number of a given book using regular expressions in JavaScript.

5. Write JavaScript to validate the following fields of the Registration page.
   a. *First Name* (Name should contain alphabets and the length should not be less than 6 characters).
   b. *Password* (Password should not be less than 6 characters length).
   c. *E-mail id* (should not contain any invalid and must follow the standard pattern name@domain.com)
   d. *Mobile Number* (Phone number should contain 10 digits only).
   e. *Last Name and Address* (Should not be Empty).

6. Develop an online application to find the transpose of the given matrix. Obtain the number elements from the user based on the number of rows and columns using JavaScript.

7. Build a basic CRUD application with Node.js and MongoDB.

8. Use Express.js to Create Node.js Web Apps.

Course Code: **CSE267C**	**Project-I**	Credits: **01** L – **0** P – **2**

Description:

- In the fourth semester, students have to take Project-I (Mini Project) of one credit.
- Students need to identify area of their interest in which they would opt (not necessarily) their Project-II, Project-III and Project-IV in later semesters.
- Students have to do extensive literature survey of their field of interest and also learn certain tools / software required.
- Emphasis should be on problem solving, innovation, and design.
- Deliverables: product, project report, codebase, user manual, demonstration etc.

SEMESTER V

Course Code: **CSE310C**	**Operating System**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Describe the detailed functioning of an operating system.
- Describe the various types, models and structures of operating systems.
- Describe how processes and threads perform inter-process and inter-thread communication.
- Describe and simulate the policies for process synchronization, scheduling and deadlocks.
- Describe and simulate memory management, file management and disk management systems.

Unit – I (7)

Introduction: What is an operating system, history of operating systems, types of operating systems, operating system concepts, functions of an operating system, kernel data structures, computing environments, open-source operating systems, user and operating-system interface, system calls, types of system calls, operating-system design and implementation, system programs, operating system structures, operating-system debugging, system boot, programming projects.

Unit – II (10)

Processes: Process concept, Process scheduling, Operations on processes, Interprocess communication.
Threads: Overview, Multicore Programming, Multithreading Models.
CPU Scheduling: Scheduling criteria, Scheduling algorithms.

Unit – III (10)

Process Synchronization: Concurrency, The Critical-Section problem, Peterson's solution, Synchronization Hardware, Mutex Locks, Semaphores, Classic problems of synchronization – The Bounded-Buffer problem, The Readers-Writers problem, The Dining-Philosophers problem, Monitors.
Deadlocks: Characterization, prevention, avoidance, detection, recovery.

Unit – IV (10)

Main Memory Management: Background, Swapping, Contiguous memory allocation, Segmentation, Paging, Structure of the page table.
Virtual Memory Management: Background, Demand Paging, Copy-on-write, Page Replacement, Allocation of Frames, Thrashing.

Unit – V (8)

Mass-Storage Management: Disk scheduling, Disk management, Swap-space management, RAID.
File System Interface: File concept, File access methods, Directory and Disk Structure, File Sharing, protection.
File-System Implementation: File-System Structure, File-System Implementation, Directory Implementation, File Allocation Methods, Free-Space Management, NFS.

Textbooks:

1. Silberschatz, Abraham, Peter Baer Galvin, and Greg Gagne. *Operating system principles*. 9th Edition, John Wiley & Sons.

Reference Books:

1. Tanenbaum, Andrew S., and Bos Herbert. *Modern operating systems. 4th Edition, Pearson.*
2. Stallings, William, and Goutam Kumar Paul. *Operating systems: internals and design principles*. 9th Edition, Pearson, 2012.
3. Tanenbaum, Andrew S., and Albert S. Woodhull. *Operating systems: design and implementation*. Prentice Hall, 1997.
4. Crowley, Charles. *Operating systems: a design-oriented approach*. McGraw-Hill Professional.
5. Nutt, Gary J. *Operating Systems: A Modern Perspective, Lab Update*. Addison-Wesley Longman Publishing Co., Inc

Online Resources:

1. https://os-book.com/OS10/index.html
2. https://nptel.ac.in/courses/106105214

Course Code: **CSE311C**	**Computer Networks**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Understanding of the fundamental concepts of computer networks.
- Demonstrate proficiency in the data link layer, including knowledge of various design issues.
- Should have a deep comprehension of the network layer like subnetting and routing mechanisms.
- Able to explain and work with key internetworking protocols.
- Students should have a strong grasp of the application layer.

Unit – I **(6)**

Introduction: Uses of Computer Networks, Network and Protocol Architecture, Reference Model (ISO-OSI, TCP/IP-Overview), Types of networks (LAN, MAN and WAN), Different network topologies like star, ring, hybrid, tree etc. IEEE standards.

Unit – II **(10)**

Data Link layer – Design Issues, Error detection and Correction techniques, Flow control algorithms, Framing techniques, Switched LANs: L2 addressing and ARP, Ethernet frame structure, learning switches. Medium access protocols: Polling vs. contention-based: TDM, Aloha, CSMA/CD.

Unit – III **(10)**

Network Layer: Network architecture and Performance, Need for an Internet address, and its design: IPv4 and IPv6, structure of IP datagram, IP forwarding, NATs, sub-netting, Routing protocols: Link state routing. Distance vector routing: count-to-infinity, routing convergence.

Unit – IV **(10)**

Internetworking: Internet control protocols: ICMP, ARP, RARP, BOOTP, DHCP, intra-domain (OSPF) and inter-domain (BGP)routing.
Transport Layer: Transport-Layer Services, Multiplexing and Demultiplexing, Connectionless Transport: UDP, Principles of Reliable Data Transfer, Connection-Oriented Transport: TCP, Flow Control, Congestion Control.

Unit – V **(9)**

Application Layer-Principles of Network Applications, The Web and HTTP, File Transfer: FTP, Electronic Mail in the Internet, Domain Name system (DNS), Peer-to-Peer Applications. Audio and video streaming over UDP, HTTP. Adaptive streaming. Voice over IP.

Textbooks:

1. J.F. Kurose and K.F. Ross, Computer networking: a top-down approach, Pearson
2. Larry L.Peterson, Peter S. Davie, “Computer Networks”, Elsevier, Fifth Edition, 2012.

Reference Books:

1. Computer Networks – Third Edition – Andrew S. Tanenbaum, Prentice Hall of India.
2. U. Black, “Computer Networks-Protocols, Standards and Interfaces”, PHI, 1996.
3. Laura Chappell, “Introduction to Cisco Router Configuration”, Techmedia, 1999.
4. Michael A. Miller, “Data & Network Communications”, Vikas Publication, 1998.
5. William A. Shay, “Understanding Data Communications & Networks”, Vikas Publication, 1999.
6. W. Stallings, “Computer Communication Networks”, PHI, 1999.

Course Code: **CSE312C**	**Python Programming**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Gain a solid understanding of Python programming language basics.
- Demonstrate proficiency in using control structures.
- Acquire the skills to manipulate common data structures in Python.
- Understand and apply the principles of Object-Oriented Programming (OOP).
- Gain exposure to Python libraries and frameworks for practical applications.

Unit – I **(8)**

Introduction to Python: Why Python?, History, Features, and Applications, Python 2 vs. Python 3, Setting up Python Environment; Installation of Python, Introduction to IDEs (Integrated Development Environments), Python Basics; Variables, Data Types, and Operators, Basic Input and Output.

Unit – II **(8)**

Control Flow and Functions: Control Structures; Conditional Statements (if, elif, else), Loops (for, while), Functions; Defining Functions, Parameters and Return Values, Scope and Lifetime of Variables, Exception Handling; Handling Errors with Try-Except Blocks.

Unit – III **(8)**

Data Structures in Python: Lists and Tuples; Operations and Methods, Dictionaries and Sets; Key-Value Pairs and Uniqueness, String Manipulation; String Operations and Methods.

Unit – IV **(8)**

Object Oriented Programming (OOP): Introduction to OOP; Classes and Objects, Inheritance and Polymorphism; Code Reusability and Flexibility, File Handling; Reading and Writing Files in Python.

Unit – V **(8)**

Python Libraries and Applications: Overview of Python Libraries; NumPy, Pandas, Matplotlib, Introduction to Web Development with Flask; Basics of Flask Framework, Introduction to Data Science with Python; Basic Concepts and Applications.

Textbooks:

1. "Python Cookbook" by David Beazley and Brian K. Jones
2. "Fluent Python" by Luciano Ramalho

Reference Books:

1. Python for Data Analysis" by Wes McKinney.
2. "Automate the Boring Stuff with Python" by Al Sweigart
3. "Python Crash Course" by Eric Matthes.

Course Code: **CSE313C**	**Microprocessors, Peripherals and Interfacing**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To have a complete knowledge of architecture of micro-processor 8085, its pin layout and description of all signals.
- To understand the instruction cycle, timing diagrams, registers, flags etc. of 8085 micro-processor in detail.
- To have a knowledge of instructions, addressing modes, interrupts, subroutines, conditional call instructions of 8085 and be able to perform assembly language programming for numerous operations in 8085.

Unit – I **(8)**

Microcomputer Structure and Operations: Basic Microcomputer Elements, Microprocessors and Memory: Typical 8, 16 and 32 bit Microprocessors, 8085 Microprocessor - Memory Technology, Pin Description and Internal Architecture of 8085, registers, flags, buses. Architecture of 8-bit Microprocessor: Instruction Classification: 1-byte, 2-byte and 3-byte instructions.

Unit – II **(11)**

Assembly Language Programming: Programming Model of 8085, Registers, Fetch, Execute Operation of CPU, Instruction Set, Addressing Modes, Basic Operations, Microprocessor Arithmetic, Program Flow, Control Using Looping and Branching. Concepts of instruction cycle, machine cycle, and t-states. Opcode fetch machine cycle, memory read/write machine cycles, I/O read/write machine cycles, time delays.

Unit – III **(9)**

Counters, time delays and related examples. Stacks and subroutines. Interrupts: Interrupt structure of 8085 microprocessor, vectored and non-vectored interrupts, Priority Management, Address Decoding.

Unit – IV **(10)**

Microprocessors Interfacing: Interfacing concepts, Parallel Input Output, Memory mapped I/O, Memory Interfacing, Direct Memory Access, 8237 DMA Controller. The Serial Subsystems. 8255 Programmable Peripheral Interface, Analog Converter Subsystem.

Unit – V **(8)**

Introduction to 8086 architecture: Pin Configuration, Main features and addressing modes, difference between 8085 and 8086. Introduction of 8088, 80186, 80386 microprocessors.

Textbooks:

1. Microprocessor Architecture, Programming, and Applications with the 8085 –Ramesh S. Gaonkar, Pub: Penram International.
2. A.K. Ray and K.M. Bhurchandi Advanced Microprocessors and Peripherals, third Edition, Tata McGraw Hill, 2012. 2. Barry B Bray, The Intel Microprocessor 8086/8088, 80186,80286, 80386 and 80486 Arcitecture, programming and interfacing, PHI, 8th Edition, 2009.
3. Hall D.V. "Microprocessor and Interfacing-Programming and Hardware", 2nd Ed., Tata McGraw-Hill Publishing Company Limited, 2008.

Reference Book:

1. 8085 Microprocessor and its Applications, by A. Nagoor Kani, Third Edition, TMH Education Pvt. Ltd.
2. Mohamed Rafiquazzaman, Microprocessor and Microcomputer based system design, Universal Book stall, New Delhi.

Course Code: **CSE314C**	**Probability and Statistics**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Solve basic probability problems.
- Understand the random variables and various related concepts like joint, conditional and marginal probabilities.
- Identify and work with different distributions
- Appreciate the common ground between probability and statistics.
- Solve basic probability problems.

Unit – I **(9)**

Statistics: Measures of central tendency and Measures of variations (Dispersions), Moments, Measures of Skewness and Kurtosis. Moment generating functions, problems.

Unit – II **(9)**

Probability: Random experiment, sample space, events, classical, statistical and axiomatic definitions of probability. Statements and proof of theorems on addition and multiplication of probabilities.

Unit – III **(9)**

Conditional Probability: Bayes theorem on conditional probability. Random variables, Derivation of formulae for mean, variance and moments of random variables for discrete and continuous cases. Laws of expectation.

Unit – IV **(9)**

Standard Distributions: Bernoulli and Binomial distributions, Poisson and Normal Distributions, Beta and Gamma Distribution, t-Distribution, F-Distribution, Chi-square Distribution. Central Limit Theorem.

Unit – V **(9)**

Method of Least Squares & Correlation: Methods of least squares, fitting of straight line and parabola of degree 'p'. Regression and Correlation. Multiple and Partial Correlation Problems.

Textbooks:

1. Fundamentals of Mathematical Statistics, by S.C.Gupta and V.K. Kapoor, Sulltan Chand & Sons New Delhi, Latest edition.
2. Statistical Theory and Methodology in Science & Engineering, by Brownlee, John Wiley & Sons.

Reference Books:

1. Introduction to Mathematical Statistics, by R.E.Walpole 3rd edition New York Macmillan publication.
2. Data Analysis for Scientists & Engineers, by Meyer, John Wiley & Sons.

Course Code: **CSE315C**	**Operating System Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

By the end of this course, students will be able to:

- Execute essential Linux commands for file system navigation, user management, and process control.
- Develop and execute shell scripts for automating tasks in a Linux environment.
- Implement system-level programs using C to perform file handling and process management using system calls.
- Perform basic Linux system administration tasks such as user/group management, software installation, and job scheduling.
- Monitor system performance and demonstrate basic networking and troubleshooting commands in Linux.

List of Experiments

1. Execute basic Linux commands for file and directory manipulation, and explore the Linux file system hierarchy.
2. Demonstrate file permission management using *chmod*, *chown*, and edit files using *vim* or *nano* editor.
3. Write a shell script to demonstrate the use of variables, user input, and command-line arguments.
4. Write a shell script using conditionals and loops to automate tasks like file checks or menu-based utilities.
5. Manage processes using Linux commands like *ps*, *top*, *kill*, *nice*, *fg*, and *bg*.
6. Implement a C program to create a child process using *fork()* and demonstrate the use of *exec()* and *wait()* system calls.
7. Write a C program using file system calls *[open(), read(), write(), close()]* to perform basic file operations.
8. Perform user and group management in Linux using commands such as *useradd*, *passwd*, *groupadd*, and *sudo*.
9. Install and remove software using package management tools (*apt*, *dpkg*) and schedule tasks using *cron* and *at*.
10. Monitor system performance using commands like *free*, *df*, *du*, *top*, and demonstrate basic networking using *ping*, *ip*, and *traceroute.*

Textbook:

1. Silberschatz, A., Galvin, P. B., & Gagne, G., Operating System Concepts, Wiley.

Reference Books:

1. Neil Matthew & Richard Stones, Beginning Linux Programming (4th Edition), Wrox Press.
2. M. G. Venkateshmurthy, Introduction to UNIX and Shell Programming, Pearson Education.
3. Yashavant Kanetkar, Let Us Linux, BPB Publications.
4. W. Richard Stevens & Stephen A. Rago, Advanced Programming in the UNIX Environment, Addison-Wesley.

Online Resources:

1. Linux Command and Shell Scripting: https://linuxcommand.org
2. https://tldp.org
3. https://explainshell.com
4. https://man7.org/linux/man-pages/
5. https://www.geeksforgeeks.org
6. https://www.tutorialspoint.com/unix

Course Code: **CSE316C**	**Computer Networks Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- Understanding of the design, troubleshooting, modeling and evaluation of computer networks
- Identify and use various networking components.
- Understand different transmission media and design cables for establishing a network.
- Implement any topology using network devices.
- Compare routing algorithms.

List of Experiments

1. Study of different types of network cables and tools
2. Implement the cross-wired cable and straight through cable using clamping tool.
3. Study of Network Devices in Detail.
4. Concept of Network IP Address
5. Creating point-to-point network and sharing files
6. Building a Local Area Network.
7. Study of basic network command and Network configuration commands.
8. Study of basic network command and Network configuration commands.
9. Configure a Network topology using packet tracer software.
10. Configure a Network topology using packet tracer software.
11. Implementation of Static Routing using Packet Tracer software.
12. Configure a Network using Distance Vector Routing protocol.
    Routing Information Protocol (RIP)
13. Configure Network using Link State Vector Routing protocol.
    Open Shortest Path First (OSPF)
14. Implementation of a VLAN using Packet Tracer
15. Implementation of a VLAN using Packet Tracer

Preferred Tool:

- Packet Tracer

Textbooks:

1. J.F. Kurose and K.F. Ross, Computer networking: a top-down approach, Pearson
2. Larry L.Peterson, Peter S. Davie, “Computer Networks”, Elsevier, Fifth Edition, 2012.

Reference Books:

1. Computer Networks – Third Edition – Andrew S. Tanenbaum, Prentice Hall of India.
2. U. Black, “Computer Networks-Protocols, Standards and Interfaces”, PHI, 1996.
3. Laura Chappell, “Introduction to Cisco Router Configuration”, Techmedia, 1999.
4. Michael A. Miller, “Data & Network Communications”, Vikas Publication, 1998.
5. William A. Shay, “Understanding Data Communications & Networks”, Vikas Publication, 1999.
6. W. Stallings, “Computer Communication Networks”, PHI, 1999.
7. William A. Shay, “Understanding Data Communications & Networks”, Vikas Publication, 1999.
8. W. Stallings, “Computer Communication Networks”, PHI, 1999.

Course Code: **CSE317C**	**Python Programming Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- Gain a solid understanding of Python programming language basics.
- Demonstrate proficiency in using control structures.
- Acquire the skills to manipulate common data structures in Python.
- Understand and apply the principles of Object-Oriented Programming (OOP).
- Gain exposure to Python libraries and frameworks for practical applications.

List of Experiments

1. Setting up Python Environment:
   - Verify the installation by running a simple Python script.
   - Introduction to Python interpreter and interactive mode.
2. Introduction to IDEs:
   - Install an Integrated Development Environment (IDE) such as PyCharm, Visual Studio Code, or IDLE.
   - Explore the features and interface of the chosen IDE.
   - Create and run a Python script within the IDE.
3. Python Basics:
   - Practice defining variables and assigning values of different data types (integers, floats, strings, booleans). Perform arithmetic operations using Python operators.
   - Use print() function for basic input and output operations.
4. Control Flow and Functions:
   - Write Python code to implement conditional statements (if, elif, else) for decision-making.
   - Create loops (for, while) to iterate over data structures or perform repetitive tasks.
   - Define and call functions with parameters and return values.
   - Demonstrate understanding of variable scope and lifetime.
5. Exception Handling:
   - Write Python code to handle errors using try-except blocks.
   - Handle specific exceptions and provide appropriate error messages.
   - Test exception handling by intentionally raising errors.
   - Perform string manipulation tasks using string operations and methods.
6. Data Structures in Python:
   - Create and manipulate lists and tuples using built-in methods and operations.
   - Explore dictionaries and sets, understanding their properties and methods.
7. Object Oriented Programming (OOP):
   - Define classes and create objects in Python.
   - Implement inheritance and polymorphism concepts to demonstrate code reusability.
   - Showcase file handling by reading from and writing to files using Python.
8. Python Libraries and Applications:
   - Explore popular Python libraries such as NumPy, Pandas, and Matplotlib.
   - Import and use functions from these libraries to perform basic data analysis and visualization tasks.
   - Learn the basics of Flask framework for web development, including routing and rendering templates.
10. Introduction to Data Science with Python:
    - Explore basic concepts of data science and its applications.
    - Perform data manipulation and analysis using Pandas library.
    - Visualize data using Matplotlib for creating plots and charts.
11. Introduction to Data Science with Python:
    - Explore basic concepts of data science and its applications.
    - Perform data manipulation and analysis using Pandas library.
    - Visualize data using Matplotlib for creating plots and charts.

Textbooks:

1. Eric Matthes, Python Crash Course, No Starch Press, 2nd Edition.

Reference Books:

1. Wes McKinney, Python for Data Analysis, O'Reilly Media.
2. Al Sweigart, Automate the Boring Stuff with Python, No Starch Press.
3. Mark Lutz, Learning Python, O'Reilly Media.

Online Resources:

1. https://www.python.org
2. https://www.w3schools.com/python/
3. https://developers.google.com/edu/python
4. https://www.kaggle.com/learn/python
5. https://www.youtube.com/c/CoreySchafer

Course Code: **CSE318C**	**Microprocessors, Peripherals and Interfacing Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- By the end of this laboratory course, students will be able to:
- Develop and execute assembly language programs using the 8085 instruction set for performing basic arithmetic, logical, and control operations.
- Analyze and implement algorithms for data manipulation tasks such as searching, sorting, and code conversion using 8085 microprocessor.
- Demonstrate proficiency in memory and register operations including stack usage, subroutine calls, and block data transfer.
- Interface simple I/O devices like LEDs and 7-segment displays with the microprocessor system (or simulator), and control them through software.
- Apply microprocessor programming concepts in solving real-time problems through structured, modular, and optimized assembly code.

List of Experiments

1. Write assembly language programs to perform
    a. 8-bit addition and subtraction
    b. 16-bit addition and subtraction using register pairs
2. Write assembly language programs to perform
    a. Multiplication of two 8-bit numbers using repeated addition
    b. Division of an 8-bit number by another using repeated subtraction
3. Perform bitwise AND, OR, XOR on two numbers.
4. Transfer a block of N bytes from one memory location to another and exchange two blocks.
5. Search the largest and smallest number from a list of N numbers.
6. Write a program to sort N numbers using Bubble Sort.
7. Count the number of even and odd elements from a list.
8. Write assembly language programs to perform
    a. Binary to BCD and BCD to Binary
    b. Binary to ASCII and ASCII to Binary
9. Use PUSH, POP instructions
10. Implement subroutines using CALL and RET
11. Display digits 0–9 on a 7-segment display
12. Create a binary counter output using LEDs

Textbook:

1. Ramesh S. Gaonkar, Microprocessor Architecture, Programming and Applications with the 8085, Penram International Publishing.

Reference Books:

1. A.K. Ray and K.M. Bhurchandi, Advanced Microprocessors and Peripherals, McGraw Hill Education.
2. Vijay Goel & Himanshu Goel, 8085 Microprocessor Programming and Interfacing, Pearson Education.
3. Douglas V. Hall, Microprocessors and Interfacing: Programming and Hardware, McGraw Hill.

Online Resources:

1. NPTEL Online Course, Microprocessors and Microcontrollers by Prof. Santanu Chattopadhyay, IIT Kharagpur
2. GNUSim8085 Simulator: https://sourceforge.net/projects/gnusim8085/
3. EMU8085 Emulator: http://www.emu8085.net/
4. https://www.tutorialspoint.com/microprocessor/microprocessor_8085_architecture.htm
5. https://www.electronicshub.org/8085-microprocessor-programs/

SEMESTER VI

Course Code: **CSE360C**	**Theory of Computation**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Comprehend regular languages and finite automata and develop ability to provide the equivalence between regular expressions, NFAs, and DFAs.
- Disambiguate context-free grammars by understanding the concepts of context-free languages and push-down automata
- Apply the concepts of recursive and recursively enumerable languages and design efficient Turing Machines.
- To classify machines by their power to recognize languages.
- Solve analytical problems in related areas of theory in computer science

Unit – I **(9)**

Introduction to finite Automata: Alphabets, Language, Regular Expression, Definitions of Finite State Machine, Transition Graphs, Deterministic and Non-deterministic Finite State Automata, Algorithm to convert NDFA to DFA, Minimization of DFA, Finite State Machine with output- Moore machine and Melay Machine, Conversion of Moore machine to Melay Machine and Vice-Versa.

Unit – II **(9)**

Regular languages and properties: Regular Languages and Regular Expressions, Regular Grammar, Conversion of DFA to Regular Expression, Thompson's Construction to Convert Regular Expression to NDFA Pumping Lemma, Properties and Limitations of Finite state machine

Unit – III **(10)**

Context Free Grammar and Languages: Context Free Grammar, Derivation tree and Ambiguity, Application of Context-free Grammars, Chomsky and Greibach Normal form, Closure properties of CFL, CKY Algorithm, Decidable properties of Context Free Grammar

Unit – IV **(7)**

Pushdown Automata: Definition of the pushdown automata, the languages of a PDA, Equivalence of PDA's and CFG's, Deterministic and Non-Deterministic PDA. Notion of Acceptance by PDA.

Unit – V **(8)**

Introduction to Turing Machine: Turing machine definition and design of Turing Machine, Church-Turing Thesis, Variations of Turing Machines, combining Turing machine, Universal Turing Machine, Chomsky Hierarchy, Post correspondence problem

Textbooks:

1. Jhon E. Hopcroft, Rajeev Motwani, Jeffery Ullman: *Introduction to Automata Theory, Languages and Computation*, 3rd edition, Pearson education, 2007
2. Martin J.C., "*Introduction to Languages and Theory of Computation*", 3rd Edition, Tata McGraw-Hill Publishing Company Limited.

Reference Books:

1. K.L.P. Mishra: *Theory of Computer Science, Automata, Languages and Computation*, 3rd edition, PHI, 2007
2. Jhon C martin: *Introduction to languages and Automata Theory*, 3rd edition, Tata McGraw-hill,2007
3. Sipser M., Introduction to the Theory of Computation, Cengage Learning 3rd ed. (2013).

Course Code: **CSE361C**	**Artificial Intelligence and Machine Learning**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Identify and pose classification and regression problems.
- Implement various machine learning algorithms using different libraries to gain hands-on experience and enhance your understanding of the field.
- Compare and study the performance of various algorithms
- Work on a machine learning project from data organization to selection of proper ML algorithm for problem solving.

Unit – I **(12)**

Approaches to AI: Turing Test and Rational Agent Approaches; State Space Representation of Problems, Heuristic Search Techniques, Game Playing, Min-Max Search, Alpha Beta Cutoff Procedures.
Neural Networks Basics: Introduction to biological neural networks, Artificial neural networks (ANN). Analogy between biological and artificial neural networks, Neuron as a basic building element of an ANN, Activation functions. Perceptron. Learning with a perceptron, Limitations of a perceptron, Multilayer neural networks, Learning with a multilayer perceptron, Gradient Descent and its types, Backpropagation algorithm.

Unit – II **(8)**

Inductive learning algorithms, Categories of inductive learning algorithms, Rule extraction with inductive learning algorithms, ID3 algorithm, AQ algorithm, Applications of inductive learning algorithms, Decision Trees.

Unit – III **(9)**

Introduction to Machine Learning. Types of Learning. Generative and Discriminative Algorithms.
Review of basic concepts of Probability Theory, Linear Algebra and Optimizations.
Introduction to machine learning libraries – scikit learn, pandas, numpy, matplotlib, seaborn, tensorflow, keras, pytorch.

Unit – IV **(10)**

Linear Regression, Logistic Regression, SoftMax Regression, Bayes Classifier and Naïve Bayes Classifier, Support Vector Machines, K-Nearest Neighbors, Ensemble Techniques, Bias-variance trade-off, overfitting and underfitting, training, validation and test split, Cross validation, Regularization, Early stopping, Dataset augmentation, Parameter sharing and tying, Injecting noise at input, Dropout.

Unit – V **(9)**

Clustering, Euclidean and Mahalanobis Distances, K-Means and k-Means++ Algorithm, Feature selection Techniques, Eigenvalues and Eigenvectors, PCA (Principal components analysis), ICA (Independent components analysis), LDA (Linear discriminant analysis).

Textbooks:

1. Artificial Intelligence: A Modern Approach by Stuart Russell
2. *Machine Learning. Tom Mitchell*. First Edition, McGraw- Hill, 1997.
3. *Introduction to Machine Learning*, Edition 2, by Ethem Alpaydin
4. Pattern recognition and machine learning by Christopher Bishop
5. Hands-On Machine Learning with Scikit-Learn and TensorFlow, O'Reilly, Aurélien Géron

Reference Books:

1. AI and Machine Learning for Coders by Laurence Moroney.
2. The Hundred-Page Machine Learning Book by Andriy Burkov.
3. Machine Learning with Python Cookbook

Online Resources:

1. https://www.coursera.org/specializations/machine-learning
2. https://course.fast.ai

Course Code: **CSE362C**	**Embedded Systems and Internet of Things**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs): Upon completion of the course, students will be able to:

1. Explain the fundamental concepts of embedded systems, their architecture, and design constraints.
2. Design and implement embedded systems using microcontrollers and appropriate interfacing techniques.
3. Analyze IoT architectures, hardware, and operating systems for developing IoT solutions.
4. Apply IoT communication protocols and wireless technologies for device connectivity and data transfer.
5. Develop IoT applications and analyze IoT data using cloud platforms and data analytics tools.

Unit – I (8)

Overview: Definition, characteristics, and applications of embedded systems.
Embedded System Design: Design constraints, hardware-software co-design, real-time systems.
Microcontrollers and Microprocessors: Architecture of 8051, ARM, and PIC microcontrollers.
Embedded Programming: C programming for embedded systems, interrupt handling, and timers.
Case Studies: Embedded systems in consumer electronics, automotive, and medical devices.

Unit – II (10)

Embedded Hardware: Memory types (RAM, ROM, Flash), I/O interfacing, GPIO.
Sensors and Actuators: Temperature, gas, IR, and proximity sensors; DC motors, stepper motors.
Communication Interfaces: UART, SPI, I2C, CAN protocols.
Power Management: Low-power design techniques, battery management for embedded systems.

Unit – III (10)

IoT Concepts: Definition, characteristics, and challenges of IoT.
IoT Architecture: Layers (perception, network, application), edge and cloud computing.
IoT Hardware: IoT devices (Raspberry Pi, Arduino), sensors, and gateways.
IoT Operating Systems: Introduction to RTOS, TinyOS, Contiki.

Unit – IV (9)

IoT Communication Models: Device-to-device, device-to-cloud, device-to-gateway.
Protocols: MQTT, CoAP, HTTP/REST, WebSocket.
Wireless Technologies: Wi-Fi, Bluetooth, Zigbee, LoRa, 6LoWPAN.
Security in IoT: Data encryption, authentication, secure communication.

Unit – V (8)

IoT Applications: Smart homes, smart cities, healthcare, industrial IoT (IIoT).
Cloud Platforms: AWS IoT, Google Cloud IoT, Microsoft Azure IoT.
Data Analytics for IoT: Data collection, storage, and analysis; introduction to edge analytics.
Case Studies: Real-world IoT implementations, scalability, and challenges.

Textbooks:

1. Raj Kamal, Embedded Systems: Architecture, Programming and Design, Tata McGraw Hill, 2nd Edition, 2011.
2. Carl Hamacher, Zvonko Vranesic, Safwat Zaky, Naraig Manjikian, Computer Organization and Embedded Systems, Tata McGraw Hill, 6th Edition, 2012.
3. Muhammad Ali Mazidi, Janice Gillispie Mazidi, Rolin D. McKinlay, The 8051 Microcontroller and Embedded Systems, Pearson Education, 2nd Edition, 2008.

Reference Books:

1. K.V.K.K. Prasad, Embedded Real-Time Systems, Dreamtech Press, 2014.
2. David E. Simon, An Embedded Software Primer, Pearson Education, 2007.
3. Arshdeep Bahga, Vijay Madisetti, Internet of Things: A Hands-On Approach, Universities Press, 2015.
4. Adrian McEwen, Hakim Cassimally, Designing the Internet of Things, Wiley, 2013.

Online Resources:

1. Introduction to the Internet of Things and Embedded Systems by University of California, Irvine (Coursera): https://www.coursera.org/learn/iot
2. The Arduino Platform and C Programming (Coursera): https://www.coursera.org/learn/arduino-platform
3. IoT Programming and Big Data (edX): https://www.edx.org/course/iot-programming-big-data
4. Official MQTT Documentation: https://mqtt.org/
5. AWS IoT Developer Guide: https://docs.aws.amazon.com/iot/

Course Code: **CSE363C**	**Java Programming**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To demonstrate basic problem-solving skills: analyzing problems, modelling a problem as a system of objects and implementing models in Java.
- To understand the basic concepts of Java and Java flow control.
- To create Java technology applications that leverage the object-oriented features of the Java language, such as encapsulation, inheritance, and polymorphism.
- To Implement error-handling techniques using exception handling and creating high performance multi-threaded applications.
- To Implement input/output (I/O) functionality to read from and write to data and text files and understand advanced I/O streams.
- To Perform multiple operations on database tables using both JDBC and JPA technology and Java Collections frameworks.

Unit – I **(8)**

Introduction to Java

Features of Java, OOPs concepts, Java Virtual Machine, Reflection bytecodes, Bytecode interpretation, Data Types, variables, arrays, expressions, operators, and control structures in Java Objects and classes.
Java Classes – Abstract, Static, Inner and Wrapper classes, Packages, Interfaces, *This*, *Super*, Access control.
Java Flow Control: Java if...else, Java switch Statement, Java for Loop, Java for-each Loop, Java while Loop, Java break Statement, Java continue Statement.

Unit – II **(10)**

Java Exception Handling and Multithreading

Java Exceptions: Java Exception Handling, Java try... catch, Java throw and throws, Java catch Multiple Exceptions, Java Annotations Types.
Multithreading: Introduction, Thread Creations, Thread Life Cycle, Life Cycle, Methods, Java Synchronization methods, User-defined packages.

Unit – III **(9)**

Java List and I/O Streams

String classes, methods, operations on Strings and 1-D Arrays, 2-D and Jagged Arrays and its operations.
Java Collections Framework, Java Collection Interface, Java List Interface, Java Array List, Java Vector, Java Stack.
Introduction to Byte-oriented and Character-oriented streams, Java I/O Streams, and Java Reader/Writer.

Unit – IV **(9)**

Database Applications with JDBC and Java Persistence API

Database Applications with JDBC: Defining the layout of the JDBC API - Connecting to a database by using a JDBC driver – Submitting queries and getting results from the database - Specifying JDBC driver information externally.
Java Persistence API: JPA architecture, ORM Components. Performing CRUD operations using the JDBC API. JPA installation, Java Persistence Query language, Creating JPA entities, Advanced mappings.

Unit – V **(9)**

Java GUI Programming using JavaFX

JavaFX Basics, Panes, UI Controls, and Shapes, Property Binding, Common Properties and Methods for Nodes, Colour Class, Font Class, Image and ImageView Classes, Panes and Shapes class, Adding Functionality, Tasks, Deleting Tasks, Save Data to File, Read Data from File.

Textbooks:

1. *"Core Java, Volume I: Fundamentals, 12e,"* Cay S. Horstmann, Addison-Wesley.
2. *"Core Java: Advanced Features, Volume 2, 11e,"* Cay S. Horstmann, Addison-Wesley.

Reference Books:

1. *"Java: The Complete Reference 12e*," Herbert Schildt, McGraw Hill.
2. *"Java 17 Programming,"* Nick Samoylov, Packt Publishing Limited.
3. *"Head First Java: A Brain-Friendly Guide 3e,"* Kathy Sierra, Bert Bates, and Trisha Gee, O'Reilly Media.
4. *"Learning Java: An Introduction to Real-World Programming with Java, 5e,"* Marc Loy, Patrick Niemeyer, and Daniel Leuck, O'Reilly Media.
5. *"Test-Driven Java Development, 2e,"* Viktor Farcic, Alex Garcia, Packt Publishing.

Online Resources:

1. https://www.udemy.com/course/java-for-beginners-learn-all-the-basics-of-java/
2. https://www.udemy.com/course/ocp11_from_oca8/
3. https://www.coursera.org/specializations/java-fullstack

Course Code: **CSE364C**	**Artificial Intelligence and Machine Learning Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- To understand the concepts of Artificial intelligence and machine learning.
- To understand and practice python.
- To understand and practice various classifiers like SVM, decision trees and neural networks.

List of Experiments

1. Implementing linear regression for House Price Prediction.
2. Implementing linear regression for Heart Disease Prediction.
3. Implementing logistic regression on Diabetes Dataset.
4. Implementing Support vector classifier on Breast cancer DATASET.
5. Implementing Naive Bayes on MNIST DATASET.
6. Implementing KNN on IRIS DATASET.
7. Implementing feed forward neural network on MNIST DATASET.
8. Implementing feed forward neural network on FMNIST DATASET.
9. Implementing feed forward neural network on CIFAR DATASET.
10. Implementing Decision detection on FRAUD Detection.

Online Resources:

1. https://www.kaggle.com
2. https://www.geeksforgeeks.org
3. https://scikit-learn.org
4. https://towardsdatascience.com
5. https://www.tensorflow.org
6. https://pytorch.org

Course Code: **CSE365C**	**Embedded Systems and Internet of Things Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):
Upon successful completion of this lab, students will be able to:

- Develop and test embedded C programs for 8051 microcontrollers using Keil µVision for basic hardware control (e.g., LED blinking, UART communication).
- Interface various sensors and peripherals (e.g., DHT11, MPU6050, SD card module) with microcontrollers and Arduino using UART, I2C, and SPI protocols.
- Implement wireless communication and control using IoT platforms like NodeMCU, integrating with apps (e.g., Blynk) for smart control applications.
- Demonstrate the ability to publish, subscribe, and manage real-time data using MQTT protocol in IoT systems including NodeMCU and Raspberry Pi.
- Analyze and implement cloud-based IoT solutions using platforms like AWS IoT and Raspberry Pi with Python to log and process sensor data.

List of Experiments

1. Write an embedded C program to blink an LED connected to an 8051 microcontroller using Keil uVision.
2. Interface a DHT11 sensor with Arduino to display temperature and humidity on an LCD.
3. Implement serial communication between an 8051 microcontroller and a PC using UART.
4. Interface an I2C-based sensor (e.g., MPU6050) with Arduino and display data on the serial monitor.
5. Connect an SPI-based device (e.g., SD card module) to Arduino and perform read/write operations.
6. Use NodeMCU to control an LED remotely via Wi-Fi using the Blynk app.
7. Set up an MQTT broker (Mosquitto) and publish/subscribe sensor data using NodeMCU.
8. Create a smart home system to control appliances using Raspberry Pi and MQTT.
9. Connect an IoT device (e.g., ESP8266) to AWS IoT Core and log sensor data to the cloud.

Lab Requirements: Hardware: 8051 development boards, Arduino, Raspberry Pi, NodeMCU, sensors (DHT11, HC-SR04, MPU6050), actuators (DC motor, relay modules).

Software Requirements: Keil uVision, Arduino IDE, Python, Node-RED, Mosquitto, AWS IoT SDK

Course Code: **CSE366C**	**Java Programming Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- To develop basic Java applications,
- To create classes, objects and apply Inheritance,
- To create Packages and build applications using default packages,
- To manage Exceptions and develop multithreaded applications, and
- To design and develop platform independent applications using a variety of component-based frameworks.

List of Experiments

1. A simple program to implement constructor in java.
2. To implement Class concept in java.
3. To implement method overloading by using static method in java.
4. To implement method overloading in a single class in java.
5. To implement simple inheritance in java.
6. To implement method overriding in java.
7. To call base class constructor using super keyword in java.
8. To call base class method using super keyword in java.
9. To implement run time polymorphism by applying dynamic dispatch method in method overriding.
10. To implement the concepts of array, stack, linked list, and queue using Java Collection Frameworks.
11. To implement Abstract class.
12. To implement Interface.
13. To implement multithreading by extending Thread class.
14. To write a Java program that correctly implements the producer – consumer problem using the concept of inter-thread communication.
15. To fetch data from the database using (JDBC-ODBC bridge).
16. To implement Java I/O streams.
17. To implement Java GUI Programming paradigm using JavaFX

Course Code: **CSE360A**	**Aptitude and Reasoning**	Credits: **-** L – **2** P – **0**

Course Outcomes (COs):

- Analyze and interpret data presented in various visual representations.
- Develop proficiency in estimating values for practical problem-solving scenarios.
- Utilize geometry to analyze and solve real-world problems.
- Understand and apply basic statistical concepts to analyze data.

Unit – I **(10)**

Quantitative Aptitude:
Data interpretation: data graphs (bar graphs, pie charts, and other graphs representing data), 2- and 3-dimensional plots, maps, and tables Numerical computation and estimation: ratios, percentages, powers, exponents and logarithms, permutations and combinations, and series Mensuration and geometry Elementary statistics and probability.

Unit – II **(10)**

Analytical Aptitude:
Logic: deduction and induction, Analogy, Numerical relations, and reasoning.

Unit – III **(10)**

Spatial Aptitude:
Transformation of shapes: translation, rotation, scaling, mirroring, assembling, and grouping Paper folding, cutting, and patterns in 2 and 3 dimensions

Textbooks:

1. Quantitative Aptitude for Competitive Examinations" by R.S. Aggarwal
2. How to Prepare for Quantitative Aptitude for the CAT" by Arun Sharma

Online Resources:

1. Brilliant.org Logic and Reasoning: https://brilliant.org/
2. Khan Academy: https://www.khanacademy.org/
3. Mathway: https://www.mathway.com/

Course Code: **CSE367C**	**Project-II**	Credits: **03** L – **0** P – **6**

Description:

- During the sixth semester, students have to take Project-II (Mini Project) of three credits.
- Students need to identify area of their interest in which they would opt (not necessarily) their Project-III and Project-IV in later semesters.
- Students have to do extensive literature survey of their field of interest and also learn certain tools / software required.
- Emphasis should be on problem solving, innovation, and design.
- Deliverables: product, project report, codebase, user manual, demonstration etc.

SEMESTER VII

Course Code: **CSE410C**	**Compiler Design**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Define and analyze various phases involved in designing of compiler
- Compare and contrast between bottom up and top down parsing
- To learn basic data structures used in compiler construction such as abstract syntax trees, symbol tables, three address code and stack machines.
- To understand the principles and techniques used to perform translation and the fundamental concepts of translator construction.
- Acquired the skills to understand, develop, and analyze recognizers for programming languages.

Unit – I **(8)**

Introduction: Introduction to compilers, Phases of Compiler, Bootstrapping and Cross compiler
Lexical Analysis: The role of the Lexical analyzer, Input Buffering, Specification of Tokens, Recognition of Tokens, Compiler construction tools, Classification of grammars, Context-free grammars.

Unit – II **(12)**

Syntax Analysis: Top Down Parsing: Brute Force Parsing, Recursive Descent Parsing, Non Recursive, Predictive Parsing, Bottom-Up parsing -Shift Reduce Parsing Operator-Precedence Parsing.
LR Parsers: LR(0) SLR(1), CLR(1) and LALR(1).

Unit – III **(9)**

Syntax Directed Translation: Syntax Directed Definitions, Evaluation Orders for SDD's, Applications of Syntax-Directed Translation.
Intermediate Code Generation: Syntax trees**,** Three Address codes, Types and Declarations, Translation of Arithmetic Expressions, Back patching for Boolean Expressions, and Flow of Control.

Unit – IV **(10)**

Code Optimization: The Principal Sources of Optimization, Basic Blocks and Flow Graphs, Loop optimization, Optimization of Basic Blocks, Introduction to Data-Flow Analysis, DAG.
Code Generation: Issues in designing a code Generator, The Target Language, A Simple Code Generator, Peephole Optimization, Register allocation, and Assignment.

Unit – V **(6)**

Static, dynamic, and heap allocation, Implementation of simple stack allocation scheme, Symbol tables, data structure used in symbol table, Language facilities for dynamic storage allocation,
Error Handling: Classification of Errors, Error-Recovery Strategies

Textbooks:

1. Aho, Ullman & Ravi Sethi, "*Principles of Compiler Design*", Pearson Education

Reference Books:

1. Tremblay, et. al., "*The Theory and Practice of Compiler Writing*", McGraw Hill, New York
2. Holub, "*Compiler Design in C*", PHI
3. Andrew L. Appel, "*Modern Compiler Implementation in C*", Delhi Foundation Books.

Course Code: **CSE411C**	**Network Security**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Understand the fundamentals of cryptography.
- Understand how the key distribution is done between the communicating parties.
- Acquire knowledge on standard algorithms used to provide confidentiality, Integrity, authenticity, and data availability.
- Understand the usage of encryption techniques to secure the data in transit across the data networks Configure the firewall system to secure the networks from external security threats.
- Learn about the latest developments in the field of Network Security.

Unit – I **(10)**

Introduction to cryptography, Elementary Ciphers (Substitution, Transposition, and their properties), Product Ciphers, Data Encryption Standard (DES), Design and analysis, Strength of DES, AES, RC4.

Unit – II **(8)**

Modulo Arithmetic, Modular Inverse, GCD, Extended Euclid Algorithm, Fermat's Little Theorem, Euler Phi-Function, Euler's theorem, Chinese remainder theorem.

Unit – III **(8)**

Introduction to Public Key Cryptography, Principles, Public-Key Cryptography Algorithms (RSA), Approaches to Message: Authentication, Secure Hash Functions, SHA – 512, Message Authentication Codes (MAC). Diffie-Hellman Key Exchange.

Unit – IV **(7)**

Network Access Control, Web Security Considerations, Secure Socket Layer / Transport Layer Security (SSL/TLS), Pretty Good Privacy (PGP), Secure Shell (SSH) Application.

Unit – V **(7)**

Wireless security, Firewalls – types, configuration, and properties. Introduction to: Quantum Cryptography, Blockchain, Cryptocurrency, IoT Security.

Textbooks:

1. Behrouz A. Forouzan, "*Cryptography and Network Security*".
2. William Stallings, "*Cryptography and Network Security – Principles and Practices*".

Reference Books:

1. Charlie Kaufman, Radia Perlman and Mike Speciner, "*Network Security: Private Communication in a Public World*".
2. Bruce Schneier, "*Applied Cryptography*", John Wiley & Sons Inc, 2001.
3. Atul Kahate, "*Cryptography and Network Security*", Tata McGraw Hill, 2003.
4. Charles B. Pfleeger, Shari Lawrence Pfleeger, "*Security in Computing*", Third Edition, Pearson Education, 2003.
5. *Networking Essentials*, by William. S. Stallings.

Course Code: **CSE412C**	**Industrial Management and Entrepreneurship Development**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Analyze structures and control, apply planning and decision-making for improved organizational effectiveness.
- Gain insights into organizational behaviour, including individual factors and group dynamics, enhancing workplace effectiveness.
- Understand leadership styles, power strategies, analyze organizational structures, adapt to change, and excel in HR functions, including recruitment, training, and performance appraisal.
- Understand entrepreneurship, its characteristics, importance in economic growth, foster an entrepreneurial mindset, and skillfully identify business opportunities through research and planning.
- Evaluate legal structures, funding options, and demonstrate proficiency in start-up finance, marketing, sales, CRM, branding, operations, resource allocation, and logistics.

Unit – I **(6)**

Industrial Management:
Definition and scope, Importance; Organizational Structure and Design; Span of control and hierarchy; Centralization vs. decentralization. Planning and Decision-Making.

Unit – II **(10)**

Organizational Behaviour:
Definition and scope of organizational behaviour. Individual Behaviour: Personality and its impact on behaviour. Perception and its role in decision-making. Motivation and job satisfaction. Attitudes and emotions in the workplace. Stress and its management. Group Behaviour: Groups and teams. Group dynamics, Communication in organizations. Conflict resolution and negotiation.

Unit – III **(8)**

Leadership and Management:
Leadership styles and theories. power and influence. Organizational Structure and Design. Organizational change and adaptation. Human Resource Management. Recruitment and selection. Training and development. Performance appraisal and feedback.

Unit – IV **(8)**

Entrepreneurship Development:
Definition and characteristics of an entrepreneur. Importance of entrepreneurship in the economy. Entrepreneurial mindset and creativity. Identifying business opportunities. Market research and feasibility analysis. Business planning and model development.

Unit – V **(10)**

Financial, Marketing and Operations Aspects of Entrepreneurship:
Legal structures for start-ups (e.g., sole proprietorship, LLC, corporation) Funding options (e.g., bootstrapping, angel investors, venture capital). Financial management and budgeting. Marketing concepts and strategies. Sales techniques and customer relationship management. Branding and promotion. Operations planning and management. Resource allocation and utilization. Supply chain and logistics.

Textbooks:

1. Robbins, Stephen P., and Mary Coulter. Management. Pearson, 2017.
2. Daft, Richard L. Organization Theory and Design. Cengage Learning, 2018.
3. Hisrich, Robert D., Michael P. Peters, and Dean A. Shepherd. Entrepreneurship. McGraw-Hill Education, 2019.
4. Kuratko, Donald F., and Richard M. Hodgetts. Entrepreneurship: Theory, Process, and Practice. Cengage Learning, 2020.

Reference Books:

1. Mintzberg, Henry. Structure in Fives: Designing Effective Organizations. Prentice-Hall, 1993.
2. Kotter, John P. Leading Change. Harvard Business Review Press, 2012.
3. Dessler, Gary. Human Resource Management. Pearson, 2017.
4. Blank, Steve, and Bob Dorf. The Startup Owner's Manual: The Step-By-Step Guide for Building a Great Company. K&S Ranch, 2012.
5. Osterwalder, Alexander, and Yves Pigneur. Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers. Wiley, 2010.
6. Ries, Eric. The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Business, 2011.

Course Code: **CSE413C**	**Project-III**	Credits: **04** L – **0** P – **8**

Description:

- During the seventh semester, students have to take Project-III (Major-Project) of 4 credits.
- In Project-III, students identify a problem in their field of interest and also try to do at least partial implementation of their stated problem. They may extend their Project-III to Project-IV in the eighth semester.
- Emphasis should be on problem solving, innovation, and design.
- Deliverables: product, project report, codebase, user manual, demonstration etc.

SEMESTER VIII

Course Code: **CSE460C**	**Internship / Industrial Training**	Credits: **02** L – **0** P – **0**

Course Outcomes (COs):

- Apply engineering knowledge in solving real-life problems.
- Attain new skills and be aware of the state-of-the-art in engineering disciplines of their own interest.
- Describe use of advanced tools and techniques encountered during training and visit.
- Acquire practical skills, organizational skills, Communication skills, lifelong learning skills, professional awareness and experience working on projects and alongside industry experts.
- Write a technical report that follows an established structure and give oral presentations with focus on the results and a credible work procedure.

Guidelines:

1. As per institution's guidelines, Industrial Training has to be done for at least 4 weeks, at any time during the degree.
2. The student must follow the instructions given by the Department.
3. The purpose of the Industrial Training is to develop the work process being performed and apprise them of the industry problems.
4. During the training, students should be given practical problems by the industry in which they are undergoing training. In case the industry does not give them the problems, the students will themselves formulate problems and carry out detailed study on them and recommend the optimum solution based on their theory knowledge.
5. Submission of Final Report - Each student must submit a Final Report and appear for viva-voce examination on industrial training at the end of the industrial training period on the date and time announced by the Teacher In-charge.
6. The Internship report must include the following:
    a. The basic history/introduction of the industry.
    b. The software and hardware used.
    c. The sequence of operations followed/ systems introduced for the project development.
    d. The formulation of practical problems.
    e. Data required in formulating the problems.
    f. Analysis of the data, steps required and commands used in industry.
    g. Certificate from the industry for the period of training undergone.
7. The student would be evaluated through Report and Viva-voce.

Course Code: **CSE461C**	**Project-IV**	Credits: **10** L – **0** P – **20**

Description:

- In the eighth semester, students have to take Project-IV (Major-Project) of 10 credits.
- Students may extend their Project-III to Project-IV so that they can be strengthen in a particular field which further helps in defining their field of interest / specialization.
- Emphasis should be on problem solving, innovation, and design.
- Deliverables: product, project report, codebase, user manual, demonstration etc.
- An external examiner along with a departmental committee will evaluate the performance of the student in the Project-IV.

DISCIPLINE CENTRIC ELECTIVES

Course Code: **CSE001E**	**Advanced Computer Architecture**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Provide knowledge on instruction level parallelism
- Describe the hardware and software level parallelism to enhance the computing performance
- Describe the different operations of performance enhancement such as pipelines, dynamic scheduling, branch prediction, caches, and vector processors
- Design the models for arithmetical calculation and to evaluate the CPU performance
- Understand the memory hierarchy model for enhanced computing performance

Unit – I **(10)**

Overview of Parallel Processing and Pipelining Processing, comparison of uni-processors and parallel processors. Evolution of parallel processors, future trends and their architecture Architectural Classification, Applications of parallel processing, Instruction level Parallelism and Thread Level Parallelism, Explicitly Parallel Instruction Computing (EPIC) Architecture.

Unit – II **(9)**

Principles and implementation of Pipelining, Classification of pipelining processors, Pipeline Architecture, Study and comparison of processors with and without pipelining. General pipelining reservation table, Pipelining hazards and resolving techniques, Data buffering techniques, Job sequencing and Collision.

Unit – III **(7)**

Study and comparison of Vector and array processors, Vector and Array Processor, Basic vector architecture, Issues in Vector Processing, Vector performance modeling, vectorizers and optimizers.

Unit – IV **(9)**

Microprocessor Architectures, study and comparison of Loosely and Tightly coupled multiprocessors. Processor characteristics of multiprocessors, Inter Processor communication network, Time shared bus, Crossbar switch, Multiport Memory Model, Memory contention and arbitration techniques.

Unit – V **(9)**

Study of Architecture of Multithreaded processors, Latency hiding techniques, Principles of multithreading, Issues, and solutions**. Parallel Programming Techniques:** Message passing program development, Synchronous and asynchronous message passing, Message passing parallel programming, Shared Memory Programming.

Textbooks:

1. Kai Hwang, Faye A. Briggs, "Computer Architecture and Parallel Processing" McGraw-Hill international Edition
2. Kai Hwang, "Advanced Computer Architecture", Tata McGraw-Hill

Reference Books:

1. V.Rajaraman, L Sivaram Murthy, "Parallel Computers", PHI.
2. William Stallings, "Computer Organization and Architecture, Designing for performance" Prentice Hall, Sixth edition.
3. Kai Hwang, Scalable Parallel Computing.
4. Harrold Stone, High performance computer Architecture.
5. Richard Y. Kain, Advanced Computer Architecture

Course Code: **CSE002E**	**Advanced Java**	Credits: **03** L – **2** P – **2**

Course Outcomes (COs):

- To implement abstract classes and abstract methods, cast objects in an inheritance hierarchy, and how to apply good inheritance design guidelines in your Java programs,
- To design Java programs that use interfaces and write code that creates and uses interfaces,
- To create GUI Applications with JavaFX,
- To write objects to files and read objects from files using object serialization,
- To use Java collections and generics to enhance your program robustness and flexibility, and
- To incorporate several common algorithms and to find out how to discern which algorithm patterns will best solve your programming challenges.
- Apply Model-View-Controller architecture to build complex client-server applications.

Unit – I **(8)**

Java Collections Framework

Utility Methods for Arrays, Using Scanner Regular Expression, Input/Output Operation in Java (java.io Package), Streams and the new I/O Capabilities, Understanding Streams, The Classes for Input and Output, The Standard Streams, Working with File Object, File I/O Basics, Reading and Writing to Files, Serializing Objects.
Java Collections Framework, Java Collection Interface, Java List Interface, Java Array List, Java Vector, Java Stack.

Unit – II **(10)**

Java Generics and Associative Collections

Java Generics and their use, Declaring a Generic Stack, Generics with Subtypes – wildcards. Bounded generics with super and extend, Generic Methods, Bounded Wildcards, Bounded generics with super and extends, Nested and Inner Classes.
Associative Collections – The Map Interface, The Treemap Class, The Hashmap Class, Binary Trees.

Unit – III **(10)**

Java GUI Programming using JavaFX

JavaFX Basics, Panes, UI Controls, and Shapes, Property Binding, Common Properties and Methods for Nodes, Colour Class, Font Class, Image and ImageView Classes, Panes and Shapes class, Adding Functionality, Tasks, Deleting Tasks, Save Data to File, Read Data from File.

Unit – IV **(9)**

Event Handling in Java

Event-Driven Programming in Java, Event-Handling Process, Event Handling Mechanism, The Delegation Model of Event Handling, Event Classes, Event Sources, Event Listeners, Adapter Classes as Helper Classes in Event Handling.

Unit – V **(10)**

Advanced JDBC Programming and Java Web Frameworks

Saving to a Database – Inserting database rows. Using PreparedStatement to do an insert, Converting enum values to strings. Updating Databases. Loading Data from Databases – Instantiating objects using information from a database, retrieving data via column names, calling one constructor from another in Java, Loading enum types from strings.
Spring MVC - Overview of Spring, Spring Architecture, bean life cycle, XML Configuration on Spring, aspect-oriented Spring, Managing Database, Managing Transactions.

Textbooks:

1. *"Java 17 Programming,"* Nick Samoylov, Packt Publishing Limited.
2. *"Database Programming with JDBC & Java: Developing Multi-Tier Applications,"* George Reese, O'Reilly Media.

Reference Books:

1. *"Core Java: Advanced Features, Volume 2, 11e,"* Cay S. Horstmann, Addison-Wesley.
2. *"Learning Java: An Introduction to Real-World Programming with Java, 5e,"* Marc Loy, Patrick Niemeyer, and Daniel Leuck, O'Reilly Media.
3. *"Test-Driven Java Development, 2e,"* Viktor Farcic, Alex Garcia, Packt Publishing.
4. *"Learn JavaFX 17: Building User Experience and Interfaces with Java, 2e,"* Kishori Sharan, and Peter Späth, Apress.

List of Experiments

1. To implement basic and advanced Java Programs on Java Collections Framework including -
    a. To get the Maximum Element from a Vector,
    b. To perform Binary Search on Java Vector,
    c. To get Elements of a LinkedList,
    d. To convert an Array into Collection in Java,
    e. To Iterate HashMap in Java,
    f. To use Enumeration to Display Elements of Hashtable in Java,
    g. To implement the Java Collections Framework methods like -
        i. *collections.reverse()*,
        ii. *collections.shuffle()*,
        iii. Hashtable *keySet()*, etc.

2. To implement basic and advanced Java Programs on Java GUI Programming using JavaFX
    a. To create a JavaFX application that displays a "Hello, JavaFX!" message in a window. Your system's Java and JavaFX versions are displayed as well,
    b. To write a JavaFX application that creates a button. On clicking the button, display an alert,
    c. To write a JavaFX application with multiple components (buttons, labels, text fields) arranged in a VBox or HBox layout.
    d. To create a JavaFX application with a text input field and a button. When the button is clicked, display the text entered in the input field in a label.
    e. To write a JavaFX application that loads an image and displays it in a JavaFX window,
    f. To create a JavaFX application that creates a basic calculator application that can add, subtract, multiply, and divide two numbers entered by the user,
    g. To write a JavaFX application with a circle that changes its color when clicked,
    h. To write a JavaFX application that animates a shape (e.g., a circle) to move across the window in response to a button click.

3. To perform database operations in Java | SQL CREATE, INSERT, UPDATE, DELETE and SELECT

4. To execute multiple SQL Commands on a database simultaneously in JDBC.

5. Create an application to keep records and retrieve records of students. The record includes student ID, enrollment number, semester, CGPA, etc. - use MVC architecture.

Course Code: **CSE003E**	**Advanced JavaScript**	Credits: **03** L – **2** P – **2**

Course Outcomes (COs):

- Understanding and applying the software engineering lifecycle models by demonstrating competence in communication, planning, analysis, design, construction, and deployment
- Translate a requirements specification into an implementable design, following a structured and organized process.
- Defining the basic concepts and importance of Software project planning concepts like cost estimation, and scheduling.
- Applying different testing and debugging techniques and analyzing their effectiveness.
- Defining software maintenance concepts, software quality and reliability on the basis of international quality standards.

Unit – I **(8)**

JavaScript Foundation
Scope in JavaScript, the var Keyword, Scope in Nested Functions, the let Keyword.
Advanced Objects – Object-Oriented Programming, Prototyping, Objects in JavaScript, prototyping in JavaScript, Composing Prototypes, ES6 Classes and Static Methods, Inheritance. Maps, Map vs Object, Sets. Nodejs

Unit – II **(10)**

Advanced functions and arrays in JavaScript
Advanced Functions – Function Declarations, Function Expressions, Default Values, Flexible Arguments, Closures, Immediately-Invoked Function Expressions (IIFEs), Callbacks, Recursion.
Advanced Arrays – Arrays, Arrays Methods, Iterators and Arrow Functions, Iterators, Arrow Functions, Static Array Methods, Array Prototype Methods.

Unit – III **(10)**

Asynchronous JavaScript
ES6 Modules and Asynchronous JavaScript Asynchronous vs Synchronous Execution, Closures, Callbacks, Promises, promises chaining, Fetch API, Async Await ES8 - Async Await, ES9 (ES2018), ES9 (ES2018) – Async,
Job Queue, Parallel, Sequence and Race.

Unit – IV **(9)**

JSON and jQuery
JSON – Review of Object Literals, Arrays, Objects, Arrays in Objects, Objects in Arrays, JSON Syntax, JSON Parsers.
jQuery – Running code when the document is ready Click events Using jQuery's slideToggle() method Supporting JavaScript disabled users, Using jQuery's hover() method Using jQuery's animate() method, Adding an animation to reveal hidden content Targeting the proper div: traversing the document Swapping the button image with jQuery.

Unit – V **(10)**

Regular Expressions in JavaScript
JavaScript's Regular Expression Methods, Flags, String Methods, Regular Expression Syntax, Start and End (^ $), Number of Occurrences (? + * {}), Common Characters (. d D w W s S), Grouping ([]), Negation (^), Subpatterns (()), Alternatives (|), Escape Character (). Form Validation with Regular Expressions, Cleaning Up Form Entries.

Textbooks:

1. *"JavaScript and JQuery: Interactive Front-End Web Development,"* Jon Duckett, Wiley.
2. *"Mastering HTML, CSS & JavaScript Web Publishing, 2022 Edition"* Jennifer Kyrnin Laura Lemay, and Rafe Colburn, BPB publishers.

Reference Books:

1. *"React and React Native: Build cross-platform JavaScript applications with native power for the web, desktop, and mobile, 4th Edition,"* Adam Boduch, Roy Derks, and Mikhail Sakhniuk, Packt Publishing Limited.
2. *"JavaScript from Beginner to Professional, 2021 Edition,"* Laurence Lars Svekis, Maaike van Putten, and Rob Percival, Packt Publishing.

Online Resources:

1. https://codedamn.com/learn/advanced-practical-javascript
2. https://www.udemy.com/course/full-practical-javascript-course-2021-beginner-to-expert/
3. https://www.coursera.org/learn/javascript

List of Experiments

1. Basic JavaScript functions usage HTML Head Tag, Including external JS file in HTML
2. JavaScript input and output using Document.write
3. Creating alert boxes and comments using JavaScript
4. Usage and application of variables and operators in JavaScript
5. Usage and applications of *Math* Functions in JavaScript
6. Using conditional *if else* and conditional *switch case* in JS
7. Implementing Exception Handling and Error Handling in JS
8. Form Validations using JS
9. Implementing object oriented principles of JS - Classes and Objects, inheritance etc.
10. Implementing the basics of regular expressions in JS including - the ^ and $ metacharacters, character classes, character classes' shorthands, etc.
11. Implementing advanced concepts of regular expressions in JS including - Quantifiers: Ranges, Quantifiers: Optional (?), Quantifiers: One or more (+), Quantifiers: Zero or more (*), Quantifiers: Greediness and Laziness
12. Implementing basic jQuery exercises including:
    a. To test if jQuery is loaded
    b. To scroll to the top of the page with jQuery
    c. To disable the right click menu in the html page using jquery
    d. To disable/enable the form submit button
    e. To disable the submit button until the visitor has clicked a check box
    f. To fix broken images automatically
    g. To find the total width of an element (including width, padding, and border) in jQuery.

Course Code: **CSE004E**	**Agile Software Development**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Interpret the concept of agile software engineering and its advantages in software development.
- Analyze the core practices behind several specific agile methodologies.
- Identify the roles and responsibilities in agile projects and their difference from projects following traditional methodologies.
- Access implications of functional testing, unit testing, and continuous integration.
- Determine the role of design principles in agile software design.
- Make use of various tools available to agile teams to facilitate the project.

Unit – I **(8)**

Introduction: Need of Agile software development, agile context– Manifesto, Principles, Methods, Values, Roles, Artifacts, Stakeholders, and challenges. Business benefits of software agility.

Unit – II **(8)**

Project Planning: Recognizing the structure of an agile team– Programmers, Managers, Customers. User stories– Definition, Characteristics and content. Estimation– Planning poker, Prioritizing, and selecting user stories with the customer, projecting team velocity for releases and iterations.

Unit – III **(9)**

Project Design: Fundamentals, Design principles–Single responsibility, Open-closed, Liskov substitution, Dependency-inversion, Interface-segregation.

Unit – IV **(10)**

Design Methodologies: Need of scrum, Scrum practices –Working of scrum, Project velocity, Burn down chart, Sprint backlog, Sprint planning and retrospective, Daily scrum, Scrum roles– Product Owner, Scrum Master, Scrum Team. Extreme Programming- Core principles, values and practices. Kanban, Feature-driven development, Lean software development.

Unit – V **(10)**

Testing: The Agile lifecycle and its impact on testing, Test driven development– Acceptance tests and verifying stories, writing a user acceptance test, Developing effective test suites, Continuous integration, Code refactoring. Risk based testing, Regression tests, Test automation.

Textbooks:

1. Ken Schawber, Mike Beedle, “Agile Software Development with Scrum”, International Edition, Pearson.
2. Robert C. Martin, “Agile Software Development, Principles, Patterns and Practices”, First International Edition, Prentice Hall.
3. Pedro M. Santos, Marco Consolaro, and Alessandro Di Gioia, “Agile Technical Practices Distilled: A learning journey in technical practices and principles of software design”, First edition, Packt Publisher.

Reference Books:

1. Lisa Crispin, Janet Gregory, “Agile Testing: A Practical Guide for Testers and Agile Teams”, International edition, Addison Wesley.
2. Alistair Cockburn, “Agile Software Development: The Cooperative Game”, 2nd Edition, Addison-Wesley.

Course Code: **CSE005E**	**Big Data**	Credits: **04** L – **3** P – **2**

Course Outcomes (COs):

- Understand the concept of Big Data, challenges faced by it and why classical data analysis tools and techniques are no longer adequate
- Get familiar with latest Big Data management tools with main focus on Hadoop ecosystem and Map-reduce.
- Understand different learning algorithms for analysis of Big Data for knowledge extraction
- Model and implement efficient Big Data solutions for various application areas
- Apply analytics on Structured, Unstructured Data.

Unit – I **(9)**

Big Data overview, introduction to the Big Data problem, key enablers for growth of big data, definitions and dimensions of big data, 3Vs and 7Vs of Big Data, current challenges, trends, and applications of Big Data, concept and types of data sets, different sources of big data, different forms of big data.

Unit – II **(7)**

Mining and learning algorithms that deal with large datasets, Supervised learning: Linear/Logistic regression, Decision trees, Naïve Bayes. Unsupervised learning: K-means clustering, Association rule mining.

Unit – III **(10)**

Technologies and tools for big data management, Hadoop: architecture, History of Hadoop, Apache Hadoop, Analyzing Data with Hadoop, Hadoop Streaming, Hadoop Echo System, **Map reduce,** Anatomy of a Map Reduce Job Run, Failures, Job Scheduling Map Reduce Types and Formats, Map Reduce Features.

Unit – IV **(6)**

Big data analytics in industry verticals, Data analytics lifecycle and methodology: discovery, data preparation, model planning, model building, communicate results, operationalize, Big data value chain.

Unit – V **(8)**

Data visualization techniques, Text mining: introduction, areas, process and applications, Web mining: introduction, categories and applications, Features and brief description of Amazon Web Services, BlueMix, Cognos, Big insight.

Textbooks:

1. Big Data: A Revolution That Will Transform How We Live, Work, and Think, by Viktor Mayer-Schönberger, Kenneth Cukier.

Reference Books:

1. Michael Berthold, David J. Hand, "Intelligent Data Analysis", Springer, 2007.
2. Jay Liebowitz, "Big Data and Business Analytics" Auerbach Publications, CRC press (2013).
3. Anand Rajaraman and Jef rey David Ulman, "Mining of Massive Datasets", Cambridge University Press, 2012

List of Practicals

1. Install and configure the Hadoop ecosystem setup with HDFS and MapReduce. Explore HDFS commands to manage files and directories.
2. Implement a Linear regression algorithm for the House Price Prediction task using **Spark MLlib**
3. Implement a Logistic regression algorithm for the classification task using **Spark MLlib**
4. Develop MapReduce programs for data processing tasks:
    a. Implement a word count program.
    b. Develop a MapReduce job to sort data.
5. Write a MapReduce program to implement Matrix Multiplication.
6. Import data from a relational database into HDFS using Apache Sqoop.
7. Gain hands-on experience with **Apache Kafka** by setting up a primary data streaming pipeline:
    a. Create a Kafka topic for data streaming.
    b. Write a simple producer script to send data to the Kafka topic.
    c. Write a consumer script to read and display data from the Kafka topic.
8. Basic data processing with Apache Spark.
    a. Setup Spark and explore the Spark Shell.
    b. Create Resilient Distributed Datasets and apply transformations and actions.
    c. Use the DataFrame API for structured data processing.
9. Integrate Big Data with Tableau or a similar visualization tool.
10. Implement text mining to extract meaningful information from unstructured text data using Hadoop and Python.

Course Code: **CSE006E**	**Computational Ethics Design**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To acquaint students with a fundamental understanding of various ethical theories.
- Explore Artificial Intelligence towards computational design of ethics.
- Introduce student to various approaches in machine ethics.
- Ability to relate ethical concepts to specific engineering problems.
- Establish foundation for engaging in further research in the subject.

Unit – I **(9)**

Introduction to ethics and its types, Metaethics, Normative ethics: Virtue theories, Duty theories, Consequentialist theories, Divine command theory, Applied Ethics: Normative principles in applied ethics, issues in applied ethics.

Unit – II **(9)**

Introduction to artificial intelligence (AI), Foundation of AI, History of AI, AI applications, Intelligent agents and environments, Structure of agents, Weak AI, Strong AI, Ethics of AI

Unit – III **(9)**

Machine ethics: nature, importance and difficulty, top down, bottom up and hybrid approaches, machine manners, ontology agnostic approach, Value Sensitive Design.

Unit – IV **(9)**

Ethically aligned design, The FAST Track Principles, Moral Machine experiment, Ethics aware Software Engineering, Bias in computing.

Unit – V **(9)**

Research trends in computational ethics, Computational models of ethical reasoning: Truth-teller, SIROCCO, JEREMY, MedEthEx, LIDA

Textbooks:

1. W. Wallach and C. Allen, *Moral machines: teaching robots right from wrong*, Oxford University Press.
2. Michael Anderson, Susan Leigh Anderson, *Machine Ethics*, Cambridge University Press.
3. Rocci Luppicini, *Technoethics and the Evolving Knowledge Society: Ethical Issues in Technological Design, Research, Development, and Innovation*, University of Ottawa.
4. S. Russell and P. Norvig, *Artifical Intelligence: A modern Approach*, Prentic Hall.

Reference Books:

1. J. Deigh, *An Introduction to Ethics*, Cambridge University Press.

Online Resources:

2. The IEEE Global Initiative for Ethical Considerations in AI and Autonomous Systems (https://standards.ieee.org/content/ieee-standards/en/initiatives/trustworthy-ai/resources/ethical-considerations.html)
3. The Association for Computing Machinery (ACM) Code of Ethics (https://www.acm.org/code-of-ethics)
4. The European Group on Ethics in Science and New Technologies (EGE) (https://ec.europa.eu/info/departments/european-group-ethics-science-and-new-technologies-ge-ethics_en)

Course Code: **CSE007E**	**Computer Graphics**	Credits: **04** L – **3** P – **2**

Course Outcomes (COs):

- Understand the core concepts of computer graphics.
- Apply the algorithms for 2-D and 3-D geometric transformations.
- Illustrate the techniques used for different area filling algorithms using C/Open GL.
- Apply the algorithms for 2-D clipping using C/Open GL.

Unit – I **(7)**

Introduction to Computer Graphics. Applications and Advantages of Computer Graphics, Graphic Display Devices (CRT, Random and Raster Scan Displays, Flat Panel Display, LCD, TFT, PDP, LED). Graphic Input Devices (Keyboard, Mouse, Trackballs, Space balls, Joy-sticks, Touch-screens).

Unit – II **(10)**

Overview of coordinate systems, Points and Lines, Line Drawing Algorithms (DDA, Bresenham's Line Algorithm). Circle – Generating Algorithms (Basic concepts and properties of circle drawing, Mid-point and Bresenham's circle drawing algorithms). Ellipse Generating Algorithms (Basic concepts and properties of ellipse drawing, Mid-point Ellipse drawing algorithm). Boundary Filling and Flood Filling Algorithms.

Unit – III **(10)**

Introduction to 2D- transformation: Basic transformations (Translation, Rotation, Scaling, Reflection and Shearing). Composite Transformations (Translation, Rotation and Scaling). 2-D Viewing: The viewing pipeline, Window-to-viewport transformation, Clipping operations, Point clipping, Line clipping (Cohen –Sutherland Line Clipping). Polygon clipping (Sutherland-Hodgeman Polygon Clipping), Text clipping.

Unit – IV **(8)**

Introduction to Projections (Parallel Projections and Perspective Projections). Visible-Surface Detection Methods (Z-buffer algorithm, Scan line algorithm, Area subdivision algorithm, Painter's Algorithm).

Unit – V **(10)**

Introduction to Object Rendering, Light Modeling Techniques, Illumination Model, Shading, Flat Shading, Polygon Mesh Shading, Gouraud Shading, Phong Shading. Interactive Picture –Construction Techniques (Basic Positioning Methods, Constraints, Grids, Gravity field, Rubber Band Techniques, Dragging, Painting and Drawing, Inking). Introduction to OpenGL, Basic OpenGL Syntax, Related Libraries, Examples and demos of OpenGL programs.

Textbooks:

1. "Computer Graphics" by Donald Hearn & M. Pauline Baker.

Reference Books:

1. "Principles of Interactive Computer Graphics" by William. M. Newman & Robert. F. Sproull.
2. Steven Harrington."Computer Graphics A Programming Approach" McGraw Hill.
3. James. D. Foley, VanDam "Fundamentals of Interactive Computer Graphics".
4. David F. Frogers & J Alan Adams- "Procedure and elements of Computer graphics".
5. Schaum's Outlines, "Computer Graphics", Second Edition, McGraw-Hill, 2000
6. FS Hill , SM. Kelley, "Computer Graphics using OpenGL" Third Edition Pearson Education, 2007

List of Experiments

1. To draw Rectangle from (100, 200) pixels to (400,500) pixels using in-built functions.
2. To draw a Hexagon centered (100, 100), Circle with center (150,150) pixels and radius 25.
3. To draw a chain of circles and concentric circles.
4. To implement the DDA line drawing algorithm.
5. To implement Bresenham's line drawing algorithm.
6. To implement Mid-Point Circle drawing algorithm.
7. To implement Boundary Fill and Flood Fill algorithms.
8. To implement Basic Transformations (translation, rotation and scaling on a rectangle).
9. To implement Composite Transformations
10. To implement animation using C function
11. To implement a cartoon using C function

Course Code: **CSE008E**	**Computer Vision**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To understand basic concepts, terminology, theories, models and methods in the field of
- computer vision.
- To understand principles of human visual system.
- Learn and understand the basic methods of computer vision related to multi-scale representation, edge detection and detection of other primitives.
- To design of a computer vision system for a specific problem.
- To learn different concepts object recognition, shape representation and segmentation.

Unit – I **(10)**

Introduction:
Human and Computer vision, Introduction to computer vision and Image processing, Image Formation Models: Monocular imaging system, Transformation: Orthogonal, Euclidean, Affine, Projective, Camera model and Camera calibration.

Unit – II **(8)**

Image Processing and Feature Extraction:
Image representations (continuous and discrete) and Modelling, Edges - Canny, LOG, DOG; Line detectors (Hough Transform), Labelling.

Unit – III **(8)**

Motion Estimation:
Regularization theory, Optical computation, Stereo Vision, Motion estimation, Structure from motion.

Unit – IV **(10)**

Shape Representation and Segmentation:
Deformable curves and surfaces, Snakes and active contours, Level set representations, Fourier and wavelet descriptors, Medial representations, Multi-resolution analysis.

Unit – V **(9)**

Object Recognition:
Hough transforms and other simple object recognition methods, Shape correspondence and shape matching, Principal Component analysis, Shape priors for recognition

Textbooks:

1. Computer Vision - A modern approach, by D. Forsyth and J. Ponce, Prentice Hall
2. Robot Vision, by B. K. P. Horn, McGraw-Hill.
3. Introductory Techniques for 3D Computer Vision, by E.Trucco and A.Verri.

Reference Books:

1. Richard Hartley and Andrew Zisserman, Multiple View Geometry in Computer Vision, Second Edition, Cambridge University Press, March 2004.
2. K. Fukunaga; Introduction to Statistical Pattern Recognition, Second Edition, Academic Press, Morgan Kaufmann, 1990.

Course Code: **CSE009E**	**Data Visualization**	Credits: **03** L – **2** P – **2**

Course Outcomes (COs):

- Understand the significance of data visualization
- Differentiate between the types of charts and plots used for data visualization
- Perform exploratory data analysis using pandas, matplotlib and seaborn
- Create various types of charts using Plotly
- Create a basic dashboard using Plotly and Dash.

Unit – I (10)

Introduction: What is Data Visualization? Benefits of Data Visualization. Types of analysis for data visualization - univariate analysis, bivariate analysis, multivariate analysis. Exploratory data Analysis: Definition and Significance.
Introduction to Charts and Plots: Charts/plots used for visualization (what do they represent and when to use where): Line, scatter (bubble plot) , histogram, bar (stacked charts), pie chart (doughnut chart) , box plot, hex bin plots, violin plot, heat maps, gantt charts, word clouds (text visualization). Effectiveness of visualization across data types (categorical, ordinal and quantitative)

Unit – II (10)

Visualization using pandas: Setting up the environment. Line plot, bar plot, stacked plot, histogram, box plot, area plot, scatter plot, hex plot, pie plot, scatter matrix, subplots.
Matplotlib: What is matplotlib? Setting up the environment. Line, scatter, hist, bar, pie subplot, box, doughnut, word clouds controlling ticks and axis - xlim, ylim, xticks, yticks, nested pie plot, labeling a pie plot.
Polar plots - brief intro, bar chart on polar axis, line plot on polar axis, scatter plot on polar axis.

Unit – III (10)

Seaborn What is Seaborn? Setting up the environment. Customizing plots - background colour, grids, despine, scaling plots, scaling fonts and line widths and rc parameter Scatter plot - style and size, hue, jitter, swarm Plotting univariate distributions – histogram Plotting bivariate distributions - hexplot, kde plot, boxen plot, ridge plot.

Unit – IV (9)

Plotly and Dash 1: Introduction to plotly - What is plotly? What better features does it offer compared to pre-existing options? Line chart, area plots, bar chart, scatter plots, pie chart, bubble chart, box plot, histogram, distplot, heatmaps, gantt chart, word clouds. Tables?
Introduction to dash - What is Dash? Scope of using dash with plotly. Layout basics.

Unit – V (6)

Dashboard components: Dash components, HTML Components, core components, markdown with dash.
Interactive components: Single callbacks, Callbacks for graph, Multiple input/output.

Textbooks:

1. Python Data Visualization: An Easy Introduction to Data Visualization in Python with Matplotlip, Pandas, and Seaborn by Samuel Burns

Reference Books:

1. Python Data Visualization Cookbook Book by Igor Milovanovic
2. Interactive Dashboards and Data Apps with Plotly and Dash (May 2021) by Elias Dabbas.

Course Code: **CSE010E**	**Deep Learning**	Credits: **04** L – **3** P – **2**

Course Outcomes (COs):

- Identify the deep learning algorithms which are more appropriate for various types of learning tasks in various domains.
- understand the theory behind deep learning algorithms such as Convolutional Neural Networks,
- Learn how to apply deep learning algorithms to various learning problems.
- Know the open issues in deep learning, and have a grasp of the current research directions.
- Implement deep learning algorithms and solve real-world problems.

Unit – I **(11)**

History, Neural networks basics, McCulloch Pitts Neuron, Thresholding Logic, Perceptrons, Perceptron Learning Algorithm, Multilayer Perceptrons (MLPs), Representation Power of MLPs, Sigmoid Neurons, Gradient Descent, Feedforward Neural Networks, Representation Power of Feedforward Neural Networks.

Unit – II **(8)**

FeedForward Neural Networks, Backpropagation, Gradient Descent (GD), Momentum Based GD, Nesterov Accelerated GD, Stochastic GD, AdaGrad, RMSProp, Adam, Eigenvalues and Eigenvectors, Eigenvalue Decomposition, Basis.

Unit – III **(9)**

Principal Component Analysis and its interpretations, Singular Value
Decomposition, Autoencoders and relation to PCA, Regularization in autoencoders, Regularization: Bias Variance Tradeoff, L2 regularization, Early stopping, Dataset augmentation, Parameter sharing and tying, Injecting noise at input, Ensemble methods, Dropout.

Unit – IV **(9)**

Greedy Layerwise Pre-training, Better activation functions, Better weight initialization methods, Batch Normalization, Convolutional Neural Networks, (Le-Net, AlexNet, ResNet), Visualizing Convolutional Neural Networks, Guided Backpropagation.

Unit – V **(9)**

Recurrent Neural Networks, Backpropagation through time (BPTT), Vanishing and Exploding Gradients, GRU, LSTMs, Encoder Decoder Models.

Textbooks:

1. *Deep Learning*, MIT Press, Ian Goodfellow and Yoshua Bengio and Aaron Courville http://www.deeplearningbook.org.
2. Deep Learning for Coders with fastai and PyTorch by Sylvain Gugger and Jeremy Howard.
3. *Neural Networks and Deep Learning* by Michael Nielsen
4. Deep Learning for Vision Systems by Mohamed Elgendy

Reference Books:

1. Deep Learning with Python Francois Chollet
2. TensorFlow Deep Learning Cookbook, Packt, Antonio Gulli and Amita Kapoor.
3. Deep Learning with TensorFlow 2 and Keras by Antonio Gulli, Sujit Pal.

Online Resources:

1. http://www.cse.iitm.ac.in/~miteshk/CS6910
2. https://www.coursera.org/specializations/deep-learning

List of Experiments

1. Classification of CIFAR-10 Dataset (TensorFlow/Keras)
2. Classification of MNIST Dataset (Keras)
3. Classification of Fashion MNIST Dataset (TensorFlow/Keras)
4. Pneumonia detection using Chest X-ray Dataset (TensorFlow)
5. Fruit Classification
6. Classification of different crop varieties
7. Parkinson Disease Detection using CNN
8. Predict IDC in Breast Cancer Histology Image Dataset
9. CNN for Skin Cancer Detection.
10. Birds Species Classification

Course Code: **CSE011E**	**Distributed Computing**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Understand the various communications models used by the distributed systems to orchestra their work.
- examine how existing systems like Torrent, Chord etc. have applied the concepts of distributed systems in designing large systems.
- design the algorithms for the efficient and scalable work of the small distributed systems.
- learn about communication scenarios in distributed systems.
- Understand about distributed multimedia and file system.

Unit – I **(10)**

Introduction to Distributed System: Goals, Hardware and Software concepts, Resource sharing and the web, Challenges arising from the construction of distributed systems. Clock Synchronization.
System Models: Architectural models Client-server model, Peer-to-peer model, Variations of the above models, Forms of computing Monolithic, Distributed, Parallel, and Cooperative.

Unit – II **(9)**

Clock synchronization, Message Ordering and Group Communication – Asynchronous Execution with Synchronous communication, Synchronous program order on an asynchronous system, casual order; Termination Detection Algorithms - Distributed snapshot, weight throwing, spanning tree based termination detection.

Unit – III **(7)**

Networking and Internetworking, Network Principles, The API for the internet protocol, External data representation and marshalling, Client-server communication, Group communication, Remote procedures call with RPC operations, Case study: interposes communication in UNIX.

Unit – IV **(7)**

Communication: Layered protocols, Remote object invocation, Message-oriented communication, Stream-oriented communication, Processes: Threads, Virtualization, Networked user interfaces in clients. Servers – design issues and server clusters. Approaches to code migration. Checkpointing and Rollback.

Unit – V **(7)**

Distributing Multimedia: DMS, Advanced Distributed Computing Paradigms: Mobile Agents
Security: Introduction, Secure channels, Access control, Security management.
Distributed File System: Sun network file system, CODA files system. Case Study: CORBA, Distributed COM.

Textbooks:

1. *Distributed Computing: Principles and Applications*, M. L. Liu, Pearson/Addison-Wesley, ISBN: 0-201-79644-9.
2. Taunenbaum, *Distributed Systems: Principles and Paradigms*.

Reference Books:

1. G. Coulouris, J. Dollimore, and T. Kindberg, *Distributed Systems: Concepts and Design*, Pearson Education.
2. Ghosh, Sukumar. *Distributed systems: an algorithmic approach*. CRC press, 2014.

Online Resources:

1. Tel, Gerard. *Introduction to distributed algorithms*. Cambridge university press, 2000.
2. Lynch, Nancy A. *Distributed algorithms*. Elsevier, 1996.
3. Attiya, Hagit, and Jennifer Welch. *Distributed computing: fundamentals, simulations, and advanced topics*. Vol. 19. John Wiley & Sons, 2004.

Course Code: **CSE012E**	**Evolutionary Computing**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Explain the principles underlying Evolutionary Computation in general and Genetic Algorithms in particular and analyze the design of a genetic algorithm, and comment on its weaknesses and strengths.
- Demonstrate an understanding of how to apply techniques of swarm intelligence to optimization problems.
- Understanding of Memetic Algorithms and its applications and hybridization with GA and a PSO.
- Understand how to solve complex search problems with discrete optimization concepts and know various constraint handling methods.
- Use of evolutionary algorithms in multi-objective optimization.

Unit – I **(10)**

Genetic Algorithms: GA concepts – encoding, fitness function, population size, selection, crossover and mutation operators, along with the methodologies of applying these operators.
Binary GA and their operators, Real Coded GA and their operators.

Unit – II **(8)**

Particle Swarm Optimization: PSO Model, global best, Local best, velocity update equations, position update equations, velocity clamping, inertia weight, constriction coefficients, synchronous and asynchronous updates, Binary PSO.

Unit – III **(8)**

Memetic Algorithms: Concepts of memes, Incorporating local search as memes, single and multi-memes, hybridization with GA and PSO, Generation Gaps, Performance metrics.

Unit – IV **(10)**

Discrete optimization: Use of evolutionary computation to solve travelling salesman problem; Time Table problem, Vehicle routing problem.
Constrained optimization: Methods based on rejection strategies, repair strategies, specialized operators and penalty functions.

Unit – V **(9)**

Multi-Objective Optimisation: Linear and nonlinear multi-objective problems, convex and non-convex problems, dominance-concepts and properties, Pareto-optimality, Use of Evolutionary computations to solve multi objective optimization, bi level optimization, Theoretical Foundations.

Textbooks:

1. *Recent Advances in Memetic algorithms*, W.E. Hart, N.Krasnogor, J.E Smith, Springer Berlin Heidelberg, New York. *Multi-Objective Optimization using Evolutionary Algorithms*, K. Deb, John Wiley and sons, New Delhi.
2. *Evolutionary Algorithms for solving Multi objective problems*, C.A.Coello Coello, D.A.Van Veldhuizen and G.B Lamont,Kluwer.

Reference Books:

1. *Genetic Algorithms and Engineering Design*, M.Gen, and R.Cheng, Wiley,New York
2. *Optimization for engineering Design Algorithms and examples*, K.Deb, Prentice Hall of India, New Delhi
3. *Genetic Algorithms + Data Structures= Evolution Programs*, Z. Michalewicz, Springer-Verlag ,3rd London,UK

Course Code: **CSE013E**	**Modeling and Simulation**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Create a relevant model for a multitude of problems from science and engineering by extracting the necessary and relevant information regarding the problem.
- Define the different modelling terms by analyzing the system or the data that is present.
- Implement the model on the computer and from the results check for the validity of the model and correctness of the assumptions present in the model.
- Analyze the outcomes (mostly through visualizations) and make predictions.
- Verify and validate the simulation models using different languages.

Unit – I **(8)**

Concepts of Systems, Models, and Simulation. Distributed Lag Model, Cobweb Models, The process of a simulation Study, Exponential Growth Models, Exponential Decay Models.

Unit – II **(8)**

Type of simulation, Discrete-Event Simulation: Time-Advance Mechanisms, Components and Organization of a Discrete-Event Simulation Model. Monte Carlo Method. Simulation of Single-Server Queuing System, Simulation of an Inventory System.

Unit – III **(8)**

Continuous Simulation: Pure-pursuit Problem.
Random Number Generators: Linear Congruential Generators, Other kinds of Generators, Testing Random-Number Generators. Generating Random Variates: General Approaches, Continuous and Discrete distributions.

Unit – IV **(8)**

Introduction to GPSS, General Description, GPSS block-diagram, Simulation of a Manufacturing Shop. SNA, Function, Simulation of a Supermarket, GPSS Model of a Simple Telephone System.

Unit – V **(8)**

Output Data Analysis for a Single System: Transient and Steady-State Behaviour of a Stochastic Process, Type of Simulations about output Analysis and Statistical Analysis for Testing Simulation. Verification and Validation of Simulation. An introduction of different types of simulation languages.

Textbooks:

1. G. Gordon, "*System Simulation*", Pearson Education
2. Law and Kelton, "*Simulation Modelling and Analysis*", McGraw Hill

Reference Books:

1. N. Deo, "*System Simulation with Digital Computer*", Prentice Hall of India
2. Fred Maryanski, "*Digital Computer Simulation*", CBSPD
3. James A. Pyne, "*Introduction to Simulation- Programming Techniques and Methods of Analysis*", McGraw Hill
4. Zeigler, Bernard P., Tag Gon Kim, and Herbert Praehofer. *Theory of modeling and simulation*. Academic press, 2000. Banks et al, "*Discrete event Simulation*", Pearson Education.
5. Sokolowski, John A., and Catherine M. Banks. *Modeling and simulation fundamentals: theoretical underpinnings and practical domains*. John Wiley & Sons, 2010.
6. Sokolowski, John A., and Catherine M. Banks, eds. *Principles of modeling and simulation: a multidisciplinary approach*. John Wiley & Sons, 2011.
7. Woods, Robert L., and Kent L. Lawrence. *Modeling and simulation of dynamic systems*. Upper Saddle River, NJ: Prentice Hall, 1997.
8. Egeland, Olav, and Jan Tommy Gravdahl. *Modeling and simulation for automatic control*. Vol. 76. Trondheim, Norway: Marine Cybernetics, 2002.

Course Code: **CSE014E**	**Multimedia Technologies**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To provide the foundation knowledge of multimedia computing.
- To understand the design and development of multimedia systems according to the
- requirements of multimedia applications.
- Learn and understand the multimedia compression techniques.
- Understand different Input and output multimedia technologies.
- To learn different concepts of multimedia communication.

Unit – I **(6)**

Multimedia System Design:
Multimedia Elements, Multimedia Applications, Multimedia System Architecture, Evolving Technologies for Multimedia Systems, Multimedia Databases

Unit – II **(12)**

Audio and Speech:
Data acquisition, sampling and quantization, human speech production mechanism, digital model of speech Production, analysis and synthesis, low bit rate speech compression, MPEG audio compression.

Unit – III **(6)**

Images and Video:
Image acquisition and representation, composite video signal, NTSC, PAL and SECAM video standards; Bi-level image compression standards, JPEG and MPEG.

Unit – IV **(12)**

Multimedia Communication:
Fundamentals of data communication and networking, bandwidth requirements of different media; Real time constraints: Audio latency, video data rate; Multimedia over LAN and WAN, multimedia conferencing.

Unit – V **(9)**

MULTIMEDIA APPLICATION DESIGN:
Types of Multimedia systems - Virtual Reality Design - Components of Multimedia system -Distributed Application Design Issues – Multimedia Authoring and User Interface -Hypermedia Messaging – Distributed Multimedia Systems

Textbooks:

1. Li, Z.N. and Drew, M.S., “Fundamentals of Multimedia”, Pearson Education.
2. Hillman, D., “Multimedia Technology and Application”, Galgotia Publication.

Reference Books:

1. Steinmetz, R., “Multimedia Computing, Communication and Applications”, Pearson Education.
2. Buford, J., “Multimedia Systems”, Addison Wesley.
3. Tay Vaughan, “Multimedia making It work”, TMH.

Course Code: **CSE015E**	**Natural Language Processing**	Credits: **04** L – **3** P – **2**

Course Outcomes (COs):

- Practice and learn the computational properties of natural languages and the commonly used algorithms for processing linguistic information
- Will learn the techniques widely used in industries
- Able to design models and solve the real-world challenges
- Learn the concept behind natural language processing and able to extract information from text and verbal data.
- Understanding semantics and pragmatics of English language for processing

Unit – I (8)

Introduction to NLP:
History of NLP, Generic NLP system, levels of NLP, Knowledge in language processing, Ambiguity in Natural language, stages in NLP, challenges of NLP, Applications of NLP: Machine translation, Information retrieval, Question answers system, categorization, summarization, sentiment analysis, Named Entity Recognition.

Unit – II (10)

Language Modelling and Text Representation: Intro. N-Gram, N-Gram probability estimation, Evaluation of Language Model, **Smoothing Techniques**: Laplace, Good Turing, Kneser-Ney, **Text Representation:** one hot encoding, Bag-of-word, TF-IDF, Vector space Model, Latent semantic Analysis Word embedding Word2Vec, Glove, fastText.

Unit – III (10)

Word Level Analysis: Morphology analysis –survey of English Morphology, Inflectional morphology and Derivational morphology, Part-Of-Speech tagging(POS)- Tag set for English (Penn Treebank) , Rule-based POS tagging, Stochastic POS tagging, Hidden Markov Model.

Unit – IV (12)

Syntactic and Semantic Analysis: Introduction, Context-Free Grammar, Parsing, Probabilistic Parsing Meaning Representation, **Semantic Analysis**: Lexical Semantics, Word Sense Disambiguation, Attachment for fragment of English- sentences, noun phrases, Verb phrases, prepositional phrases, Relations among lexemes and and their senses – Homonymy, Polysemy, Synonymy, Hyponymy.

Unit – V (5)

Pragmatics: Discourse –reference resolution, reference phenomenon, syntactic and semantic constraints on co-reference.

Textbooks:

1. Steven Bird, Ewan Klein, Edward Loper, "*Natural Language Processing in Python*", O'Reilly.
2. Jurafsky, D. and J. H. Martin, *Speech and Language Processing*, Prentice-Hall.
3. C. D. and H. Schutze, *Foundations of Statistical Natural Language Processing*,
4. Manning, The MIT Press.
5. Allen,J, *Natural Language Understanding*, The Benajmins Cummings Publishing
6. Company Inc.
7. Cover,T. M. and J.A.Thomas, *Elements of Information Theory*, Wiley.
8. Charniak, E, *Statistical Language Learning*, The MIT Press.

Reference Books:

1. Natural Language Processing with Transformers by Lewis Tunstall, Leandro von Werra, and Thomas Wolf
2. Natural Language Processing with Python Written by Steven Bird, Ewan Klein and Edward Loper.

Online Resources:

1. https://nptel.ac.in/courses/106105158
2. https://www.youtube.com/watch?v=5hKxvh4RAsY&list=PLyqSpQzTE6M_EcNgdZ2qOtTZe7YI4Eedb

List of Experiments

1. Learning installation of Python and NLTK Library
2. Tokenizing words and Sentences, Stemming and Lemmatization Using NLTK
3. Stop Word Removal Using NLTK
4. Parts of Speech Tagging
5. Named Entity Recognition
6. Text Modelling Using Bag of Words
7. Text Modelling Using TF-ID
8. Building Character N-Gram Model
9. Building Word N-Gram Model
10. Latent Semantic Analysis in Python
11. Utilizing Consumer Complaints Dataset for Complaint classification using NLP
12. Explore NLP models on Email Spam Detection.
13. Explore NLP models on Disaster Tweets.

Course Code: **CSE016E**	**Open Source Technologies**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To provide a basic idea of Open source technology
- To Understand the open source ecosystem.
- Understand the role and future of open source software in the industry.
- Describe the impact of legal, economic, and social issues of Open Source.
- Impact of Open Source Technology.

Unit – I **(8)**

Introduction to Open Source: Why open Source, what is Open Source, Open Source Principles, open standard requirements, History, Free Software, Free software vs. Open Source software, Public Domain Software FOSS, Open Source Development Model.

Unit – II **(10)**

Principles and Open Source Methodology: History, Open Source Initiatives, Open Standard Principles, Open Source Methodology, Software Freedom, OSSD, Licenses – Copy right, Copy left, Patent, Zero Marginal Technologies, Income generation opportunities, Internalization.

Unit – III **(6)**

Case Studies – Apache, BSD, Linux, Mozilla (Firefox), Wikipedia, Joomla, GCC, Open Office.

Unit – IV **(10)**

Open Source Project - Starting, Maintaining, Open Source Hardware, Virtualization Technologies, Containerization Technologies: Docker, Development tools, IDEs, debuggers, Programming languages, LAMP, Open Source database technologies.

Unit – V **(8)**

Open Source Ethics, Open Vs Closed Source, Government Ethics, Impact of Open Source Technology, Shared Software, Shared Source, Social and Financial impacts of open source technology.

Textbooks:

1. Kailash Vadera, Bhavyesh Gandhi, "Open Source Technology", Laxmi Publications Pvt Ltd. 2012, 1st Edition.

Reference Books:

1. Fadi P. Deek and James A. M. McHugh, "Open Source: Technology and Policy", Cambridge Universities Press 2007.

Course Code: **CSE017E**	**Optimization Techniques**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Recall the theoretical foundations of various issues related to linear programming modeling to formulate real-world problems as a L P model
- Explain the theoretical workings of the graphical, simplex and analytical methods for making effective decision on variables so as to optimize the objective function.
- Identify appropriate optimization method to solve complex problems involved in various industries.
- Find the appropriate algorithm for allocation of resources to optimize the process of assignment.

Unit – I **(7)**

Introduction, Concept of optimization, optimization problems, Types of Optimization Techniques.

Unit – II **(9)**

Linear programming –formulation-Graphical and simplex methods-Big-M method Two phase method-Dual simplex method-Primal Dual problems.

Unit – III **(9)**

Unconstrained one-dimensional optimization techniques -Necessary and sufficient conditions – Unrestricted search methods-Fibonacci and golden section method Quadratic Interpolation methods, cubic interpolation and direct root methods.

Unit – IV **(9)**

Unconstrained n dimensional optimization techniques – direct search methods – Random search – pattern search and Rosen brooch's hill claiming method- Descent methods-Steepest descent, conjugate gradient, quasi -Newton method.

Unit – V **(10)**

Constrained optimization Techniques- Necessary and sufficient conditions – Equality and inequality constraints-Kuhn-Tucker conditions-Gradient projection method-cutting plane method- penalty function method.

Textbooks:

1. Ashok D. Belegundu, Tirupathi R. Chandrupatla, "Optimization Concepts and Applications in Engineering", Cambridge University Press.
2. An Introduction to Optimization Techniques "Vikrant Sharma, Vinod Kumar Jain, Atul Kumar"

Reference Books:

1. Taha,H.A., Operations Research –An Introduction,Prentice Hall of India,2003.
2. Fox, R.L., „Optimization methods for Engineering Design", Addition Welsey

Online Resources:

1. https://nptel.ac.in/courses/111105039

Course Code: **CSE018E**	**Reinforcement Learning**	Credits: **04** L – **3** P – **2**

Course Outcomes (COs):

- Understand the elements of an MDP and pose a problem as an MDP.
- Understand and appreciate the concepts like policy, value function, reward and the Bellman equations.
- Understand the Temporal difference learning and solve an RL problem with TD learning
- Appreciate the need and power of Deep RL and solve basic problems using Deep RL.
- Get acquainted with the various RL libraries.

Unit – I **(12)**

Introduction and Language models
Introduction to Reinforcement learning, comparison with supervised and unsupervised learning. The reinforcement learning environment – agent, environment, action and reward. Exploration vs Exploitation – brief idea

Unit – II **(9)**

Markov Decision Processes
Bandit and online problems. Markov Decision Processes. Return and value functions. Dynamic Programming and Bellman equations. Q-Learning.

Unit – III **(8)**

Monte Carlo Methods Temporal Difference Learning
Introduction to Monte Carlo estimation. Monte Carlo Policy Evaluation. Gradient Based Monte Carlo Algorithm.

Unit – IV **(10)**

Temporal Difference Learning
Temporal Difference Learning, TD Prediction Methods, Advantages of TD Prediction Methods, Optimality of TD (0), Sarsa: On-policy TD Control,
Q-learning: Of-policy TD Control, Expected Sarsa.

Unit – V **(8)**

Deep RL and toolkits
Deep Q-Learning: OpenAI gym – modelling and policy learning.
Actor-critic algorithm, Stable Baseline.

Textbooks:

1. Deep Reinforcement Learning Hands-On - Second Edition.
2. Deep Reinforcement Learning in Action by Alexander Zai and Brandon Brown.

Reference Books:

1. Reinforcement Learning: An Introduction, Sutton and Barto, 2nd Edition.
2. Grokking Deep Reinforcement Learning by Miguel Morales.

Online Resources:

1. https://www.youtube.com/watch?v=sHcO0hzdp0o&list=PLuWx2S0SyaDctJtVKHhmjYACmHZ3nX9ew

List of Experiments

1. Implement a simple Q-learning algorithm to solve the classic "Frozen Lake" problem using Python.

2. Apply Q-learning to learn a winning strategy in the game of Tic-Tac-Toe.

3. Design and implement an environment for a custom grid world problem and then apply the SARSA algorithm to solve it.

4. Create a Python script to simulate a Markov Decision Process (MDP) and apply both policy iteration and value iteration to find an optimal policy.

5. Implement the OpenAI Gym's "Taxi-V3" problem. It is a grid world problem where a taxi must navigate to pick up and drop off passengers. Implement Q-learning or SARSA to solve it.

6. Implement the "CartPole" problem using OpenAI Gym. The goal is to balance a pole on a moving cart.

7. Cliff Walking: Create a custom grid world where an agent must navigate around a cliff to reach a goal. Apply SARSA or Q-learning to solve the problem.

8. Implement DQN for a Classic Control Problem: Choose a classic control problem from OpenAI Gym (e.g., CartPole or Acrobot). Implement a DQN agent to learn the optimal policy

9. Implement Actor-Critic for Continuous Control: Choose a continuous control task from OpenAI Gym (e.g., Pendulum or Lunar, Lander Continuous). Implement an Actor-Critic agent to solve it.

10. Experiment with different exploration strategies in DQN, such as epsilon-greedy, Boltzmann exploration, or noisy networks.

Course Code: **CSE019E**	**Ruby on Rails**	Credits: **03** L – **2** P – **2**

Course Outcomes (COs):

- To leverage Ruby's succinct and flexible syntax to maximize your productivity,
- To balance Ruby's functional, imperative, and object-oriented features,
- To gain experience using Ruby to write scripts that automate common tasks and to write programs that solve more significant problems,
- To learn Ruby's built-in methods for manipulating strings, files, and hashes,
- To develop powerful web applications with the Ruby on Rails framework,
- To manipulate strings using regular expressions, and
- To create useful stand-alone applications in Ruby.

Unit – I **(8)**

Ruby Fundamentals

Ruby Data Types and Variables – String, Integer, Float, Boolean and Nil values Properties of Ruby data types Instance variables and Local variables Global variables.

Functions and Control Flow – Built-in functions, creating your own functions, passing arguments and returning values, if/else and unless statements, while/until loops.

Unit – II **(10)**

Ruby data structures and classes

Ruby Data Structures - Arrays: The Simplest Collections, Hashes, Enumerators, Common Iterators

Classes – Creating classes, Inheritance, Class Methods, Overriding Methods.

Unit – III **(10)**

Ruby Models and Forms

Models and Forms – Generating a Model, How Migration Files work, Migrating the Database, Rolling back a migration Rails Forms – Rails forms vs HTML forms, Rails Form Helpers, Rails forms: form_for, form_tag, and form_with, connecting a form to a Model Views – Creating a View, Adding Dynamic Data, rendering a Partial, Rendering a View.

Unit – IV **(9)**

Ruby Advanced Models

Model Validations, the purpose of validations, adding basic validations, preventing submission of empty forms, customizing validations, adding Error Messages, Built-in Model methods, adding methods to models.

Model Relationships – has_one and belongs_to relationships, has_and_belongs_to_many: Simple Many-to-Many Relationships, has_many, through: Advanced Many-to-Many Relationships with Additional Metadata, and Polymorphic Relationships.

Unit – V **(10)**

Ruby on Rails

Putting the Ruby into Rails, Formatting in Rails, Working with Dynamic web pages, Scaffolding in Rails, applying categories to Posts. Deploying Rails to Production, Deploying to Production on Heroku with Puma, Deploy Rails to Production on Ubuntu Server 22.04.

Textbooks:

1. "The Book of Ruby: A Hands-On Guide for the Adventurous," Huw Collingbourne, and Chris Takemura, No Starch Press.
2. "The Ruby Programming Language," Yukihiro Matsumoto, David Flanagan, Shroff / O'Reilly.
3. "The Ruby Way: Solutions and Techniques in Ruby Programming, 3e" Hal Fulton, and André Arko, Addison-Wesley.

Reference Books:

1. "Head First Rails: A Learner's Companion to Ruby on Rails," David Griffiths, Shroff / O'Reilly.
2. "Learn Rails 6: Accelerated Web Development with Ruby on Rails," Adam Notodikromo, Apress.

Course Code: **CSE020E**	**Soft Computing**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Understand the components of Soft Computing and their advantages, dis-advantages and applications.
- Apply various Neural Learning Algorithms to design, analyze and perform experiments on real life problems.
- Conceptualize fuzzy logic and its implementation for various real world applications.
- Understand the fundamentals of components of Evolutionary computation and their applications.

Unit – I **(7)**

Introduction to Soft Computing and its components: Machine Learning: Neural Networks, Support Vector Machines. Fuzzy Logic. Evolutionary Computation: Genetic Algorithms. Introduction to Ant Systems and Swarm Intelligence.

Unit – II **(12)**

Artificial Neural Networks: History, overview of biological Neuro-system, Mathematical Models of Neurons, ANN architecture, Learning rules, Learning Paradigms-Supervised, Unsupervised and Reinforcement Learning, ANN training Algorithms perceptions, Training rules, Delta, Back Propagation Algorithm, Multilayer Perceptron Model, Hopfield Networks, Associative Memories, Applications of Artificial Neural Networks.

Unit – III **(8)**

Fuzzy Logic: Introduction to Fuzzy Logic, Classical and Fuzzy Sets: Overview of Classical Sets, Membership Function, Fuzzy rule generation. Operations on Fuzzy Sets: Compliment, Intersections, Unions, Combinations of Operations, Aggregation, Operations.

Unit – IV **(8)**

Fuzzy Arithmetic: Fuzzy Numbers, Linguistic Variables, Arithmetic Operations on Intervals and Numbers, Lattice of Fuzzy Numbers, Fuzzy Equations. Fuzzy Logic: Classical Logic, Multivalued Logics, Fuzzy Propositions, Fuzzy Qualifiers, Fuzzy Decision Making, Fuzzy Control Systems.

Unit – V **(10)**

Genetic Algorithm: Concept of Genetics and Evolution and its application to probabilistic search techniques. Basic GA framework and different GA architectures. GA operators: Encoding, Crossover, Selection, Mutation, etc. Solving single-objective optimization problems using GAs.

Textbooks:

1. S. Rajasekaran, G. A. Vijayalakshmi Pai, "Neural Networks, Fuzzy Logic and Genetic Algorithm: Synthesis and applications," PHI Learning Pvt. Ltd

Reference Books:

1. Jacek M. Zurada, "Introduction to Artificial Neural Systems," Jaico Publishing House
2. Timoth J. Ross, "Fuzzy Logic with Engineering Applications," 3rd ed. Wiley India
3. S. N. Sivananandam and S. N. Deepa, "Principles of Soft Computing," 2nd ed. Wiley India.
4. H. J. Zimmerman, "Fuzzy Set Theory and its Applications," Allied Publishers Ltd.
5. Zbigniew Michalewicz, Martin Schmidt, Matthew Michalewicz, Constantin Chiriac, "Adaptive Business Intelligence," Springer.

Course Code: **CSE021E**	**Software Project Management**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Comprehend Project Life Cycle (PLC).
- Demonstrate project planning, monitoring and controlling.
- Comprehend ethics and leadership skills for managing projects.
- Comprehend project procurement and risk management.

Unit – I **(10)**

An overview of IT Project Management and Project Integration Management: The State of IT Project Management, Context of Project Management, Project Goals, PMBOK, Project Life Cycle and IT Development, Agile Project Management, Green IT. Information Technology Project Methodology (ITPM), Project Feasibility, Business Case, Project Selection and Approval, Project Management Processes. Project Integration Management Processes, Project Charter, Project Planning Framework. Case Studies.

Unit – II **(12)**

Project Scope Management, Project Time Management and Project Cost Management: Scope Planning, Project Scope Definition, Project Scope Verification, Scope Change Control. Developing the Project Schedule, Logic Diagrams and Networks, AON, Critical Path, PERT, CPM. Allocating Resources to the Project, Resource Loading, Resource Leveling, Constrained Resources and Goldratt's Critical Chain. Cost Estimating, Cost Escalation, Cost Estimating Process, Elements of Budgets and Estimates. Case Studies.

Unit – III **(10)**

Project Human Resource Management and Stakeholder Management: Organization and Project Planning, Formal Organization, Project Team, Multidisciplinary Teams. Project Leadership, Leadership Styles, Ethics in Projects, Role of Project Manager. Managing Change, Resistance and Conflicts, the Nature of Change, Change Process, Change Management Plan, Dealing with Resistance and Conflicts. Introduction to Project Stakeholder Management, Stakeholder Analysis. Case Studies.

Unit – IV **(6)**

Project Procurement Management and Risk Management: Project Procurement Processes, Outsourcing. Request for Proposal (RFP), Project Proposal, Project Contracting. Risk Identification, Risk Assessment, Risk Projection, RMMM Plan. Case Studies.

Unit – V **(7)**

Project Communication Management and Quality Management: Monitoring and Controlling the Project, Project Communication Plan, Plan-Monitor-Control Cycle. Project Metrics, Reporting Performance and Progress, Information Distribution. Software Quality, Five Views of Software Quality, McCall's Quality Factors and Criteria, Quality Cost and Benefits, Capability Maturity Model (CMM), Six Sigma. Case Studies.

Textbooks:

1. *Information Technology Project Management* by Jack T. Marchewka, Wiley India, 4th Edition, 2013.
2. *Project Management for Engineering, Business, and Technology* by John M. Nicholas, A Butterworth-Heinemann Title, 4th Edition, 2011.

Reference Books:

1. *Software Project Management* by Bob Hughes, Mike Cotter, Rajib Mall, Tata McGraw Hill, 6th Edition, 2017.
2. *Green IT: Managing Your Carbon Footprint* by BCS, The Chartered Institute for IT, 1st Edition, 2012.

Course Code: **CSE022E**	**Software Testing and Quality Assurance**	Credits: **03** L – **3** P – 0

Course Outcomes (COs):

- Understand software testing concepts and strategies.
- Demonstrate designing and execution of test cases using testing techniques.
- Apply recent automation tools for testing software.
- Comprehend different approaches of quality management for software systems.

Unit – I **(10)**

Goals and Model for Software Testing, Software Testing Terminology and Methodology, Software Testing Life Cycle (STLC). Project Verification, Verification of Requirements, High Level Design, Low Level Design and Code. Validation, Unit Testing, Integration Testing, Function Testing, System Testing, Acceptance Testing.

Unit – II **(8)**

Testing Techniques: Static Testing, Inspection, Review and Walkthrough. Dynamic Testing, White Box Testing, Basis Path Testing, Loop Testing, Data Flow Testing, Mutation Testing, Black Box Testing, Boundary Value Analysis, Equivalence Class Testing, Decision Table Based Testing, Usability and Accessibility Testing. Regression Testing, Objectives, Need and Types.

Unit – III **(7)**

Testing Metrics: Measurement Objectives, Attributes and Corresponding Metrics, Estimation Models for Estimating Testing Efforts, Architectural Design Metrics, Information Flow Metrics, Cyclomatic Complexity Measures, Function Point Metrics, Test Point Analysis, Testing Progress Metrics.

Unit – IV **(10)**

Test Automation and Testing for Specialized Environment: Need of Automation, Guidelines for Automated Testing, Categorization of Testing Tools, Selection of Testing Tools , Data Driven Testing, Keyword Driven Testing. Testing Web Based Systems, Challenges, Security, Navigation and Performance Testing. Testing Agile Based Software, Mobile Application Testing.

Unit – V **(10)**

Software Quality Management: Software Quality, Five Views of Software Quality, McCall's Quality Factors and Criteria, Software Quality Metrics. PDCA Cycle, Quality Plan, Assurance, Control and Methods, Quality Cost and Benefits, Defect Prevention and Root Cause Analysis. Software Quality Tools, Ishikawa Diagram, Check List, Control Chart, Pareto Chart. SQA Models, Software Total Quality Management, Six Sigma, Test Maturity Model (TMM).

Textbooks:

1. Software Testing Principles and Practices by Naresh Chauhan, Oxford Higher Education, Second Edition, 2016.
2. Practical Software Testing by Ilene Burnstein, Springer International Edition, First Edition,2010.

Reference Books:

1. Foundations of Software Testing by Rex Black , Erik Van Veenendaal, Dorothy Graham, Cengage Publications, Third Edition,2015.
2. Software Testing Effective Methods, Tools and Techniques by Renu Rajani, Pradeep Oak, McGraw Hill Education, Second Edition,2017

Course Code: **CSE023E**	**System Software**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To understand the relationship between system software and machine architecture.
- To understand the processing of an HLL program for execution on a computer.
- To understand the process of scanning and parsing.
- To know the design and implementation of assemblers, macro processor, linker and compiler.
- To have an understanding of loader, system software tools.

Unit – I **(8)**

Introduction to System Software and software tools

Language Processors: Introduction Language Processing Activities, Fundamentals of Language Processing and Language Specification, Language Processor Development Tools, Data Structures for Language Processing: Search Data structures, Allocation Data Structures.

Software Tools: Software Tools for Program Development, Editors, Debug Monitors Programming Environments, User Interfaces.

Unit – II **(8)**

Assemblers: Elements of Assembly Language Programming, A Simple Assembly Scheme, Pass Structure of Assemblers, Design of a Two Pass Assembler, A single pass Assembler for IBM PC.

Unit – III **(7)**

Macros and Macro Processors: Macro Definition and Call, Macro Expansion, Nested Macro Calls, Advanced Macro Facilities, Design of a Macro Preprocessor.

Unit – IV **(9)**

Interpreters and Introduction of Compilers: Interpreters, Use and overview of interpreters, Pure and impure interpreters, Phases of the Compiler, Introduction of scanning and parsing, Aspects of compilation

Unit – V **(9)**

Linkers and Loaders: Introduction to linkers, Relocation and Linking Concepts, Design of a linker, Self-Relocating Programs, A Linker for MS-DOS, Linking for Overlays and Loaders

Textbooks:

1. D. M. Dhamdhere, “Systems Programming and Operating Systems”, Second Revised Edition Tata McGraw-Hill, 1999.

Reference Books:

1. Leland L. Beck, “System Software – An Introduction to Systems Programming”, 3rd Edition, Pearson Education Asia, 2000.
2. Santanu Chattopadhyay, “System Software”, Prentice-Hall India, 2007
3. Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman, “Compilers: Principles, Techniques, and Tools”,2nd Edition, Pearson Education Asia

GENERIC ELECTIVES

Course Code: **CSE001G**	**Applied ML for Embedded IoT Devices**	Credits: **04** L – **3** P – **2**

Course Outcomes (COs):

- To learn techniques to design embedded systems used in IoT.
- To apply artificial intelligence, machine learning and statistical algorithms for IoT systems.
- To learn Python based coding of machine learning algorithms applied to set of bench-marked generic data for IoT devices.
- To utilize sensor modules to design environment monitoring IoT devices.
- To implement machine learning models and techniques for the IoT in Python and embedded C/C++ using TensorFlow.
- To deploy and verify different ML models for IoT devices on real hardware.
- To explore TensorFlow Lite for Microcontrollers, Google's toolkit for TinyML.

Unit – I **(8)**

Foundations of Machine Learning (ML)
The Machine Learning Paradigm, Building Blocks of Deep Learning (DL) - Introduction Building Blocks of DL - Regression with Dense NN Building Blocks of DL - Classification with Dense NN Image Classification using CNN Introduction to Edge Impulse – CNN with Cifar-10 Datasets and Model Performance Metrics Preventing Overfitting.

Unit – II **(10)**

Sensors in IoT
Sensors and IoT applications, Overview of working of Sensors, Analog and Digital Sensors, Interfacing of Temperature, Humidity, Motion, Light and Gas Sensor with Arduino. Interfacing of Actuators with Arduino, Interfacing of Relay Switch and Servo Motor with Arduino.

Unit – III **(10)**

IoT Data Analytics
Internet of Things: Sensor data collection, Sensor fusion, Data Analytics, IoT analytics challenges, IoT data acquisition, Data Exploration and Pre-processing, IoT technologies, Architecture and Networking protocols, IoT Communication Technologies, Devices and Gateways.

Unit – IV **(8)**

Tiny Machine Learning (TinyML)
TinyML Overview, TinyML Applications and use-cases, TinyML algorithms machine learning algorithms and optimizations, TinyML frameworks, tools, and techniques.
The "Hello World" of TinyML: Building an application – Creating an Interpreter, Inspecting the Input Tensor, Running Inference on an Input, Reading the Output, Running the Tests.

Unit – V **(9)**

Applications and Deployment to Microcontrollers
TFLite and TFLite-Micro, TFL-Micro Hello World Hello World Code Walkthrough. Motion Classification - Introduction Motion Classification using MCU (Nano 33), SparkFun Edge – Handling Output on SparkFun Edge, Running the Example, Testing the Program, Viewing Debug Data.

Textbooks:

1. *"Tiny ML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers,"* Pete Warden, and Daniel Situnayake, O'Reilly Media.
2. *"Hands-On Artificial Intelligence for IoT,"* Amita Kapoor, Packt Publishing.

Reference Books:

1. *"Big Data, IoT, and Machine Learning: Tools and Applications,"* Marcin Paprzycki, Neha Gupta, and Rashmi Agrawal, CRC Press.
2. *"Applied Machine Learning and AI for Engineers,"* Jeff Prosise, Shroff/O'Reilly.

Course Code: **CSE002G**	**Bioinformatics**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Understanding of the fundamentals of Bioinformatics and its applications
- Ability to analyze and interpret large-scale biological data
- Ability to apply computational methods to solve biological problems
- Create an understanding of the intersection of life and information sciences
- Prepare students for interdisciplinary research that combines biology, computer science, and mathematics

Unit – I **(9)**

Introduction to bioinformatics and history of bioinformatics, Interdisciplinary nature of Bioinformatics, Major Bioinformatics databases and tools, overview of the bioinformatics applications.

Unit – II **(9)**

Basic chemistry of nucleic acids, Structure of DNA, Structure of RNA, DNA Replication, -Transcription, Translation, Genes- the functional elements in DNA, Analyzing DNA, DNA sequencing. Proteins: Amino acids, Protein structure, Secondary, Tertiary and Quaternary structure, Protein folding and function, Nucleic acid-Protein interaction.

Unit – III **(9)**

Biological information categories, Biological information analysis, Transcription profiling, Biomarker discovery, Biological networks and human diseases.

Unit – IV **(9)**

Representation of patterns and relationships: sequence alignment algorithms, regular expressions, hierarchies and graphical models, Phylogenetic BLAST.

Unit – V **(9)**

Visual analysis of biological network to understand disease, Genetic variations and personalized medicine, Microbiome and human health, Biological information dissemination-Web resources and tools, ethics, security and practice.

Textbooks:

1. Understanding Bioinformatics by Marketa Zvelebil and Jeremy O. Baum
2. Fundamental concepts of Bioinformatics – D E Krane and M L Raymer, Pearson Education.
3. Bioinformatics Methods and applications,Genomics, Proteomics & Drug Discovery–Rastogi, Mendiratta and Rastogi, PHI, New Delhi.

Reference Books:

1. Bioinformatics: with fundamentals of genomics and proteomics – Shubha Gopal, et.al., Mc Graw Hill.
2. Developing Bio informatics computer skills – O'Reilly, CBS.
3. Evolutionary Bioinformatics – Forsdyke, Springer.

Course Code: **CSE003G**	**Blockchain**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Understand the structure of a blockchain and why/when it is better than a simple distributed database
- Evaluate the setting where a blockchain based structure may be applied, its potential and its limitations
- Understand what constitutes a "smart" contract, what are its legal implications and what it can and cannot do, now and in the near future
- Attain awareness of the new challenges that exist in monetizing businesses around blockchains and smart contracts.

Unit – I **(10)**

Introduction:
Blockchain, Public Ledgers, Bitcoin, Blockchain 2.0, Smart Contracts, Block in a Blockchain, Transactions, Distributed Consensus, Cryptocurrency to Blockchain 2.0, Types of Blockchain, Limitations of blockchain as a technology, and myths vs. reality of blockchain technology, Blockchain security
Basic Crypto Primitives: Cryptographic Hash Function, Properties of a hash function, Hash pointer and Merkle tree.

Unit – II **(8)**

Bitcoin Basics:
Creation of coins, Payments and double spending, FORTH (the precursor for Bitcoin scripting), Bitcoin Scripts, Bitcoin P2P Network, Transaction in Bitcoin Network, Block Mining.

Unit – III **(11)**

Distributed Consensus:
Distributed Consensus: Distributed Consensus, Distributed consensus in open environments, Consensus in a Bitcoin network, Proof of Work (PoW) Attacks on PoW and the monopoly problem, Proof of Stake, Proof of Burn, Proof of Elapsed Time, Mining Difficulty.

Unit – IV **(9)**

Permissioned Blockchain:
Permissioned Blockchain: Design issues for Permissioned blockchain, Consensus models for permissioned blockchain, RAFT Consensus, Byzantine general problem, Byzantine fault tolerant System, Practical Byzantine Fault Tolerance, Hyperledger fabric platform.

Unit – V **(7)**

Applications in Different Sectors:
Blockchain for Enterprise, Blockchain use Cases, Blockchain in Financial Service, Blockchain for Trade Logistics, Blockchain in Supply Chain, Blockchain in Government, Research directions in Blockchain technology.

Textbooks:

1. Mastering Bitcoin: Unlocking Digital Cryptocurrencies, by Andreas Antonopoulos, Blockchain by Melanie Swa, O'Reilly.
2. Zero to Blockchain - An IBM Redbooks course, by Bob Dill, David Smits, Hyperledger Fabric.

Reference Books:

1. Elrom, E. (2019). The Blockchain Developer: A Practical Guide for Designing, Implementing, Publishing, Testing, and Securing Distributed Blockchain-based Projects. Germany: Apress.

Course Code: **CSE004G**	**C# and .Net Programming**	Credits: **03** L – **2** P – **2**

Course Outcomes (COs):

- Understand C#, .NET framework basics.
- Develop, implement program control statements.
- Understand and implement object-oriented concepts using C#.
- Understand and implement Exception handling using C#.
- Develop Application Development on .NET.

Unit – I **(8)**

Introduction to C#:
Introducing C#, Understanding .NET framework, Literals, Variables, Data Types, Operators, Checked and Unchecked Operators, Expressions.

Unit – II **(10)**

Program Control Statements
Branching, Looping, Methods, Implicit and Explicit casting, Constant, Arrays, Array Class, Array List, String, String Builder, Structure, Enumerations, Boxing and Unboxing.

Unit – III **(11)**

Object Oriented Aspects of C#:
Classes, Objects, Constructors and its types, Inheritance, Properties, Indexers, Index overloading, Polymorphism, Sealed class and methods, Interface, Abstract class, Abstract and Interface, operator overloading, Delegates, Events.

Unit – IV **(9)**

Exception Handling and I/O:
Threading, Exception-Handling Fundamentals, try and catch block, Multiple catch Clauses, Nesting try Blocks, Using finally, Exception Class, user defined Exceptions. I/O: Byte Streams and Character Streams, Predefined Streams, Stream Classes, Binary Streams, Console I/O. File Handling: Write, Read, Append data to a file.

Unit – V **(7)**

Application Development on .NET:
Building windows application, creating window forms with events and controls, menu creation, inheriting window forms, Dialog Box.

Textbooks:

1. Herbert Schildt, “The Complete Reference: C# ”, Tata McGraw Hill, 2012.
2. Christian Nagel, “Professional C# 6 and .NET Core 1.0”, Wiley, 2016.

Reference Books:

1. Andrew Troelsen, “Pro C# 2010 and the .NET 4 Platform”, Fifth edition, A Press, 2010.
2. Ian Griffiths, Matthew Adams, Jesse Liberty, “Programming C#”, Sixth Edition, O'Reilly, 2010.

Course Code: **CSE005G**	**Cloud Computing**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To understand the fundamental concepts and architecture of cloud computing.
- To develop the skills to design, implement, and deploy cloud solutions.
- To become familiar with the key technologies used in cloud computing.
- To learn how to manage cloud resources and ensure the security of cloud solutions.
- To prepare for careers in cloud computing, including positions in cloud architecture, management, and security.

Unit – I **(9)**

Overview of Cloud Computing, Types of Cloud Computing (Public, Private, Hybrid), Characteristics of Cloud Computing (On-demand Self-Service, Broad Network Access, Resource Pooling, Rapid Elasticity, Measured Service), Service Models (Software as a Service, Platform as a Service, Infrastructure as a Service), Deployment Models (Community Cloud, Public Cloud, Private Cloud, Hybrid Cloud), Cloud Computing Benefits and Risks.

Unit – II **(9)**

Cloud Architecture, Virtualization, Networking in Cloud Computing, Cloud Storage (Object Storage, Block Storage, File Storage), Cloud Database (Relational Database, NoSQL Database), Cloud Management and Monitoring.

Unit – III **(9)**

Introduction to Major Cloud Providers (Amazon Web Services, Microsoft Azure, Google Cloud Platform, IBM Cloud), Cloud Computing Services (Compute Services, Storage Services, Database Services, Networking Services, Security Services, Management Services).

Unit – IV **(9)**

Cloud DevOps, Continuous Integration and Continuous Deployment, Cloud Native Applications, Microservices Architecture, Serverless Computing, Containers and Container Orchestration (Docker, Kubernetes), Cloud-based Application Development (Web Applications, Mobile Applications, Big Data Applications).

Unit – V **(9)**

Hybrid Cloud Management, Cloud Federations, Cloud Computing Standards, Cloud Security (Encryption, Access Control, Identity and Access Management), Cloud Deployment and Migration, Cloud Cost Management and Optimization, Advanced Topics in Cloud Computing.

Textbooks:

1. "Cloud Computing: Principles, Systems and Applications" by Rajkumar Buyya, James Broberg, and Andrzej Goscinski
2. "AWS Certified Solutions Architect Official Study Guide: Associate Exam" by Joe Baron, Hisham Baz, Tim Bixler, Biff Gaut, Kevin E. Kelly, Sean Senior, and John Stamper
3. "Microsoft Azure Administrator - Exam Guide AZ-104" by Paul Marino
4. "Architecting Microsoft Azure Solutions" by Michael Washam, Rick Rainey, and Dan Patrick

Online Resources:

1. Amazon Web Services (AWS) official documentation (https://aws.amazon.com/documentation/)
2. Microsoft Azure official documentation (https://docs.microsoft.com/en-us/azure/)
3. Google Cloud Platform official documentation (https://cloud.google.com/docs)

Course Code: **CSE006G**	**Cyber Physical Systems**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To understand the need and purpose of the different components of Cyber Physical Systems.
- To ideate, design, and prototype Cyber Physical Systems.
- To evaluate the performance of a Cyber Physical System.
- To learn a "systems" perspective for designing, monitoring, and managing large scale infrastructure.
- To have a hands-on experience in prototyping cyber-physical systems.
- To address real-world problems through Cyber Physical Systems.

Unit – I **(8)**

Introduction to Cyber Physical Systems

Cyber Physical Systems – Overview, Memory Architectures, Axioms of Cyber Physical Systems – infrastructure, hardware sensing/actuation, data analysis, connectivity, and visualization. CPS vs IoT.

Modelling Cyber-Physical Systems: Overview of Continuous, Discrete, and Hybrid Models. Sensors and Actuators, Continuous Dynamics and Lyapunov Stability. Discrete Dynamics, Reactivity, and Termination.

Unit – II **(10)**

Designing Cyber Physical Systems

Cyber-Physical System Requirements Execution, Extended-State Machines Executions, Extended State Machines, Architectural Choices of the Cyber Physical Systems. Real-Time Operating Systems. Networking Embedded Systems. End-to-End System Design.

Unit – III **(10)**

Dynamical Systems Modelling

Cyber-Physical Systems (CPS) in the real world, Dynamical Systems: stability and performance, Different notions of stability, Controller Design techniques, Logic based system specification, Controller Synthesis as a logic problem. Real time sensing and communication for CPS, Real time task scheduling for CPS.

Unit – IV **(8)**

Analyzing Cyber Physical Systems

Real-Time Tasks and Worst-Case Execution Time. Finding and Proving Invariants. Convergence, Liveness, and Termination using Lyapunov Functions.

CPS Compute/Communicate/Scheduling, Real time scheduling theory, CAN bus scheduling, Wireless CPS. Packet drops and their effects on stability/performance, Delay/Deadline-miss aware control design.

Unit – V **(9)**

Security for Cyber Physical Systems

CPS Security and Cyber-attacks, Challenges in CPS Security – Quantifying Security and Risk Management for CPS – Analyzing Risk Management Frameworks (ISO 27000, NIST RMF, OCTAVE), ICS/SCADA Security, Embedded System Security, Distributed Control System Security.

Textbooks:

1. *"Principles of Cyber-Physical Systems,"* Rajeev Alur, MIT Press.
2. *"Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems,"* Peter Marwedel, Springer.

Reference Books:

1. *"Cyber-Physical Systems: From Theory to Practice,"* Danda B. Rawat, Joel J.P.C. Rodrigues, and Ivan Stojmenovic, CRC Press.
2. *"Cyber Security for Cyber Physical Systems,"* Saqib Ali, Taiseera Al Balushi, Zia Nadir, and Omar Khadeer Hussain, Springer.
3. *"Introduction to Embedded Systems: A Cyber-Physical Systems Approach,"* Edward Ashford Lee, and Sanjit Arunkuma Seshia, MIT Press.

Course Code: **CSE007G**	**Data Mining**	Credits: **04** L – **3** P – **2**

Course Outcomes (COs):

- Can define what data mining and data warehousing is and what it can be used for.
- Can determine the different steps followed in Data mining and pre-processing for data mining.
- Can describe and apply at least two of the algorithms used for generating frequent patterns and association rules for data mining.
- Can apply at least two of the Classification methods for data mining.

Unit – I **(10)**

Introduction: Knowledge Discovery from Data, Kinds of data and patterns that can be mined, Data mining functions, Applications of data mining, Major issues in data mining;
Knowing data: Types of attributes, Basic statistical descriptions of data, Measuring data similarity and dissimilarity (for each attribute type), cosine similarity, Data visualization;

Unit – II **(10)**

Data pre-processing: Data cleaning, Data integration, Data reduction, Data transformation, Data discretization.
Data warehousing: Basics, Operational database system verses data warehouse, Data warehousing architecture; Data warehouse modelling - Data cube, Schemas for multidimensional data models, Online analytical processing operations (roll-up, drill-down, slice and dice, pivot, etc.), OLAP server architectures, From OLAP to multidimensional data mining.

Unit – III **(8)**

Frequent pattern mining: Basic concepts, Market basket analysis, Frequent itemset mining algorithms (Apriori algorithm, FP-growth algorithm, Eclat algorithm), Mining closed and maximal patterns, Association rule mining, Correlation analysis, Interestingness Measures.

Unit – IV **(10)**

Classification: What is classification, Decision tree induction, Attribute selection measures, Tree pruning, Bayes' theorem, Naive Bayesian classification, Rule-based classification, Metrics for evaluating classifier performance, Ensemble methods, Classification by Backpropagation (Neural network-based classification), Support vector machines (SVMs), *k*-nearest-neighbor (*k*NN) classifier.

Unit – V **(7)**

Clustering: What is clustering; Partitioning methods for clustering: *k*-means clustering, Bisecting *k*-means, *k*-medoids clustering; Hierarchical methods: AGNES, DIANA, BIRCH; Density-based method: DBSCAN; Probabilistic model-based method: Expectation-Maximization algorithm.

Textbooks:

1. Jiawei Han, Micheline Kamber and Jian Pei, "Data Mining: Concepts and Techniques", Morgan Kaufmann Publishers, Elsevier, Third Edition.

Reference Books:

1. Aggarwal C, "Data Mining", Springer.
2. Pang-Ning Tan, Michael Steinbach and Vipin Kumar, "Introduction to Data Mining", Addison-Wesley, Pearson.
3. Mohammed J. Zaki, Wagner Meira Jr., "Data Mining and Analysis: Fundamental Concepts and Algorithms", Cambridge University Press.

Online Resources:

1. https://hanj.cs.illinois.edu/bk3/bk3_slidesindex.htm
2. https://onlinecourses.nptel.ac.in/noc21_cs06/preview
3. https://archive.nptel.ac.in/courses/106/105/106105174/
4. https://www.coursera.org/specializations/data-mining

List of Practicals

1. Apply data cleaning, integration, reduction, transformation, and discretization techniques on a *Census Income* dataset using WEKA or Anaconda (with Python libraries) software.

2. Design a simple data warehouse using a star or snowflake schema and perform basic OLAP operations *(roll-up, drill-down, slice, dice, pivot)*.

3. Implement the *Apriori* algorithm for frequent itemset mining on the *Groceries* dataset.

4. Implement the *FP-growth* and *Eclat* algorithm to perform frequent pattern mining on the *Groceries* dataset and compare the results with the *Apriori* algorithm.

5. Apply Naïve Bayesian algorithm for SPAM Email classification.

6. Implement Neural Network model for classifying *MNIST, FMNIST* datasets. Explore different activation functions and compare the results.

7. Implement Support Vector Machine (SVM) and K-Nearest Neighbor (KNN) classifiers on the *Iris* dataset.

8. Implement the K-Means clustering algorithm on the *Customer Segmentation* dataset to determine the optimal number of clusters.

9. Implement hierarchical clustering methods, students need to perform AGNES (Agglomerative Nesting) to create a dendrogram and interpret the hierarchical clustering results.

10. Implement *DBSCAN* algorithm for density-based clustering on *Geospatial data* to identify clusters and outliers, experimenting with different epsilon and minimum number of points *(minPts)* values.

Note: Implementation may be done using *Python* or *R*. The datasets may be downloaded from the *Kaggle/ UCI Machine Learning Repository*.

Course Code: **CSE008G**	**Digital Image Processing**	Credits: **03** L – **3** P – 0

Course Outcomes (COs):

- To learn and understand the fundamentals of digital image processing and its applications.
- To understand and implement the basics involved like image representation, resolution, interpolation, digitization, pixel relationships, etc.
- Learn and understand the Image Enhancement techniques.
- Understand different concepts and approaches of image restoration, and compression.
- To learn different concepts of colour image processing.

Unit – I **(5)**

INTRODUCTION:
What is digital image processing? The origins of digital image processing, Fundamental steps in digital image processing, components of an image processing system.

Unit – II **(8)**

DIGITAL IMAGE FUNDAMENTALS:
Image sensing and acquisition, Image sampling and quantization, basic relationships between pixels, linear and non-linear operations.

Unit – III **(12)**

IMAGE ENHANCEMENT IN THE SPATIAL DOMAIN:
Grey level transformations, histogram processing, enhancement using arithmetic/logic operations, spatial filtering, Image enhancement: smoothing and sharpening.
Image Restoration: A Model of the Image Degradation/Restoration Process. Inverse Filtering, Minimum Mean Square Error (Wiener) Filtering. Constrained Least Squares Filtering, Geometric Mean Filter, Geometric Transformations.

Unit – IV **(12)**

COLOUR IMAGE PROCESSING:
Fundamentals, models, colour transformations, smoothing and sharpening, colour segmentation and noise.
Image Segmentation: Detection of discontinuities, edge linking and boundary detection, thresholding, region-based segmentation, morphological watersheds.
Representation and description: Representation, boundary descriptors, regional descriptors, relational descriptors.

Unit – V **(8)**

Image Compression, Morphological Image Processing, Representation and Description.

Textbooks:

1. Rafael C Gonzalez, Richard E Woods, Digital Image Processing - Pearson Education
2. Rafael C Gonzalez, Richard E Woods, Digital Image Processing with MATLAB Pearson Education.

Reference Books:

1. William K Pratt, Digital Image Processing, John Willey
2. A.K. Jain, PHI, Fundamentals of Digital Image Processing, pearson Education.
3. Chanda & Majumdar, “Digital Image Processing and Analysis” , PHI.
4. Mark Nelson, Jean-Loup Gailly “The Data compression Book”, bpb Publications.

Course Code: **CSE009G**	**Ethical Hacking**	Credits: **04** L – **3** P – **2**

Course Outcomes (COs):

- Understand the core concepts related to information security with main focus on hardware and software vulnerabilities, threats, risks, malware, backdoors and key loggers.
- Learn the underlying principles and techniques associated with ethical hacking and penetration testing.
- Learn the tools associated with foot printing, concepts of denial of service and social engineering.
- Understand different approaches as well as related possible attacks of cryptography and steganography and system hacking.
- Learn the various legal, professional, and ethical issues likely to face the domain of ethical hacking and some appropriate tools and techniques associated with ethical hacking.

Unit – I **(9)**

Introduction to Security and Hacking

Types of Data Stolen from the Organizations, Elements of Information Security, Security Challenges, Vulnerability, Threat, Exploit, Risk, Hacker – Types of Hackers, Role of Security and Penetration Tester, Penetration Testing Methodology, Malicious Software (Malware), Types of Malware, Protection Against Malware, Intruder Attacks on Networks and Computers.

Unit – II **(9)**

Foot Printing and Social Engineering

Web Tools for Foot Printing, Conducting Competitive Intelligence, Google Hacking, Scanning–Types, Enumeration, Trojans and Backdoors, Virus and Worms, Key Loggers, Denial of Service, Sniffer, Social Engineering – shoulder surfing, Dumpster Diving, Piggybacking.

Unit – III **(7)**

Data Security

Physical Security – Attacks and Protection, Steganography – Methods, Attacks and Measures, Cryptography – Methods and Types of Attacks, Password Cracking, Wireless Hacking.

Unit – IV **(7)**

Network Vulnerabilities and Web Application Attacks

Networking and Computer Attacks, Network Sniffing –Types, IP Spoofing, ARP Poisoning, DNS Spoofing, Session Hijacking, Packet Sniffing using Wireshark and Burp suite, SQL Injection, Cross Site Scripting, Buffer Overflow, Email Hacking.

Unit – V **(8)**

Network Protection System and System Hacking

Routers, Firewall and Honeypots, Intrusion Detection System and Intrusion Prevention System, Vulnerability assessment using NMAP and Nessus, Use of Metasploit framework for password attack, privilege escalation etc. Windows Hacking, Linux Hacking, Introduction to Kali Linux.

Textbooks:

1. Michael T. Simpson, Kent Backman, James E. "*Corley, Hands -On Ethical Hacking and Network Defense*", Second Edition, CENGAGE Learning, 2010.
2. Baloch, Rafay. *Ethical hacking and penetration testing guide*. CRC Press, 2017.
3. Maurushat, Alana. *Ethical hacking*. University of Ottawa Press, 2019.

Reference Books:

1. Steven DeFino, Barry Kaufman, Nick Valenteen, "*Official Certified Ethical Hacker Review Guide*", CENGAGE Learning, 2009-11-01.
2. Patrick Engebretson, "*The Basics of Hacking and Penetration Testing: Ethical Hacking and Penetration Testing Made Easy*", Syngress Basics Series –Elsevier, August 4, 2011.
3. Whitaker & Newman, "*Penetration Testing and Network* Defense", Cisco Press, Indianapolis, IN, 2006.
4. Kimberly Graves, "*Certified Ethical Hacker Study Guide*", Wiley Publishing, Inc. 2010.
5. Kevin Beaver, "*Hacking for Dummies*", Wiley Publishing, Inc. 2013.

List of Experiments

1. Setup a honey pot and monitor the honey pot on network.
2. Write a script or code to demonstrate SQL injection attacks.
3. Create a social networking website login page using phishing techniques.
4. Write a code to demonstrate DoS attacks.
5. Install rootkits and study variety of options.
6. Study of Techniques uses for Web Based Password Capturing.
7. Install jcrypt tool (or any other equivalent) and demonstrate Asymmetric, Symmetric Crypto algorithm, Hash and Digital/PKI signatures studied in theory Network Security and Management
8. Implement Passive scanning, active scanning, session hizaking, cookies extraction using Burp suit tool

Course Code: **CSE010G**	**Go Language**	Credits: **03** L – **2** P – **2**

Course Outcomes (COs):

- learn Go programming language fundamentals.
- apply GoRoutines and channels to build massive parallel systems.
- Design and develop the projects using Go runtime.
- use Interfaces to simplify complex programs.
- build real-time apps using Golang

Unit – I **(5)**

Go overview: features of Go programming, Go environment setup: text editor, Go compilers, Go program structure, Go basic syntax: tokens, line separator, comments, identifiers, keywords.
Go data types: integers, float, numeric, variables: static type declaration, dynamic type declaration, mixed variable declaration, lvalue and rbalue, constants.

Unit – II **(4)**

Go operators: Arithmetic operators, relational operators, logical operators, bitwise operators, assignment operators, miscellaneous operators, operators precedence in Go.
Go decision making: if statement, else if statement, nested if statement, switch statement, select statement.

Unit – III **(7)**

Go loops: for loop, nested loop, loop control statements: break, continue, goto, infinite loop.
Go functions: defining a function, parameters, return type, calling a function, call by value, call by reference, Go scope rules: local variables, global variables, formal parameters, initializing local and global variables.

Unit – IV **(6)**

Go strings: creating strings, string length, concatenating strings
Go array: declaring array, initializing array, accessing array elements, multi-dimensional array, passing array to function, pointers: how to use pointers, nil pointer, array of pointers, pointer to pointer, passing pointer to function, structure.

Unit – V **(7)**

Go slice: len() and cap() functions, slice, subslicing, append() and copy() functions, Go- range, Go-map, Go-recursion, Go type casting, Go- interfaces, Go-error handling.

Textbooks:

1. An introduction to programming in Go by Caleb Doxsey
2. Programming in Go by Mark Summerfield
3. The Go programming language by David Chisnall

Reference Books:

1. Donovan, Alan AA, and Brian W. Kernighan. *The Go programming language*. Addison-Wesley Professional, 2015.

Course Code: **CSE011G**	**High Performance Computing**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Describe and use the main design techniques for sequential and parallel algorithms.
- Understand the architecture of modern CPU's and how this architecture influences the way programs should be written.
- Understand the differences among parallel and sequential algorithms solving the same problem and recognize which one is better under different conditions.
- Understand parallel algorithm for different architectures
- Describe and use basic and advanced parallel algorithms

Unit – I **(10)**

Introduction and overview, Applications of High Performance Computers, Basic Computer Organization, Pipelining, Memory Hierarchy, Multicore processors, Process Management, Multithreaded processors, Hyperthreading

Unit – II **(12)**

Taxonomy of parallel computing paradigms, Shared memory computers, UMA, NUMA, Amdhal's Law, Gustafson's Law, Creating parallel programs – Decomposition, Assignment, Orchestration, Mapping, Fork Join Model, pthreads, OpenMP, Example programs in OpenMP.

Unit – III **(15)**

Distributed memory, Characteristics of communication models, Latency, Bandwidth, Bisection Bandwidth, Diameter, Switching strategy, Flow control, Topologies, Ring, Torus, Meshes, Hypercubes, Trees, Butterflies, Routing algorithms, Randomized Routing, Performance Models, Bulk Synchronous Parallel, PRAM.
Introduction to MPI, Communications and collectives, Synchronization and barriers, Blocking and non-blocking communications, Enquires, Buffers, Example Programs in MPI.

Unit – IV **(10)**

Data Parallelism, Dense Matrix Multiplication, MapReduce, Scatter/Gather, Vector Processors for Data Parallelism, GPUs, Programming GPUs using CUDA, Heterogenous Computing – CPU GPU combination, Example programs in CUDA.

Unit – V **(10)**

Machine learning on HPC, Parallelism in machine learning, Training Neural Networks and Deep Neural Networks, Data Parallelization, Model Parallelization and Pipeline Parallelization, Applications in Computational Biology.

Textbooks:

1. Georg Hager, Gerhard Wellein, "*Introduction to High Performance Computing for Scientists and Engineers*", Chapman & Hall / CRC Computational Science series, 2011.
2. Ian Foster, *"Designing and Building Parallel Programs"*, Addison-Wesley, 1995

Reference Books:

1. Charles Severance, Kevin Dowd, "High Performance Computing", O'Reilly Media, 2nd Edition, 1998.

Online Resources:

2. OpenMP tutorial: http://www.llnl.gov/computing/tutorials/openMP
3. MPI online tutorial: https://hpc-tutorials.llnl.gov/mpi/
4. CUDA online tutorial: https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#

Course Code: **CSE012G**	**Programming in R**	Credits: **03** L – **2** P – **2**

Course Outcomes (COs):

- To understand the basics in R programming in terms of constructs, control statements, string functions,
- To identify and use available R packages and associated open-source software to meet given Scientific objectives
- To design and write efficient programs using R (and similar high-level languages) to perform routine and Specialized data manipulation/management and analysis tasks,
- To understand the use of R for Big Data analytics,
- To learn to apply R programming for Text processing,
- To appreciate and apply the R programming from a statistical perspective, and
- To understand the concepts of Interfacing R to other languages.

Unit – I **(8)**

Introduction to R and Data Structures: Introduction to R and RStudio; Basic syntax: variables, operators, data types; Vectors, lists, matrices, arrays, factors; Indexing and subsetting; Control structures: if, else, for, while, repeat; Writing user-defined functions; Apply family: apply(), lapply(), sapply(), tapply().

Unit – II **(10)**

Data Frames and Tidy Data Manipulation: Data frames: creation, access, and manipulation; Reading/writing CSV, Excel, JSON files (readr, readxl, jsonlite); Handling missing values and outliers; Tidyverse introduction; dplyr for data wrangling: filter(), mutate(), summarise(), arrange(), group_by(); tidyr: pivot_longer(), pivot_wider(), separate(), unite(); Using pipes (%>%) for clean code.

Unit – III **(10)**

Data Visualization with ggplot2: Grammar of graphics; Creating plots: bar, line, scatter, boxplots, histograms; Aesthetics and layers: aes(), geom_*(), labs(), theme(); Faceting, color mapping, legends; Exporting and saving plots; Comparing base R plots with ggplot2.

Unit – IV **(8)**

Statistical Computing and Simulation: Descriptive statistics and summary functions, Probability distributions: Normal, Binomial, Poisson; Sampling techniques; Hypothesis testing: t-test, chi-square, ANOVA (basic intro); Correlation and linear regression; Random number generation and Monte Carlo simulation.

Unit – V **(9)**

Reporting, Interfacing and Applications: R Markdown for report generation; Introduction to Shiny; String manipulation with stringr; Connecting to databases using DBI, RSQLite; Calling APIs using httr, basic web scraping using rvest; Interfacing with Python (reticulate) and using external packages; Final mini project / real dataset case study.

Textbooks:

1. *"R for Data Science,"* Hadley Wickham, and Garrett Grolemund, Shroff/O'Reilly.
2. *"The art of R Programming: A Tour of Statistical Software Design,"* Norman Matloff, No Starch Press, US.

Reference Books:

1. *"R for Everyone: Advanced Analytics and Graphics,"* Jared P. Lander, Pearson Addison-Wesley Professional.
2. *"Hands on Programming with R,"* Garrett Grolemund, Shroff/O'Reilly.

Online Resources:

1. https://www.coursera.org/learn/r-programming
2. https://www.udemy.com/course/the-comprehensive-programming-in-r-course/
3. https://www.coursera.org/learn/data-analysis-r
4. https://ggplot2.tidyverse.org
5. https://rstudio.com/
6. https://statmethods.net/
7. https://cran.r-project.org/

Course Code: **CSE013G**	**Robotics**	Credits: **04** L – **3** P – **2**

Course Outcomes (COs):

- Understand the key components of a robot.
- Ability to identify the sensory needs for a robot.
- Ability to design a good interface for the robot.
- Ability to evaluate the computational needs of a robot and selecting a proper system for designing a robot.
- Ability to develop a robotic solution for the given environment.

Unit – I **(7)**

Robot Introduction- Seven Criteria of Defining a Robot, Robot Controllers-Major Components, Robotics Middleware Basics. Historical Perspective, Specifications of Robots, Classifications of robots, Work envelope, Flexible automation versus Robotic technology, Applications of Robots.

Unit – II **(10)**

Use of Sensors and Sensor Based System in Robotics, Machine Vision System, Description, Sensing, Digitizing, Image Processing and Analysis, segmentation- Thresholding- edge detection- binary morphology – grey morphology and Application of Machine Vision System, Robotic Assembly Sensors and Intelligent Sensors, visual servo-control.

Unit – III **(10)**

Object recognition, Approaches to Object Recognition, Recognition by combination of views – objects with sharp edges, using two views only, using a single view, use of depth values. Histogram of oriented gradients (HOG). Image classification algorithms.

Unit – IV **(10)**

Programming the Robot's Sensors, Programming the Actuators, Building Robot's Softbot, ROS Basics-ROS Equation, ROS Architecture and Concepts, ROS Filesystem Level, ROS Computation Graph Level, Ubuntu Linux for Robotics-Ubuntu Graphical User Interface, Shell Commands, C++ and Python for Robotic Programming- Basic Concepts with Examples. Creating ROS Workspace and Package, Using ROS Client Libraries, Programming Embedded Board using ROS-Interfacing Arduino with ROS, ROS on a Raspberry Pi.

Unit – V **(5)**

Applications of robotics in active perception, medical robotics, autonomous vehicles, and other areas, Quadcopter. Drones.

Textbooks:

1. Peter Corke, Robotics, Vision and Control: Fundamental Algorithms, Springer Tracts in Advanced Robotics,Volume 118, Second Edition, 2016.
2. D. Patranabis, "Sensors and Actuators", 2nd Edition, PHI Learning, New Delhi, India, 2013.
3. Jonathan Cacace; Lentin Joseph, Mastering ROS for Robotics Programming: Design, build, and simulate complex robots using the Robot Operating System, 2nd Edition, Packt Publishing, 2018.

Reference Books:

1. Robert J. Schilling, "Fundamentals of Robotics Analysis and Control", PHI Learning, 2009.
2. Deb S R and Deb S, "Robotics Technology and Flexible Automation", Tata McGraw Hill Education Pvt. Ltd, 2010.
3. D. Patranabis, "Sensors and Transducers", 2nd Edition, PHI Learning Pvt. Ltd., New Delhi, India, 2011.
4. Hughes, C. and Hughes, T., Robot programming: a guide to controlling autonomous robots. Que Publishing, 2016

Course Code: **CSE014G**	**Simulation Technologies**	Credits: **03** L – **2** P – **2**

Course Outcomes (COs):

- Understand the concept of simulating the real-world problems through software tools.
- Understand the basics of MATLAB and apply the methods of MATLAB for problem solving.
- Design and simulate various network topologies used in computer networks.
- Using Cooja Contiki for implementing IoT based Networks.

Unit – I **(10)**

MATLAB: Introduction to MATLAB, Identify and use various windows, Data types, Rules about variable names, Express numbers in either floating-point or scientific notation, save a series of commands. Built in functions, elementary math functions (common math functions, rounding functions, discrete mathematics functions, trigonometric functions), data analysis functions (maximum and minimum, mean and median, sums and products), sorting functions, random numbers.

Unit – II **(6)**

Relational and Logical Operators, If-else statements, Switch-case statements, For loop, While loop, Special commands (Break and continue), Script file. Defining matrices, accessing matrix elements, Matrix Operations and Functions. Two-Dimensional Plots, Three Dimensional Plots, Saving Your Plots.

Unit – III **(10)**

Image Processing Using MATLAB: Read and Store Various Image Formats, Mathematical Operations on Image, Channel Separation, Edge Detection, Thresholding and Image Segmentation, Histogram Equalization, Noise and Filtering.

Introduction to Video Processing Using MATLAB.

Unit – IV **(6)**

Cooja: Introduction to cooja, Running Cooja Simulator, creating a new simulation, creating a new mote type, adding motes and running the simulation, saving simulation file, Debugging with Cooja Simulator.

Unit – V **(10)**

NS2: Introduction to Open-Source Software, Introduction to Network Simulator – 2 (NS2), TCL Scripting, TCL Script Components, Introduction to NAM, TRACE, XGRAPH etc., Wired Scenarios - Writing your first TCL Script, Network Dynamics etc., Simulation of Different Network Topologies – BUS topology, RING Topology, STAR topology etc. Creating Wireless Scenarios, Wired-cum-Wireless Scenarios etc.

Textbooks:

1. Holly Moore, "MATLAB for Engineers", Pearson
2. Introduction to Network Simulator NS2 by Teerawat Issariyakul and Ekram Hossain, Springer-Verlag New York Inc.; 2nd ed. 2012 edition (3 March 2014)

Reference Books:

1. Stephen.J.Chapman, "Programming in MATLAB for Engineers", Cengage Learning, 2011.
2. Bansal R.K, Goel A.K., Sharma M.K., "MATLAB and its Applications in Engineering", Pearson Education, 2012.

Online Resources:

1. https://github.com/contiki-os/contiki/wiki/An-Introduction-to-Cooja

Course Code: **CSE017G**	**Wireless Communication**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Comprehend the knowledge of communication and data communication
- Understand the basic cellular concepts including capacity expansion in wireless communication systems, Shadowing and Fading.
- Distinguish and compare the performance of different Multiple Access Schemes
- Understand the basic principles of GSM, CDMA, WCDMA technologies
- Understand the latest trends in wireless communication systems including 4G, 5G, WPAN, LTE and A-LTE.

Unit – I **(8)**

Introduction to Wireless Communication Systems: Evolution of mobile radio communications, paging systems; Cordless telephone systems; overview of generations of cellular systems 2g/3g/4g/5g, Introduction to Personal Communication Services (PCS): PCS architecture.

Unit – II **(8)**

Introduction to cellular system, Frequency reuse, Cochannel and Adjacent channel Interference, Increasing the capacity of cellular systems, Introduction to Wireless Channels and Diversity: Fast Fading Wireless Channel Modeling, BER Performance in Fading Channels.

Unit – III **(8)**

2G Networks: Second generation, digital, wireless systems: overview of GSM, IS_136 (D-AMPS), IS-95 CDMA. Global system for Mobile Communication (GSM) system overview: GSM Architecture, Introduction to 2.5G, GPRS and edge.

Unit – IV **(8)**

Voice signal processing and coding. Spread Spectrum Systems- Cellular code Division Access Systems-Principle, Power Control, effects of multipath propagation on code division multiple access. Mobility Management (Handoff), Erlang Capacity.

Unit – V **(8)**

Overview of Third Generation (3G) Mobile Services: Introduction to International Mobile Telecommunications 2000 (IMT 2000) vision, Wideband Code Division Multiple Access (W-CDMA), and CDMA 2000, Quality of services in 3G. Introduction to: UWB, 4G and 5G, Cognitive Radio, Network on a chip.

Textbooks:

1. William Stallings “Wireless Communications & Networks”
2. Theodore S. Rappaport, “Wireless Communication- Principles and practices,” 2nd Ed., Pearson Education Pvt. Ltd, 5th Edition, 2008.

Reference Books:

1. T.L.Singhal “Wireless Communication”, Tata McGraw Hill Publication.
2. Jochen Schiller, “Mobile communications,” Pearson Education Pvt. Ltd., 2002.
3. Yi –Bing Lin & Imrich Chlamatac, “Wireless and Mobile Networks Architecture,” John Wiley & Sons,2001.
4. Raj Pandya, “Mobile & Personnel communication Systems and Services”, Prentice Hall India, 2001.