B. TECH IN COMPUTER SCIENCE AND ENGINEERING

I SEMESTER (2023-27 BATCH)

Sl. No.	Course Code	Course Title	Hours per week				Credits	Tools / Languages	Course Type
			L	T	P	S	C		
1	UE23CS151A	Python for Computational Problem Solving	4	0	2	5	5	Python	FC- Lab Integrated

II SEMESTER (2023-27 BATCH)

Sl. No.	Course Code	Course Title	Hours per week				Credits	Tools / Languages	Course Type
			L	T	P	S	C		
1	UE23CS151B	Problem Solving with C	4	0	2	5	5	C Programming Language, GCC Compiler, GDB Debugger.	FC- Lab Integrated

III SEMESTER (2022-26 BATCH)

Sl. No.	Course Code	Course Title	Hours per week				Credits	Tools / Languages	Course Type
			L	T	P	S	C		
1	UE22CS251A	Digital Design and Computer Organization	4	0	2	5	5	Icarus, Verilog Simulator, GTKWave waveform viewer	CC- Lab Integrated
2	UE22CS252A	Data Structures and its Applications	4	0	2	5	5	C Programming Language	CC-Lab Integrated
3	UE22CS241A	Statistics for Data Science	4	0	0	4	4	Jupyter Notebook**,** Python, Pandas, Matplotlib, Scipy, Seaborn, BeautifulSoup, Numpy, Scikit learn.	CC- Independent
4	UE22CS242A	Web Technologies	4	0	0	4	4	HTML, CSS, JavaScript, MERN Technologies.	CC- Independent
5	UE22CS243A	Automata Formal Languages and Logic	4	0	0	4	4	JFLAP	CC- Independent
6	UE23MA221A*	Bridge Course Mathematics –I (Applicable for the Lateral Entry Students)	2	0	0	2	0		FC- Independent
Total			**20/22**	**0**	**4**	**22/ 24**	**22**		
* - Audit Course									

IV SEMESTER (2022-26 BATCH)

Sl. No.	Course Code	Course Title	Hours per week				Credits	Tools / Languages	Course Type
			L	T	P	S	C		
1	UE22CS251B	Microprocessor and Computer Architecture[%]	4	0	2	5	5	ARM Simulator.	CC-Lab Integrated
2	UE22CS252B	Computer Networks	4	0	2	5	5	Wireshark, Python	CC-Lab Integrated
3	UE22CS241B	Design and Analysis of Algorithms	4	0	0	4	4	C-Programming Language, GCC Compiler	CC-Independent
4	UE22CS242B	Operating Systems[@]	4	0	0	4	4	C, Linux/Unix OS for system call implementation.	CC-Independent
5	UE22MA241B	Linear Algebra	4	0	0	4	4		CC-Independent
6	UE23MA221B*	Bridge Course Mathematics – II (Applicable to Lateral Entry Students)	2	0	0	2	0		FC-Independent
Total			**20/22**	**0**	**4**	**22/ 24**	**22**		

Note: Desirable Knowledge - [%]UE22CS251A, [@]UE22CS252A.
* - Audit Course

V SEMESTER (2021-25 BATCH)

Sl. No.	Course Code	Course Title	Hours per week				Credits	Tools / Languages	Course Type
			L	T	P	S	C		
1	UE21CS351A	Database Management System	4	0	2	5	5	MySQL 13.3, Python, Erwin or any other tool.	CC-Lab Integrated
2	UE21CS352A	Machine Intelligence*	4	0	2	5	5	Pytorch	CC-Lab Integrated
3	UE21CS341A	Software Engineering	4	0	0	4	4	GitHub, MS Project/GanttPro/Jira, Jenkins.	CC-Independent
4	UE21CS342AAX	Elective I	4	0	0	4	4		EC
5	UE21CS343ABX	Elective II	4	0	0	4	4		EC
Total			**20**	**0**	**4**	**22**	**22**		
Elective – I									
6	UE21CS342AA1	Advanced Algorithms%	4	0	0	4	4	C or C++ Programming Language	EC-Independent
7	UE21CS342AA2	Data Analytics&	4	0	0	4	4	R and Python Programming Languages	EC-Independent
8	UE21CS342AA3	Internet of Things$	4	0	0	4	4	Python, Embedded-C, Cloud Platforms, Single Board Computers	EC-Independent
9	UE21CS342AA4	Applied Cryptography	4	0	0	4	4	SEED labs, gpg, Python Programming Language	EC-Independent
10	UE21CS342AA5	Augmented and Virtual Reality !!!	4	0	0	4	4	C/ C++/ JAVA/ Python using OpenGL.	EC-Independent
11	UE21CS342AA6	Human Computer Interaction	4	0	0	4	4	Figma, Adobe Creative Cloud	EC-Independent
Elective – II									
12	UE21CS343AB1	Unix System Programming	4	0	0	4	4	C or C++ Programming Language, Unix OS.	EC-Independent
13	UE21CS343AB2	Big Data $	4	0	0	4	4	Hadoop, HDFS Spark, Streaming spark, HIVE, Hbase, Mllib	EC-Independent
14	UE21CS343AB3	Graph Theory, and its Applications !	4	0	0	4	4	Python, Neo4j, NetworkX	EC-Independent
15	UE21CS343AB4	Bio-inspired Computing %	4	0	0	4	4	Matlab	EC-Independent
16	UE21CS343AB5	Advanced Computer Networks%%	4	0	0	4	4	GNS3, Cisco Packet Tracer.	EC-Independent

17	UE21CS343AB6	Computer Network Security[%%]	4	0	0	4	4	SEED Ubuntu VM, Wireshark, Snort, Netwox, Scapy.	EC-Independent

Note: Desirable Knowledge
Core : [*]- UE21CS241A, UE21MA241B, UE21CS241B.
Elective I : [%]- UE21CS241B, [&]- UE21CS241A, [$] - UE21CS151A, UE21CS151B, UE21EC141A, UE21CS252B, [!!!]- UE21CS252A.
Elective II : [$]-UE21CS252A, UE21CS241B, [!]- UE21CS151B, UE21CS252A, [%]- UE21CS241B, [%%]- UE21CS252B.

ELECTIVES TO BE OPTED FOR SPECIALIZATION

Sl. No.	SPECIALIZATION	ELECTIVE – I	ELECTIVE – II
A	System and Core Computing(SCC)	UE21CS342AA1, UE21CS342AA5.	UE21CS343AB1, UE21CS343AB2, UE21CS343AB3.
B	Machine Intelligence and Data Science(MIDS)	UE21CS342AA2, UE21CS342AA3, UE21CS342AA5, UE21CS342AA6	UE21CS343AB2, UE21CS343AB3, UE21CS343AB4
C	Cyber Security & Connected Systems (CSCS)	UE21CS342AA3, UE21CS342AA4, UE21CS342AA5.	UE21CS343AB5, UE21CS343AB6.

VI SEMESTER (2020-24 BATCH)

Sl. No	Course Code	Course Title	Hours per week				Credits	Tools / Languages	Course Type
			L	T	P	S	C		
1	UE21CS351B	Cloud Computing[@@]	4	0	2	5	5	Amazon AWS (or equivalent), AWS Skill Builder, AWS Educate, Qwiklabs, Docker, Kubernetes, Jenkins, Zookeeper, Github, NoSQL database, Flask, Python, GoLang	CC
2	UE21CS352B	Object Oriented Analysis & Design using Java	4	0	2	5	5	Star UML, Java Programming Languages	CC
3	UE21CS341B	Compiler Design[!]	4	0	0	4	4	Lex and Yaac	CC
4	UE21CS342BAX	Elective III	4	0	0	4	4		EC
5	UE21CS343BBX	Elective IV	4	0	0	4	4		EC
6	UE21CS320A	Capstone Project Phase-1	0	0	8	8	2		PW
Total			**20**	**0**	**12**	**30**	**24**		

Elective – III

7	UE21CS342BA1	Generic Programming in C++[#]	4	0	0	4	4	Compiler Explorer (godbolt.org), Cpp Insights (cppinsights.io), Quick-bench (quick-bench.com), wandbox (wandbox.org), Visual Studio Code (code.visualstudio.com), Clang-Tidy, Clang-Format, Cppcheck, Valgrind, Conan (conan.io)	EC-Independent
8	UE21CS342BA2	Algorithms for Information Retrieval and Intelligence Web[??]	4	0	0	4	4	Scikit, Tensorflow, Solr, Lucene Search Engines/ Python Programming Languages	EC-Independent
9	UE21CS342BA3	Image Processing and Computer Vision I **	4	0	0	4	4	MatLab, Python Programming Languages	EC-Independent
10	UE21CS342BA4	Natural Language Processing [##]	4	0	0	4	4	Tensorflow, Scikit Learn, Python 3.x. CoreNLP, Natural Language Toolkit (NLTK), TextBlob,Gensim, SpaCy,PyTorch-NLP , OpenNLP.	EC-Independent
11	UE21CS342BA5	BlockChain [!]	4	0	0	4	4	Solidity, Remix, Ganache, Metamask	EC-Independent
12	UE21CS342BA6	Digital Forensics and Incident Responses	4	0	0	4	4	Open source tools on Forensics.	EC-Independent
13	UE21CS342BA7	Digital Twin and eXtended Reality [!]	4	0	0	4	4	C/ C++/ JAVA/ Python using OpenGL.	EC-Independent
Elective – IV									
14	UE21CS343BB1	Heterogeneous Parallelism [!!!]	4	0	0	4	4	pthread, OpenMP CUDA, openCL.	EC-Independent
15	UE21CS343BB2	Topics in Deep Learning [##]	4	0	0	4	4	Pytorch.	EC-Independent
16	UE21CS343BB3	Database Technologies***	4	0	0	4	4	MySQL, postgres, Oracle, Apache Spark, Apache Kafka, Amazon Kinesis.	EC-Independent
17	UE21CS343BB4	Meta Learning for Graph Abstraction[%%%]	4	0	0	4	4	NetworkX for statistical features of graphs,Tensor flow Keras and Scikit Learn for traditional graph ML, and Pytorch Geometric for Graph Neural Networks.	EC-Independent

18	UE21CS343BB5	Wireless Mobile Networking%%	4	0	0	4	4	Wireshark, Claynet, Cisco Packet Tracer.	EC-Independent
19	UE21CS343BB6	Information Security	4	0	0	4	4	SSEED Labs VM, Scapy, Burp Suite, Metasploit, Nmap, etc.	EC-Independent
20	UE21CS343BB7	Mobile and Autonomous Robotics	4	0	0	4	4	C, C++, Python, ROS	EC-Independent
21	UE21CS343BB8	Security for Internet of Things.	4	0	0	4	4	Wireshark, Yersinia, VoIP Hopper, Bettercap,aircrack-ng	EC-Independent

Note: Desirable Knowledge
Core: !UE21CS252A, UE21CS243A @@UE21CS241B,UE21CS252B.

Elective III: # - UE21CS151A, UE21CS151B, UE21CS252A, UE21CS252B, UE21CS241B, ?? - UE21CS241B, UE21MA241B, UE21CS352A **- UE21CS241B, ## - UE21CS352A, ! - UE21CS252A.

Elective IV: !!!-UE21CS151B, UE21CS251B, ##-UE21CS352A, *** - UE21CS351A, %%% - UE21CS343AB3, UE21CS352A, %% - UE21CS252B.

ELECTIVES TO BE OPTED FOR SPECIALIZATION

Sl. No.	SPECIALIZATION	ELECTIVE – III	ELECTIVE – IV
A	System and Core Computing(SCC)	UE21CS342BA1, UE21CS342BA2, UE21CS342BA7.	UE21CS343BB1, UE21CS343BB3.
B	Machine Intelligence and Data Science(MIDS)	UE21CS342BA2, UE21CS342BA3, UE21CS342BA4, UE21CS342BA7.	UE21CS343BB2, UE21CS343BB4, UE21CS343BB7.
C	Cyber Security & Connected Systems (CSCS)	UE21CS342BA5, UE21CS342BA6, UE21CS342BA7.	UE21CS343BB5, UE21CS343BB6, UE21CS343BB7, UE21CS343BB8.

VII SEMESTER (2020-24 BATCH)

SI No.	Course Code	Course Title	Hours / week				Credits	Course Type
			L	T	P	S	C	
1	UE20CS461A	Capstone Phase-2	0	0	24	24	6	PW
2	U20CS461AX`	Special Topic/ Directed Independent Study - Swayam/MOOC	2/4/4			2	6	ST
Total			**2/4/4**	**2/4/4**	**24**	**26**	**12**	

VIII SEMESTER (2020-24 BATCH)

Sl. No.	Course Code	Course Title	Hours / week				Credits	Course Type
			L	T	P	S	C	
1	UE20CS421B	Capstone Phase-3	0	0	8	8	2	PW
2	UE20CS461XB/ UE20CS462XB	Internship/Special Topic/ Directed Independent Study – Swayam/MOOC.	2	0	2	2	6	ST
Total			**2**	**0**	**10**	**10**	**8**	

NOTE:
2 weeks of Internship = 1 credit.
In case the student does not have an internship, he/she may earn those credits using Sl no 2.

UE23CS151A: Python for Computational Problem Solving (4-0-2-5-5)

Python is an easy to learn, general-purpose, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

Course Objectives:

- Learn the syntax and semantics of Python programming language.
- Illustrate the process of structuring the data using lists, tuples, sets and dictionaries.
- Demonstrate the use of built-in functions to navigate the file system.
- Learn various paradigms of programming and implement the Object Oriented Programming concepts in Python.

Course Outcomes:

At the end of this course students will be able to,

- Program effectively using the Python language.
- Identify the methods to create and manipulate lists, tuples and dictionaries.
- Discover commonly used operations involving file system.
- Think using different paradigms of programming and interpret the concepts of Object-Oriented Programming as used in Python.

Course Contents:

Unit 1: Introduction

Computation Problem Solving-Limits of Computational Problem Solving - Computer Algorithm - Computer Hardware. Digital Computer - Operating System- Limits of IC technology - Computer Software - Syntax, semantics and program translation ,Introduction to Python Programming Language, IDLE Python Development Environment, Output function - variables, types and id, input function , operators and expressions, Control structures.

12 Hours

Unit 2: Collections & Functions

Lists, Tuples, Dictionaries, Sets, Strings and text file manipulation: reading and writing files. Functions: Definition, call Positional and keyword parameter, Default parameters, Variable number of arguments.

16 Hours

Unit 3: Functions, GUI, Modules, Testing and Debugging

Recursion, Call-backs, Closure, Decorators, generators. Graphical User Interface with Tinkter package- Different geometric methods – Tk, mainloop, Creating simple GUI - buttons, canvas, check button, labels, entry fields, dialogs Widgets - sizes, fonts, colours layouts, nested frames, Modules - import mechanisms Testing- Pytest , Function testing with Doctest, pdb debugger commands.

14 Hours

Unit 4: Functional & Object Oriented Programming

Lambda function, Map, filter, and reduce, max, min, Zip, list comprehension. Classes and objects - inheritance, polymorphism, iterators, Error handling & Exceptions - try, except and raise, exception propagation.

14 Hours

Lab / Hands-on:

1: Programs on Input Output Functions, Operators and Expressions, Usage of Libraries and Control Structures.
2: Programs on Collections (Lists, Tuples, Sets , Dictionaries , Strings).
3: Programs on Files and File manipulations.
4: Programs on Functions.
5: Programs on Functional Programming.
6: Programs on Object Oriented Programming.

Tools / Languages: Python.

Text Book(s):

1: “Introduction to Computer Science Using Python: A Computational Problem- Focus”, Charles Dierbach,Wiley India Edition,John Wiley, 2015.

Reference Book(s):

1: “Learn python Programming”, Fabrizio Romano, 2nd Edition, Packet Publishing,2018.
2: “Fundamentals of Python: First Programs”, Kenneth A. Lambert, Cengage, 2019.
3: “Introduction to Computation and Programming Using Python: With Application to Understanding Data”, John V. Guttag, MIT Press, MIT with Library of Congress Cataloguing- in-PublicationData, 2016.

UE23CS151B – Problem Solving with C (4-0-2-5-5)

In Problem Solving with C Course, will be learning to solve common types of computational problems, by analyzing the given problem statement and developing an algorithm to solve the given problem. Then make use of c language constructs to develop well-structured programs.

Course objectives:

- Acquire knowledge on how to solve relevant and logical problems using computing machine
- Map algorithmic solutions to relevant features of C programming language constructs
- Gain knowledge about C constructs and its associated eco-system
- Appreciate and gain knowledge about the issues with C Standards and its respective behaviours
- Get insights about testing and debugging C Programs

Course outcomes:

At the end of the course, the student will be able to

- Understand and apply algorithmic solutions to counting problems using appropriate C Constructs
- Understand, analyse and apply text processing and string manipulation methods using C Arrays, Pointers and functions
- Understand prioritized scheduling and implement the same using C structures
- Understand and apply sorting techniques using advanced C constructs
- Understand and evaluate portable programming techniques using preprocessor directives and conditional compilation of C Programs

Course Content:

Unit 1 : Introduction

Introduction to Programming, Salient Features of 'C', Program Structure, Variables, Data Types & range of values, Operators and Expressions, Control Structures, Input/ Output Functions, Language specifications-Behaviors.

10 Hours

Unit 2 : Counting, Text Processing and String manipulation

Arrays – 1D and 2D, Functions, Storage classes, Pointers, Strings, String Manipulation Functions & errors.

18 Hours

Unit 3 : Dynamic Memory Management, Structures and File Handling: Dynamic Memory Allocation and Deallocation functions & errors, Structures, #pragma, File IO using redirection, File Handling functions, Searching, Sorting, Combination of Structures, Arrays and Pointers, Call-back, Code Review .

18 Hours

Unit 4 : Queuing and Portable Programming

Lists, Priority Queue, Enums, Unions, Bit Fields, Pre-Processor Directives, Conditional Compilation, Code Review

10 Hours

Tools/ Languages: C Programming Language, GCC Compiler, GDB Debugger.

Lab/ Hands-on: 14 Hours

1: Programs on IO, Operators and Control structures.
2: Programs on Arrays and Pointers, Functions using arrays and pointers
3 : Programs on Strings, Structures and Dynamic Memory Management functions
4 : Programs on Structures and Array of structures
5 : Introduction to GDB and Debugging using GDB.
6 : Programs on Inclusion of files using redirection operator.
7 : Programs using File handling functions in C - Sorting and Searching using Array of pointers to structures.
8 : Implementation of Ordered List.
9 : Implementation of Priority based scheduling.
10 : Programs on unions, enums and Preprocessor directives.

Text Book(s) :

1: “The C Programming Language”, Brian Kernighan and Dennis Ritchie, Prentice Hall PTR, 2nd Edition, 1988.

Reference Book(s):

1: “How To Solve It By Computer”, R G Dromey, Pearson, 2011.
2: “C Programming: A Modern Approach”, 2nd Edition by K.N. King. W. W. Norton & Company.
3: “Learn C the Hard Way”: Zed Shaw’s Hard Way Series, 1st Edition.
4: “C Puzzle Book” by Alan R. Fever, Pearson Education.
5: “Expert C Programming: Deep C Secrets” by Peter Van Der Linden, Pearson Education.

UE22CS251A : Digital Design and Computer Organization (4-0-2-5-5)

This course focuses on the structure, design and operation of a computer system at different levels of abstraction. The digital design part of the course describes low level digital logic building blocks while the computer organization part explains the structure and operation of microprocessors.

Course Objectives:

- Fundamental (combinational and sequential) building blocks of digital logic circuits.
- Design of more complex logic circuits such as adders, multipliers and register files.
- Design of Finite State Machines based on problem specification.
- Construction, using above logic circuits, of a microprocessor, and its functioning at the clock cycle level.
- Use of studied digital building blocks to construct more complex systems.

Course Outcomes:

At the end of this course, the student will be able to,

- Perform analysis of given synchronous digital logic circuit.
- Design and implement small to medium scale data path logic circuits from given specification.
- Design and implement control logic using Finite State Machines.
- Understand hardware level microprocessor operation, providing a foundation for the higher layers.
- Utilize the concepts and techniques learnt to implement complex digital systems.

Course Contents:

Unit 1: Combinational Logic & Sequential Logic Design

Introduction, Boolean Functions, Truth tables, K-Maps, Adder/Subtractor, Overflow, Muxes, Decoders, Shifters , Latches, Flip-flops,

14 Hours

Unit 2: Sequential and Arithmetic Circuits

Synchronous logic design, Finite State Machines, FSM examples, Counters, Memory arrays

14 Hours

Unit 3 : Arithmetic Circuit and Architecture

Carry-look ahead and Prefix adders, Shift/add multiplier/divider, Wallace tree multiplier, Floating point,

14 Hours

Unit 4 : Micro Architecture

Assembly Language, Machine Language. Addressing Modes, Performance analysis, Single-cycle, Multi-cycle processor data path and control, Systolic array matrix multiply, PC I/O Systems.

14 Hours

Lab/Hands-on : 14 Hours

1: Verilog Basics- Basic gates, Adder/ Subtractor (One bit & n-bit).
2: Design of an ALU.
3: Design of a Register File.
4: Design of a Datapath (Integration of ALU and Register File).
5: Design of a Program Counter.
6: Design of a Control Logic.

Tools/Languages: Icarus, Verilog Simulator, GTKWave waveform viewer.

Text Book (s):

1: “Digital Design and Computer Architecture”, David Money Harris, Sarah L Harris, 2nd Edition, Morgan Kaufmann, 2012.

Reference Book(s):

1: “Digital Design”, M Morris Mano, Michael D Ciletti, 6th Edition, Pearson, 2018.

2: “Computer Organization and Design”, David A Patterson, John L Hennessey, 5th Edition, Elsevier, 2016.
3: “Computer Organization and Design”, Carl Hamacher, Safwat Zaky, Zvonko Vranesic, 5th Edition, Tata McGraw-Hill, 2011.

UE22CS252A : Data Structures and its Applications (4-0-2-5-5)

This course introduces abstract concepts, shows how the concepts are useful for problem solving and then shows how the abstractions can be made concrete by using a programming language. Equal emphasis is placed on both the abstract and the concrete versions of a concept so that the student learns about the concept itself, its implementation and its application.

Course Objectives:

- Basic approaches and mindsets for analyzing and designing data structures and construct essential skills of data structures to store and retrieve data quickly and usefully (efficiently).
- Usage the of different data structures that support different set of operations which are suitable for different type of tasks.
- Implement how to insert, delete, search and modify data in any given data structures- Stack, Queue, List, Tree, heap, Graphs.
- Implement a given application using the available data structure.

Course Outcomes:

At the end of this course, the student will be able to,

- Choose relevant data structures for any given application appl
- Apply the required to implement any data structure.
- Appropriate data structure in competitive programming.
- Design and develop efficient software systems with good knowledge of data structures.

Course Content:

Unit 1: Linked List and Stacks

Review of C , Static and Dynamic Memory Allocation. Linked List: Doubly Linked List, Circular Linked List – Single and Double, Multilist: Introduction to sparse matrix (structure). Skip list Case study: Dictionary implementation using skip list Stacks: Basic structure of a Stack, Implementation of a Stack using Arrays & Linked list. Applications of Stack: Function execution, Nested functions, Recursion: Tower of Hanoi. Conversion & Evaluation of an expression: Infix to postfix, Infix to prefix, Evaluation of an Expression, Matching of Parenthesis.

15 Hours

Unit 2: Queues and Trees

.Queues & Dequeue: Basic Structure of a Simple Queue, Circular Queue, Priority Queue, Dequeue and its implementation using Arrays and Linked List. Applications of Queue: Case Study – Josephus problem, CPU scheduling- Implementation using queue (simple /circular). General: N-ary trees, Binary Trees, Binary Search Trees and Forest: definition, properties, conversion of an N-ary tree and a Forest to a binary tree. Implementation of BST using arrays and dynamic allocation : Insertion and deletion operations, Traversal of trees: Preorder, Inorder and Postorder.

13 Hours

Unit 3: Application of Trees and Introduction to Graphs

Implementation of binary expression tree., Threaded binary search tree and its implementation. Heap: Implementation using arrays. Implementation of Priority Queue using heap - min and max heap. Applications of Trees and Heaps: Implementation of a dictionary / decision tree (Words with their meanings). Balanced Trees: definition, AVL Trees, Rotation, Splay Tree, Graphs: Introduction, Properties, Representation of graphs: Adjacency matrix, Adjacency list. Implementation of graphs using adjacency matrix and lists. Graph traversal methods: Depth first search, Breadth first search techniques. Application: Graph representation: Representation of computer network topology.

14 Hours

Unit 4: Applications of Graphs , B-Trees, Suffix Tree and Hashing

Application of BFS and DFS: Connectivity of graph, finding path in a network. Case Study –Indexing in databases (B Tree: K-way tree)- Insertion and deletion operations with examples. Suffix Trees: Definition, Introduction of Trie Trees, Suffix trees. Implementations of TRIE trees, insert, delete and search operations. Hashing: Simple mapping / Hashing: hash function, hash table, Collision Handling: Separate Chaining & Open Addressing, Double Hashing, and Rehashing. Applications: URLs decoding, Word prediction using TRIE trees / Suffix Trees.

14 Hours

Lab / Hands-on : 14 Hours

1:Implementation of singly linked list and advanced operations.
2: Implementation of circular linked list and an application based on it.
3: Implementation of stack and its application for prefix and postfix expression evaluation.
4: Implementation of Binary expression tree from given prefix expression and evaluate it.
5: Implementation of Graph Data structure and application based on it.
6: Implementation of Hashing Techniques.

Tool/ Languages: C Programming Language

Text Book(s):
1: "Data Structures using C / C++" , Langsum Yedidyah, Moshe J Augenstein, Aaron M Tenenbaum Pearson Education Inc, 2nd edition, 2015.

Reference Book(s):
1: "Data Structures and Program Design in C", Robert Kruse, Bruce Leung, C.L Tondo, Shashi Mogalla, Pearson, 2nd Edition, 2019.

UE22CS241A: Statistics for Data Science (4-0-0-4-4)

Data Science is the study of data. It is about extracting, analyzing, visualizing, managing and storing data to create insights. This course covers both Descriptive statistics to understand the data and the inferential statistics which seeks to infer something about a population on the basis of a statistical sample and build a simple linear regression model.

Course Objectives:

- Provide insights about the basic roles of a Data Scientist. Develop a greater understanding of the importance of Data Visualization techniques.
- Provide students with knowledge of Random Variables and Distributions of it.
- Provide students with knowledge of Confidence Intervals and its importance. Make inferences about the population parameters using sample data.
- Make inferences about the population parameters using sample data and test it to draw meaningful conclusions.
- Provide an understanding on the importance and techniques of predicting a relationship between the two sets of data and determine the goodness of fit model.

Course Outcomes:

At the end of this course, the student will be able to,

- Use Python and other tools to extract, clean and analyze data from several data sources (files, web) and analyze an extremely large data set and perform exploratory data analysis to extract meaningful insights.
- Analyze a real-world problem and solve the same with the knowledge gained from various distributions study.
- Compute Confidence Intervals.
- Develop and test a hypothesis about the population parameters to draw meaningful conclusions.
- Fit a regression model to data and use it for prediction.

Course Content

Unit 1: Introduction to Data Science and Data Visualization

Introduction, Motivating Examples and Scope, Getting and Analyzing Data: Reading Files, Scraping the Web, Need for Data Preprocessing and its Basics. Statistics: Introduction, Types of Statistics, Types of Data, Types of Experiments – Controlled and Observational study, Data Visualization and Interpretation: Histogram, Line plots, Bar Charts, Box Plots, Scatter Plots, Heat Maps, Good vs. Bad Visualization, Sampling: Sampling Methods, Sampling Errors, Case Study.

14 Hours

Unit 2: Probability Distributions and Principles of Point Estimation

Random Variables, Probability Distributions: Discrete Distributions (Binomial), Continuous Distributions (Normal), Chebyshev's inequality, Normal Probability Plots, Generation of Random Variates with applications, Sampling Distribution, The Central Limit Theorem and Applications, Continuity Correction-Normal approximation to Binomial, Principles of Point Estimation - Mean Squared Error, Maximum Likelihood Estimate Case Study

14 Hours

Unit 3: Confidence Intervals and Hypothesis Testing

Confidence Intervals: Interval Estimates for Mean of Large and Small Samples, Student's t Distribution, Interval Estimates for Proportion of Large and Small Samples, Confidence Intervals for the Difference between Two Means, Interval Estimates for Paired Data. Factors affecting Margin of Error, Hypothesis Testing for Population Mean and Population Proportion of Large and Small Samples, Drawing conclusions from the results of Hypothesis tests, Case Study.

14 Hours

Unit 4: Distribution Free Tests and Linear Regression

Distribution Free Tests, Chi-squared Test, Fixed Level Testing, Type I and Type II Errors, Power of a Test, Factors Affecting Power of a Test. Simple Linear Regression: Introduction, Correlation, the Least Square Lines, Predictions

using regression models - Uncertainties in Regression Coefficients, Checking Assumptions and transforming data, The Multiple Regression Model, Case Study.

14 Hours

Tools / Languages/Libraries: Jupyter Notebook**,** Python, Pandas, Matplotlib, Scipy, Seaborn, BeautifulSoup, Numpy, Scikit learn.

Text Book(s):

1: "Statistics for Engineers and Scientists", William Navidi, McGraw Hill Education, India, 4th Edition, 2015.

Reference Book(s):

1: "The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modelling", Raj Jain,Wiley, 2008.
2: "Sampling- Design and Analysis" ,Sharon L. Lohr, 2nd edition (stats), Cengage, 2010.
3: "Data Science from Scratch", Joel Grus, O'Reilly, 1st Edition, 2015.
4: "Statistics for Engineers and Scientists", William Navidi, 3rd Edition, McGraw Hill, 2010.

UE22CS242A: Web Technologies (4-0-0-4-4)

Web Technologies course demonstrates an in-depth understanding of the technologies necessary for designing and developing a rich web application in an efficient way.

Course Objectives:

- Basic web technologies and building blocks of a website using HTML, CSS, JavaScript and Advanced JavaScript
- The core concepts of HTML5, JQuery and AJAX, MERN (MongoDB, ExpressJS, ReactJS and NodeJS) stack and build an UI of the application using React JS
- Building a multi-tier application by interfacing UI to NodeJS
- Integrate database MongoDB through Express JS Framework and Web services.

Course Outcomes:

At the end of the course, the student will be able to,

- Understand basic web technologies like HTML, CSS and JavaScript
- Achieve rich user experience by implementing HTML5 features and Asynchronous communication using AJAX, JQuery and MERN stack layers (MongoDB, ExpressJS, ReactJS and NodeJS) and Create rich User Interface using React JS
- Understand and Integrate the UI with NodeJS
- Create RESTful Web services using ExpressJS and MongoDB database

Course Content:

Unit 1: HTML, CSS and Client Side Scripting

Introduction to Web Architecture and Web protocols (HTTP Request Response Formats, URLs), Basic Mark-ups & syntax, HTML elements & attributes, Web Form, HTML5 (New Tags, Inputs, Elements and Controls), CSS3.0 - Styles and Style sheets, Selectors, Style properties, Box Model, JavaScript Basics(variables, scope, Builtin Objects), JavaScript objects and Prototypal Inheritance, DOM Manipulations, Events and Event Handling in JavaScript

14 Hours

Unit 2: HTML5, JQuery and Ajax

HTML5 (APIs), JQuery Introduction, Callbacks and Promises, Introduction to Single Page Application, XML Vs JSON, Asynchronous Communication using AJAX and fetch API. **ReactJS** – MERN Introduction, React Classes and Components, JSX, Rendering of elements

14 Hours

Unit 3: ReactJS

Properties, State, Context, Component lifecycle methods, Refs & Keys, Event Handling, Stateless components. React Hook **NodeJS –** Understanding Node JS Architecture, Set up Node JS app, Node Modules, call-backs, buffers, streams, File system Module, HTTP Module, Handling HTTP Requests

14 Hours

Unit 4: MongoDB

MongoDB-Documents, Collections, Reading and Writing to MongoDB, MongoDB NodeJS Driver, Running a react application on NodeJS, React Router. **ExpressJS –** Introduction to Web services and REST API's , Express Framework Overview, Routing and URL building, Error Handling, Express Middleware, Form Data and File Upload.

14 Hours

Tools / Languages: HTML, CSS, JavaScript, MERN Technologies.

Text Book(s):

1: "Learning PHP, MySQL & JavaScript", Robin Nixon., 5th edition, O'Reilly Media, Inc. ISBN: 9781491978917, May 2018.

2: "Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React, and Node", Vasan Subramanian, Apress, March 2017.

Reference Book(s) :

1: “Beginning Node.js, Express & MongoDB Development”, Greg Lim, July 2019.
2:“Learning React, Functional Web Development with React and Redux”, Alex Banks and Eve Porcello, O’Reilly Media, May 2017.

Reference Link(s);

1: https://reactjs.org/docs/.
2:https://www.kirupa.com/react.

UE22CS243A: Automata Formal Languages and Logic (4-0-0-4-4)

The course introduces fundamental concepts in Automata and Formal Languages and their application to Logic. The course covers the notions of Finite State Automaton, Regular expression, Push down Automaton, Context Free Languages and Turing Machines. These abstract and formal models and their usage in Propositional and First Order Predicate Logic, allow for solving problems in Formal language Generation and Recognition.

Course Objectives:

- Teach students to construct basic machines like DFA, NFA which represent Regular Languages, Regular Expressions, and Regular Grammars and to identify Non – Regular Languages.
- To familiarize students to construct Teach students to identify Context Free Languages, to construct Push down Automata which represent Context Free Languages, to convert the given grammar to various normal forms and to make use of Membership Algorithm.
- Teach students to understand closure properties of Context Free Languages, to identify Non – Context Free Languages and to construct Turing Machines and
- To familiarize students with concepts like Recursively Enumerable languages, Recursive Languages, Undesirable Problems. And to familiarize notions of mathematical logic: logical notations (syntax) and how to assign meaning to them (semantics).

Course Outcomes:

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

- Design simple machines like DFA, NFA, convert NFA to DFA and minimize a given DFA, Construct regular expressions for different languages, verify that some languages are regular and some are not.
- Analyze the difference between Regular Languages and Context Free Languages, design Push Down automata, construct Context Free Grammars, and convert one form of the grammar to another form.
- Enumerate the properties of Context Free Grammars, verify that some languages are context free and some are not, design Turing Machines, and analyze the difference between acceptability and decidability and
- Analyze the difference between Recursive and Recursively Enumerable Languages, Decidable Languages, Turing – Recognizable and Co – Turing – Recognizable, some problems that cannot be solved by Turing Machines, reduce one Undesirables Problem to another, Undeniable Problems for Recursively Enumerable Languages. And make use of Propositional Logic and Predicate Logic in knowledge representation and truth verification.

Course Content:

Unit 1: Introduction

Mathematical Preliminaries and Notation, Three Basic Concepts. Finite Automata: Deterministic Finite Accepters, Non-Deterministic Finite Accepters, Equivalence of Deterministic and Non-Deterministic Finite Accepters, Reduction of the number of states in Finite Automata. Regular Expressions, Connection between Regular Expressions and Regular Languages Regular Grammars.

14 Hours

Unit 2: Regular Languages and Context Free Languages

Properties of Regular Languages: Closure Properties of Regular Languages, Elementary Questions about Regular Languages, Identifying Non Regular Languages. Definitions of PDA and CFL, Deterministic Pushdown Automata, Non-Deterministic Pushdown Automata, Pushdown Automata and Context Free Languages, Context Free Grammars.

14 Hours

.

Unit 3: Properties of Context Free Languages and Turing Machine

Parsing and Ambiguity. Simplification of Context–Free Grammars and Normal Forms: Methods for Transforming Grammars, Two Important Normal Forms, A Membership algorithm for Context Free Grammar. Properties of Context-Free Languages: Closure Properties and Questions about Context–Free Languages, Pumping Lemma for Context–Free Languages. Turing Machines: The Standard Turing Machine, Constructing Turing Machines, Combining Turing Machines for Complicated Tasks, Turing's Thesis.

14 Hours

Unit 4: Undecidability and design of a declarative language.

Hierarchy of Formal Languages and Automata: Recursive and Recursively Enumerable Languages, the Chomsky Hierarchy. Limits of Algorithmic Computation: Some Problems that cannot be solved by Turing Machines, Undecidable Problem for Recursively Enumerable Languages, idea of reduction.

A very simple Logic, Syntax, Semantics, A simple knowledge Base, A simple inference procedure. Propositional Theorem Proving: Inference and Proofs, Proof by Resolution, Conjunctive Normal Form, A resolution algorithm. Syntax and Semantics of First Order Logic: Models for First Order Logic Symbols and interpretations, Terms, Atomic Sentences, Complex Sentences Quantifiers, Equality, Numbers, sets and Lists. Example - The electronic circuits' domain.

14 Hours

Tools / Languages: JFLAP.

Text Book(s):

1: "An Introduction to Formal Languages and Automata", Peter Linz, Jones and Bartlett, New Delhi, India, 6th Edition, 2016.

2: "Artificial Intelligence – A Modern Approach", Stuart Russell and Peter Norvig, Pearson, 3rd Edition (Paperback), 2016.

Reference Book(s):

1: "Theory of Computation", Michael Sipser, Cengage Learning, New Delhi, India, 2008.

2: "Introduction to Automata Theory, Languages, and Computation", John E Hopcroft, Rajeev Motwani, Jeffrey D Ullman, Pearson Education, New Delhi, India, 3rd Edition, 2009.

3: "Theory of Computation: A Problem–Solving Approach", Kavi Mahesh, Wiley India, New Delhi, 2012.

UE22CS251B: Microprocessor and Computer Architecture (4-0-2-5-5)

This course will give you an in-depth understanding of the inner-workings of modern digital computer systems and trade-offs present at the hardware-software interface. The course focuses on key topics in microprocessor such as the system architecture, low level programming aspects and interface with other key components. Also the course will help in understanding the core computer architecture concepts such as multilevel in memory hierarchies, pipelining and super scalar techniques. A desirable knowledge of Digital Design and Computer organization is required.

Course Objectives:

- Introduce concepts of basic processor architecture and its design.
- Understanding the concept of concepts of pipeline architecture and hazards.
- Study of memory hierarchy, cache memory and its optimizations.
- Introduction to I/O Systems, interface and interaction with processor.
- Introduce advanced concepts in processor architecture like multi-core/ many core processor architectures.

Course Outcomes:

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

- Demonstrate ability to understand the design of different instruction sets like RISC/ CISC and their addressing modes.
- Demonstrate the ability to understand the design of a pipelined processor and its challenges.
- Demonstrate the use of tools to analyze the performance of programs on different architectures.
- Design alternative memory hierarchy layouts and optimizations.
- Demonstrate and appreciate modern trends in architecture such as multicore architectures.

Desirable Knowledge: UE22CS251A- Digital Design and Computer Organization.

Course Content:

Unit 1 : Architecture

Introduction, ISA Classification - RISC and CISC, Memory Addressing, Operands - Types and Size, Instruction Set - Operations, Control Flow, Instruction Encoding, Case Study - ARM/ MIPS/ x86 Processor.

14 Hours

Unit 2 : Pipeline & Basics of Caches

3- Stage Pipelining, 5 - Stage Pipelining, Pipeline Datapath and Control, Data Hazards – Forwarding vs. Stalling, Control Hazards, Branch Prediction Mechanisms and Exceptions, Performance Metrics. Basics of Caches - Fully Associative, Direct Mapped and Set Associative.

14 Hours

Unit 3 : Cache Optimization & I/O Architecture

Cache Performance, Basic Cache Optimization- Reduce in Miss Rate. Basic Cache Optimization- Reduce Miss Penalty, Reduce Hit Time.

I/O and bus architecture – accessing I/O devices, DMA Controller, Bus Architecture.

14 Hours

Unit 4 : Advances in Architecture

Introduction to Parallel Computing, PC – Applications, Memory architecture, Flynn's taxonomy, parallel programming models, Hardware Multi threading, Parallel examples: matrix multiplication, PC-Design Issues, Amdahl's Law, Gustafson Law, Multi-Core Architecture, Introduction to GPU computing.

14 Hours

Tools / Languages: ARM Simulator.

Text Book(s):

1: "Computer Organization and Design", Patterson, Hennessey, 5th Edition, Morgan Kaufmann, 2014.

2. "Computer Organization and Design – ARM Edition", Patterson, Hennessey, 4th Edition, Morgan Kaufmann, 2010.

3. "ARM System-on-Chip Architecture", Steve Furber, 2nd Edition, Pearson India, 2015.

Reference Book(s):

1: “Computer Architecture: A Quantitative Approach”, Hennessey, Patterson, 5th Edition, Morgan Kaufmann, 2011.
2: “The Definitive Guide to the ARM Cortex-M0 and Cortex MO+ processors”, Joseph Yiu, 2nd Edition, Newnes, 2015.

UE22CS252B: Computer Networks (4–0–2–5–5)

This is a foundation course on Computer Networking which focuses on building blocks of the Internet. We trace the journey of messages sent over the Internet from an application residing on one host machine (source) to another (Destination) using a top down layered approach. The course contents are organized based on TCP/IP Protocol stack.

Course Objectives:

- To present a broad overview of computer networking, the Internet and network layered architecture.
- To study the conceptual and implementation aspects of network applications and socket programming.
- To provide an insight of the Internet's connection–oriented and connectionless end–to–end transport service protocols and TCP's approach to congestion control, to learn exactly how the network layer can provide its host–to–host communication service.
- To study the IPv4 and IPv6 protocols, to explore several important link–layer concepts and technologies, LAN and Wireless LANs.

Course Outcomes:

At the end of this course, the student will be able to:

- Demonstrate in a concise way how the Internet is constructed and functions with respect to TCP/IP or OSI reference models.
- Explain basic concepts of application layer protocols like DNS, HTTP and implement simple client–server applications using socket programming.
- Understand the concept of reliable and unreliable data transfer protocols and how TCP and UDP implement these concepts.
- Demonstrate the ability to configure the routers and services such as DHCP, ICMP and NAT and implement logical addressing schemes.
- Construct and troubleshoot a wired or wireless LAN, and be able to understand wider networking issues.

Course Content:

Unit 1: Computer Networks and the Internet, Application Layer

Introduction to Computer Networks, Internet: A Nuts–and–Bolts Description, A Services Description, Protocol, The Network Edge: Access Networks, Physical Media, The Network Core, Packet Switching, Circuit Switching, A Network of Networks, Delay, Loss, and Throughput in Packet–Switched Networks, Overview of Delay in Packet–Switched Networks – Queuing Delay and Packet Loss, End–to–End Delay, Throughput in Computer Networks, The OSI Model and the TCP/IP Protocol Suite, Protocol Layers, The OSI Model, TCP/IP Protocol Suite.

Network Application Principles: Network Application Architectures, Processes Communication, Transport Services available to Applications, Transport Services provided by Internet; The Web, HTTP and HTTPS, Non persistent and persistent connection, HTTP Message Format, User Server Interaction: Cookies, Web Caching.

16 Hours

Unit 2: Application Layer, Transport Layer – UDP

DNS, The Internet's Directory Service: Services provided by DNS; How DNS works: DNS Records and messages; Peer to peer Applications; Socket Programming with TCP and UDP; Other Application Layer Protocols: FTP, SMTP, SNMP, Telnet, SSH.

Introduction to Transport Layer Services: Relationship Between Transport and Network Layer, Overview of the Transport layer in the Internet, Multiplexing and Demultiplexing; Connectionless Transport UDP: UDP Segment Structure, UDP Checksum.

12 Hours

Unit 3: Transport Layer – TCP, Network Layer and Internet Protocol

Principles of Reliable Data Transfer: Building a Reliable Data Transfer Protocol, Pipelined Reliable Data Transfer Protocol, Go–Back–N Protocol, Selective–Repeat; Connection Oriented Transport TCP: The TCP Connection, TCP Segment Structure, Flow Control, TCP Connection Management, and TCP Congestion Control. Numerical on TCP congestion control mechanisms–TCP Tahoe, Reno.

Overview of Network Layer: Forwarding and routing, what's Inside a Router? The Internet Protocol (IP) IPV4: Datagram Format, Fragmentation, Addressing, NAT.

14 Hours.

Unit 4: Network Layer and Internet Protocol, Link Layer and LAN

Introduction to Network layer Protocols: DHCP, ICMP; IPv6 Protocol: Packet Format, Transition from IPv4 to IPv6; Introduction to Routing Algorithms: Link State: Dijkstra's algorithm and Distance Vector: Bellman–Ford Algorithm.

Link layer – Error–Detection and Correction techniques, Parity checks, Internet Checksum, Cyclic Redundancy Check, and Multiple Access Protocols: CSMA/CD, CSMA/CA; Switched LAN: Link layer addressing and ARP, Ethernet: Link–layer switches. Retrospective: A Day in the Life of a Web Page Request. Physical Layer – Purpose, Signals to Packets, Transmission media. Wireless LANs: IEEE 802.11 LAN architecture, 802.11 MAC Protocol, IEEE 802.11 Frame.

14 Hours

Lab/ Hands-on: 14 Hours

1. Program on ping, tcpdump and wireshark.
2. Program on Exploring HTTP with wireshark, Web Server setup, FTP/ SMTP and SNMP Clients, Telnet, SSH and DNS
3. Program on Wireshark based TCP congestion window plotting, UDP traffic analysis.
4. Program on Cisco Packet Tracer based Router experiments; IPv4 Fragmentation based wireshark experiments, Inspection of DHCP, ICMP.
5. Program on Wireshark based Link Layer protocol inspection.

Tools/ Languages: Wireshark, Python.

Text Book(s):

1: "Computer Networking: A Top – Down Approach", James F. Kurose, Keith W. Ross, 7th Edition, Pearson Publication, 2017.

Reference Book(s):

1: "TCP IP Protocol Suite", Behrouz Forouzan, 4th Edition, McGraw–Hill, 2010.

UE22CS241B: Design and Analysis of Algorithms (4-0-0-4-4)

Algorithms play a key role in science and practice of computing. Learning algorithm design technique is a valuable endeavour from practical standpoint and algorithm design techniques have considerable utility as general problem solving strategies, applicable to problems beyond computing. This course includes classic problems of computer science, application of design techniques and analysis in terms of time and space.

Course Objectives:

- Learn to design and analyze algorithms with an emphasis on the resource utilization in terms of time and space.
- Learn various techniques in the development of algorithms so that the effect of problem size and architecture
- Design on the efficiency of the algorithm is appreciated.
- Learn to apply appropriate algorithmic design techniques for specific problems.
- Learn to trade space for time in algorithmic design using input enhancement and per-structuring.
- Learn to improve the limitations of algorithmic power.

Course Outcomes:

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

- Identify the design technique used in an algorithm.
- Analyse algorithms using quantitative evaluation.
- Design and implement efficient algorithms for practical and unseen problems.
- Analyse time efficiency over trading space.
- Understand the limits of algorithms and the ways to cope with the limitations.

Course Content:

Unit 1: Introduction and Brute Force

Algorithms, Fundamentals of Algorithmic Problem Solving, Important Problem Types. Analysis of Algorithm Efficiency: Analysis Framework, Asymptotic Notations and Basic Efficiency Classes, Mathematical Analysis of Non Recursive and Recursive Algorithms. Brute Force: Selection Sort, Bubble Sort, Sequential Search, Brute Force String Matching, Exhaustive Search.

14 Hours

Unit 2: Decrease – and – Conquer & Divide-and-Conquer

Decrease-and-Conquer: Insertion Sort, Topological Sorting, Algorithms for Generating Combinatorial Objects, Decrease-by-a-Constant-Factor Algorithms. Divide-and-Conquer: Master Theorem, Merge Sort, Quick Sort, Binary Search, Binary Tree Traversals, Complexity analysis for finding the height of BST, Multiplication of Large Integers, Strassen's Matrix Multiplication.

14 Hours

Unit 3: Transform-and-Conquer Space and Time Tradeoffs & Greedy Technique

Transform and- Conquer**:** Pre-sorting, Heap Sort, Red-Black Trees, 2-3 Trees and Analysis of B Trees. Problems on - Decrease by a constant factor /constant number. Space and Time Tradeoffs: Sorting by Counting, Input Enhancement in String Matching - Horspool's and Boyer-Moore Algorithms. Greedy Technique: Prim's Algorithm, Kruskal's Algorithm and union-find algorithm, Dijkstra's Algorithm, Huffman Trees

16 Hours

Unit 4: Limitations, Coping with the Limitations of Algorithm Power & Dynamic Programming,

Limitations of Algorithm Power**:** Lower-Bound Arguments, Decision Trees, P, NP, and NP-Complete, NP-Hard Problems. Coping with the Limitations of Algorithm Power: Backtracking, Branch-and-Bound. Dynamic Programming: Computing a Binomial Coefficient, The Knapsack Problem and Memory Functions, Warshall's and Floyd's Algorithms.

12 Hours

Tools / Languages: C Programming Language, GCC Compiler.

Text Book(s):

1: “Introduction to the Design and Analysis of Algorithms”, Anany Levitin, Pearson Education, Delhi (Indian Version), 3rd Edition, 2012.

Reference Book(s):

1: “Introduction to Algorithms”, Thomas H Cormen, Charles E Leiserson, Ronald L Rivest and Clifford Stein, Prentice-
Hall India, 3rd Edition, 2009.
2: “Fundamentals of Computer Algorithms”, Horowitz, Sahni, Rajasekaran, Universities Press, 2nd Edition, 2007.
3: “Algorithm Design”, Jon Kleinberg, Eva Tardos, Pearson Education, 1st Edition, 2006.

UE22CS242B: Operating Systems (4-0-0-4-4)

This course focuses on fundamental operating systems concepts including various algorithms and trade-offs for efficient management of resources such as CPU, Memory, Storage and I/ O. This course requires the student to have a desirable knowledge of Data Structures and its Applications.

Course Objectives:

- Focus on fundamental Operating System concepts.
- Provide an understanding of various components of the Operating System (OS).
- Delve deeper into various algorithms and associated trade-offs for efficient resource management such as process, disk, and memory management.
- Introduce design principles and trade-offs in the design of Operating Systems.

Course Outcomes:

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

- Understand the principles and modules of Operating Systems.
- Understand the design of various algorithms for scheduling and their relative performance.
- Understand the concept of Deadlocks that typically occur in OS. Deadlocks - Avoidance and Detection.
- Implement Operating Systems Concepts related to process management, Concurrent processes, Threads and Memory Management.

Desirable Knowledge: UE22CS252A - Data Structures and its Applications.

Course Content:

Unit 1 : Introduction and Process Management

What Operating Systems Do, Operating-System Structure & Operations, Kernel Data Structures, Operating-System Services, Operating System Design and Implementation.

Shell programming: Overview of bash shell programming – variables, control flow

Processes: process concept, Process Scheduling, Operations on Processes, System calls for process management- fork (), vfork (), wait () and exec ().

CPU Scheduling: Basic Concepts, Scheduling Criteria, Scheduling Algorithms. Case Study: Linux Scheduling Policies.

Shell programming – cron

14 Hours

Unit 2 : IPC, Threads and Concurrency –

IPC – Introduction, Shared Memory systems, Message Passing, Communication in Client–Server Systems- Pipes, ordinary pipes and named pipes, system calls for shared memory, pipes and fifo's.

Threads: Overview, Multicore Programming, Multithreading Models, Thread Libraries, Thread Scheduling.

Process Synchronization: Background, 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, Synchronization Examples- Synchronization in Linux. System calls for threads creation and synchronization-POSIX Threads.

Deadlocks: System Model, Deadlock Characterization, Deadlock avoidance, Banker's Algorithm, Deadlock Detection.

14 Hours

Unit 3: Memory Management

Main Memory: Background- Basic Hardware, Address Binding, Logical Versus Physical Address Space, Dynamic Loading, Dynamic Linking and Shared Libraries, Swapping, Contiguous Memory Allocation, Segmentation, Paging, Structure of the Page Table.

Virtual Memory: Background, Demand Paging, Copy-on-Write, Page Replacement Algorithms-FIFO, LRU, Optimal, Allocation of Frames, Thrashing.

14 Hours

Unit 4 : File and Storage Management

File-System Interface: File Concept, system calls for file operations-open(), read(),write(), lseek(), close() and system call to retrieve file attributes and file types-stat(), lstat(), Access Methods, Directory and Disk Structure, system calls for reading directories, system calls to create hard links (link()) and symbolic links-symlink().
File-System Implementation: File-System Structure, File-System Implementation, Directory Implementation, Allocation Methods, File Sharing, Protection.
Storage management: Overview of Mass-Storage Structure, Disk Scheduling, Swap-Space Management, RAID Structure.
System Protection: Goals, Principles and Domain of Protection, Access Matrix, Implementation of the Access Matrix, Access Control
Shell programming - awk, sed

14 Hours

Tools/Languages/OS : C, Linux/Unix OS for system call implementation.

Text Book(s):
1: "Operating System Concepts", Abraham Silberschatz, Peter Baer Galvin, Greg Gagne 9th Edition, John Wiley & Sons, India Edition ,2016.
2: "Advanced Programming in the Unix Environment", Richard Stevens and Stephen A Rago, Pearson, 3rd edition, 2017.

Reference Book(s):
1: "Operating Systems, Internals and Design Principles", William Stallings, 9th Edition, Pearson, 2018.
2: "Modern Operating Systems", Andrew S Tanenbaum, 3rd edition, Pearson, 2007.
3: "Learning the bash shell", Cameron Newham, 3rd edition, O'Reilly, 2005.

UE21CS351A: Database Management System (5-0-2-7-5)

This course offers a solid theoretical foundation in Database Management System (DBMS) and explores the practical applications of DBMS in a real-world scenario. The course emphasizes on the creation and the design of relational database systems. It also introduces the databases that provides flexible schemas and scale easily with large amounts of data and high user loads.

Course Objectives:

- Understand fundamental concepts, terminology, and application of relational databases and Construct ER diagrams for a desired application and transform the same into a relational schema.
- Understand database design concepts and design relational databases and construct basic and advanced SQL queries.
- Understand, and apply Normal Forms in database design, Transactions, Concurrency control, Locking, and Recovery, implement them in a real-time setup, and transform the given Normal Form into the desired form.
- Understand the concepts of Database Security, and NoSQL databases such as MongoDB, Neo4j, and DynamoDB

Course Outcomes:

At the end of this course, the student will be able to:

- Demonstrate an ability to explain the basic concepts of database management, construct and transform ER diagrams into Relational Schema
- Design databases and construct simple and advanced SQL queries for given contexts.
- Explain Normal Forms, employ them in Database Design, and apply database security concepts in application contexts
- Demonstrate the ability to use semi structured and NoSQL databases

Course Content:

Unit 1: Introduction to Database Management

Database System Applications, Purpose, View of data, Database Languages, Database design- Introduction to databases, Database application architecture, Users and Administrators, E-R Model, reducing ER to a relational schema. Structure of relational databases, Database schema, and its constraints, Keys

14 Hours

Unit 2: Relational Model and Database Design

Relational operations (Algebra), Unary Operations - Unity, Binary, Aggregate Functions, Grouping, SQL overview, Data definition, Structure of SQL queries, Additional Basic Operations, Set Operations, Null Values, Aggregate Functions, Nested Subqueries, Database Modification, Join expressions, Views, Triggers, Functions, and Procedures

14 Hours

Unit 3: Advanced Design Concepts and Implementation

Functional Dependencies, Inference Rules, Closure, Equivalence, Minimal Cover Normal Forms Based on Primary Keys (1NF, 2NF, and 3NF), General Definitions of Second and Third Normal Forms Boyce-Codd Normal Form, Properties of Relational Decompositions, Overview of Higher Normal Forms. Database transactions, Concurrency control, Locking, Recovery, Database Security

14 Hours

Unit 4: Advanced Databases

Query Processing and Optimization, Accessing SQL from a Programming Language, Structured, Semi structured, Unstructured data, Introduction to NoSQL databases, CAP theorem, Document database (MongoDB), Key-Value database (DynamoDB), Graph databases (Neo4j) **14 Hours**

Lab/Hands-on sessions

1: Draw an ER diagram for a given problem statement
2: Conversion of an ER diagram into Relational schema
3: DDL – create, constraints, alter, rename, drop, truncate table, Views.
4: DML – Insert, Update, Delete, Transactions - commit, rollback, savepoint
5: SQL - Set operators: union, intersect, minus.
6: SQL – Aggregate functions.
7: SQL – Joins: inner, outer; Sub queries: correlated and uncorrelated

8: SQL – Creating Functions and Procedures
9: SQL – Creating Triggers and Cursors
10: XML- Database access
11: NoSQL database queries
12: High-level programming language accessing a database using an API.

Tools/ Languages: MySQL Workbench, Python, ERwin, Any other tool for ER modeling

Text Book(s):
1: "Database System Concepts", Silberschatz, H Korth and S Sudarshan, McGrawHill, 7th Edition, 2019.
2: "Fundamentals of Database Systems", Ramez Elamsri, Shamkant B Navathe, Pearson, 7th Edition, 2017.

Reference Book(s):
1: Database Management Systems, R Ramakrishnan, J Gehrke, 3rd Edition, McGraw Hill, 2002
2: Data on the Web: From Relations to Semistructured Data and XML, S Abiteboul, P Buneman, D Suciu, Morgan Kauffman, 1999

UE21CS352A: Machine Intelligence (4-0-2-5-5)

Machine Intelligence (MI) surrounds us today: in phones that respond to voice commands, programs that beat humans at Chess and Go, robots that assist surgeries, vehicles that drive in urban traffic, and systems that recommend products to customers on e-commerce platforms. This course aims to familiarise students with the breadth of modern MI, to impart an understanding of the dramatic surge of MI in the last decade, and to foster an appreciation for the distinctive role that MI can play in shaping the future of our society. This course requires the student to have a desirable knowledge of Statistics for Data Science, Linear Algebra and its Applications and Design and Analysis of Algorithms.

Course Objectives:

- Familiarize the concepts of Intelligent Agents and Search Methods.
- Formulate a well - defined Machine Learning problem with clear metrics.
- Understand the notions of Hypotheses Space, Hypotheses Structure and Search.
- Become conversant with types of Machine Learning Algorithms, their applicability and Inductive Bias.
- Familiarize with techniques for Ensemble Learning, Nature based Optimization.

Course Outcomes:

At the end of this course, the student will be able to:

- Apply Intelligent Search methods for a variety of problems.
- Distinguish categories of Data Attributes, Dimensions, and Sample Sizes.
- Acquire a thorough understanding of Supervised, Unsupervised Learning,
- Ensemble Methods and Nature based Optimization.
- Apply deep learning methods.

Desirable Knowledge: UE21CS241A-Statistics for Data Science, UE21MA241B-Linear Algebra & its Applications UE21CS241B- Design and Analysis of Algorithms.

Course Content:

Unit 1: Introduction, Performance Metrics, Classification with Decision trees and KNN

Introduction to AI and ML, Intelligent Agents and its Types, Machine Learning and its Models, Concepts of hypotheses, Version space, inductive bias, Performance metrics-accuracy, precision, recall, sensitivity, specificity, AUC, ROC, Bias Variance decomposition. Decision Trees- Basic algorithm (ID3) - for classification, Decision boundary for decision trees(x-y axis), Hypothesis search and Inductive bias, Issues in Decision Tree Learning – Over fitting, Solutions to over fitting, dealing with continuous values. Linear Regression with GD, Logistic Regression, Instance-based learning: k-nearest neighbour learning (Classification & Regression), Decision boundary for KNN,

14 Hours

Unit 2 : Supervised Learning with ANN, Introduction to Deep Learning techniques:

Artificial Neural networks: Introduction, Perceptions, Multi-layer networks and back-propagation, Activation functions (Step, Sigmoid, Tanh, ReLU) Various Optimizers(GD, SGD, Momentum-based, Adagrad, Adam,) Introduction to Deep Learning, Introduction to Convolution operation and Convolution Neural Network. (Parameter calculation, Max pooling, Avg pooling). Introduction to Recurrent Neural Network, Vanishing and exploding gradient, Variants of RNN : LSTM, GRU(mention of gates), Introduction to Large Language Models(LLM).

14 Hours

Unit 3 : SVM, Boosting and Stochastic Models:

Support Vector Machines – margin and maximization, SVM - The primal problem, the Lagrangian dual, SVM – Solution to the Lagrangian dual,(Hard Margin and Soft Margin, Classification ONLY), Kernel functions: Linear, polynomial (Derivation only for linear function).

Combining weak learners, Improving performance with Gradient Boost, Random Forest, Bayesian Learning – Bayes theorem, Concept learning, Maximum likelihood, Bayes optimal classifier, Naïve Bayes classifier, Expectation maximization and Gaussian Mixture Models,.

14 Hours

Unit 4 : HMM, Unsupervised Learning ,Dimensionality Reduction and Genetic Algorithms, PSO:
Hidden Markov Models, Hierarchical vs. non-hierarchical clustering, Agglomerative and divisive clustering, K-means clustering, Bisecting k-means, K-Means as special case of Expectation Maximization, Dimensionality reduction techniques – PCA, SVD applications. Genetic Algorithms – Representing hypothesis, Genetic operators and Fitness function and selection - Application in Decision Trees, Weight Determination, clustering. Introduction to PSO and application in Single Objective optimization problems.

14 Hours

Lab / Hands-on : 14 Hours

Applications using

1. Decision Trees.
2. ANN (Basics).
3. CNN model.
4. SVM (SVC).
5. Naive Bayes
6. GMM
7. HMM

Tools / Languages : Pytorch

Text Books(s):

1: “Machine Learning”, Tom Mitchell, McGraw Hill Education (India), 2013.
2: “Neural Networks in Deep Learning” Charu agarwal, Springer International Publishing AG, part of Springer Nature 2018.
3: “Pattern Recognition and Machine Learning”, Christopher Bishop, Springer (2nd Printing), 2011.

Reference Book(s):

1: “Machine Learning: The Art and Science of Algorithms that Make Sense of Data”, Peter Flach, Cambridge University Press (2012).
2. “Hands-on Machine Learning with Scikit-Learn and TensorFlow”, Aurelian Geron, O’REILLY, 1st Edition, 2017.
3: “Artificial Intelligence: A Modern Approach (3rd Edition)”, Stuart Russel and Peter Norvig, Pearson , 2009.

UE21CS341A: Software Engineering (4-0-0-4-4)

Software Engineering course deals with the Software development life cycles, their individual phases, principles, methods, procedures and tools associated. This also deals with making of choices, implications of making choices and exposes students to the Software eco-system which will be experienced by students in a post college environment.

Course Objectives:

- Ensure the relevance and need of an engineering approach to software development.
- Learn Software Engineering concepts.
- Expose students to the tools available as part of the Software Development and the Life Cycle.
- Enable the students to practice the principles of Software Product Development.
- Enable students to understand the continuous development, build, test and release of software products.

Course Outcomes:

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

- Relate to the challenges of Software Development and relate to Software Engineering as a methodical approach for development.
- Use Software Development Life Cycles with an understanding of when and where to use.
- Work in different lifecycle phases and produce artifacts expected at each phase, and evaluate using quality metrics.
- Work on a project plan, track and manage projects.
- Understand the connectivity of the development process to operations, and relate to the DevOps activities.

Course Content:

Unit 1: Introduction to Software Engineering and Requirements Engineering

Introduction, Context and Drivers of Software Engineering, Processes Phases and Development lifecycle, Product Lifecycle, Legacy SDLCs -Waterfall Model, V Model, Incremental Model, Evolutionary Model; Agile approach of software development, Contrasting Agile and Plan driven approaches, Agile SCRUM model and exposure to other Agile approaches like Lean Agile, Reuse focussed Software Development approaches- CBSE, Product Line. **Requirements Engineering:** Requirement's introduction and properties, Feasibility, Requirements Elicitation, Analysis and modelling, Specification and Verification, Requirement Management and Requirements Traceability.

14 Hours

Unit 2: Software Project Management and Software Architecture and Design

Software Project Management Fundamentals, Software Project Management Lifecycle, Planning activities - Choice of Lifecycles, Project Organization, WBS, Software Estimation, Scheduling, Risk Management, Quality Management, Project Plan illustration using Microsoft Project or similar tool, Monitoring of Execution and Control, Project Closure. **Software Architecture and Design:** Software Architecture, Characteristics and factors, Architectural approach including decomposition, Design and enabling techniques, key issues to be addressed as part of design, Architectural choices, impacts and conflicts, Generic Design approach, Architectural View, Styles, Architectural and Design Patterns. Contrast procedural and Object Orientation in Architecture and Design, Service Oriented Architecture (SOA).

14 Hours

Unit 3: Implementation, SCM and Software Quality & Relationship to Testing

Introduction to software construction, Coding principles, standards and guidelines, Factors for effective coding - Defensive, Secure and Testable programming, Code Review / Peer Review, Managing Construction and exposure to tools.

Software Configuration Management (SCM): Elements of a Configuration Management System, Configuration Items, Baselines, Repositories, Branch Management, Build Management, Install, Change and Release Management, Patching and Patch Management, Configuration Management Plan – Relate to tools like GitHub. **Introduction to Software Quality:** Software Quality, A Taxonomy of Quality Attributes, Perspectives on Quality, Cost of Quality, Metrics, Software Quality Assurance, SEI CMM, Testing and Quality-

14 Hours

Unit 4: Software Testing, Maintenance and Software Ethics

Fundamentals of testing, Test Objectives, Verification and Validation, Terminologies, Testing Types – Black box, White box, Gray box, Static and Dynamic testing, Technique based testing – Coverage based, Fault based, Levels of testing – Unit testing, Integration testing, System testing and Acceptance testing, Manual and Automated testing Test Planning: test adequacy criteria, test strategy, test models, test schedule, test resource management, milestones, risks and measure and test plan. Test roles and responsibilities, test process/lifecycle Test case generation, test execution and debugging. Test metrics. Exposure to test tools like Junit, Selenium. Software Maintenance, Lifecycle, activities and techniques. SE in Global Environment, ITSM, ITIL and DevOps, CI/CD pipeline

14 Hours

Tools / Languages: GitHub, MS Project/GanttPro/Jira, Jenkins.

Text Book(s):

1: "Software Engineering: A Practitioner's Approach", Roger S Pressman, McGraw Hill, 6th Edition, 2010
2: "Software Testing – Principles and Practices", Srinivasan Desikan and Gopalaswamy Ramesh, Pearson, 2006.

Reference Book(s):

1: "Effective DevOps: Building a Culture of Collaboration, Affinity, and Tooling" by By Jennifer Davis, Ryn Daniels, O' Reilly Publications, 2018
2. "Software Engineering: Principles and Practice", Hans van Vliet, Wiley India, 3rd Edition, 2010.
3: "Software Engineering", International Computer Science Series, Ian Somerville, Pearson Education, 9th Edition, 2009.
4: "Foundations of Software Testing ", Aditya Mathur, Pearson, 2008.
5: "Software Testing, A Craftsman's Approach ", Paul C. Jorgensen, Auerbach, 2008.
6:IEEE SWEBOK, PMBOK, BABOK and Other Sources from Internet

UE21CS342AA1: Advanced Algorithms (4-0-0-4-4)

Algorithm Design and Analysis is fundamental and important part of computer science. The course on Advanced Algorithms introduces the learner to advanced techniques for design and analysis of algorithms and explores a variety of applications. This course requires the student to have a desirable knowledge of Design and Analysis of Algorithm.

Course Objectives:

- Enable the learner with basics of Amortized Complexity Analysis of Data Structures.
- Empower the learner with String Matching/ Prediction Algorithms.
- Hone the problem solving skills of the learner by introducing them to Flow Networks, Bipartite Matching and DFT.
- Introduce the learner to Number Theoretic Algorithms.
- Enable the learner with design strategy of Dynamic Programming, Randomized Algorithms and Approximation Algorithms.

Course Outcomes:

At the end of this course, the student will be able to:

- Perform Amortized Analysis of complex Data Structures.
- Apply String Matching Algorithms to solve string related problems.
- Implement Max Flow and FFT Algorithms.
- Apply Number Theoretic concepts in designing Cryptographic Algorithms.
- Solve complex problems using Dynamic Programming, Randomized Algorithms and Approximate Algorithms.

Desirable Knowledge: UE21CS241B – Design and Analysis of Algorithms.

Course Content:

Unit 1: Basics of Complexity and String Matching Algorithms

Asymptotic Notations, Standard notations and common functions, Recurrences and Solution of Recurrence equations- The Substitution method, the Recurrence tree method, the Master method, Amortized Complexity Analysis: Aggregate, Accounting and Potential Methods.NP-Completeness, NP Reduction.

Naive String Match, Boyer–Moore, Rabin–Karp, String matching with Finite State Automata, and Knuth–Morris–Pratt Algorithms

16 Hours

Unit 2: Suffix Trees and Maximum Flow,

Suffix Trees - Applications of Suffix Trees, Regular Expression Searches Using Suffix Trees.

Flow Networks, The Ford-Fulkerson method, The Edmonds-Karp algorithm, Maximum Bi-Partite Matching,

12 Hours

Unit 3: Polynomials and FFT & Number-Theoretic Algorithms

Polynomials and FFT: Representation of Polynomials, Efficient Polynomial Multiplication, DFT and FFT, Efficient Implementation of FFT.

Number-Theoretic Algorithms**:** Elementary notions; GCD, Modular Arithmetic, Solving modular linear equations, Modular Inverse, The Chinese remainder theorem, Powers of an element; RSA cryptosystem; Primality testing; Integer factorization.

16 Hours

Unit 4: Dynamic Programming Randomized Algorithms and Approximation Algorithms

Elements of Dynamic Programming, Problems - Coin- Row, Rod-Cutting, Matrix-Chain Multiplication, Longest Common Subsequence. Randomized Algorithms: Hiring Problem, Indicator random variables, Approximation Algorithm: Vertex Cover Problem, TSP, The Subset Sum Problem, Linear Programming.

12 Hours

Tools / Languages: C or C++ Programming Language

Text Book(s):

1: “Introduction to Algorithms”, T H Cormen, C E Leiserson, R L Rivest and C Stein, PHI, 3rd Edition, 2010.

Reference Book(s):

1:“The Algorithm Manual”, Steven Skiena, Springer, ISBN: 9788184898651, 2nd Edition, Springer, 2008.
2: “Randomized Algorithms”, R Motwani and P Raghavan, Cambridge University Press, 2011.

UE21CS342AA2: Data Analytics (4-0-0-4-4)

The course explores the data analytics lifecycle: question formulation, data collection and cleaning, exploratory, analysis, visualization, statistical inference, prediction, and decision-making. Focuses on building analytical models using key principles and techniques. This course requires the student to have a desirable knowledge of Statistics for Data Science.

Course Objectives:
The objective(s) of this course is to,

- Assess a dataset for the type of data, inter-relationships between features and interpret the data meaningfully.
- Understand the assumptions that underlie regression models and determine the appropriate model and parameters for a problem.
- Recognize patterns such as trend and seasonality in a time series data and design appropriate forecasting models.
- Understand the nuances of recommender systems, build and validate appropriate models for recommendations.
- Understand the role of data analytics in business decision making.

Course Outcomes:
At the end of the course, the student will be able to:

- Perform exploratory data analysis on a given set of data and identify/ interpret correlation between features.
- Narrow down a subset of regression techniques to model data and evaluate the efficacy of these models.
- Analyze a time series and build appropriate time series models for prediction and evaluate them.
- Analyze data to infer underlying patterns, select the appropriate techniques and formulate recommendations.
- Develop intelligent decision support systems using techniques such as stochastic models, A/B testing.

Desirable Knowledge : UE21CS241A – Statistics for Data Science

Course Content:
Unit 1: Exploratory data analysis, ANOVA and correlation analysis
Introduction to business analytics, review of descriptive analysis and dimensionality reduction, ANOVA, correlation analysis, hidden variables

13 hours

Unit 2: Regression Analysis
Simple linear regression, multiple linear regression and variations (such as ridge regression and lasso regression), an introduction to multivariate and nonlinear regression, logistic regression

15 Hours.

Unit 3: Time Series Analysis and Markov Chains
Time series data components, moving average and exponential smoothing methods, regression models for forecasting, AR model identification using ACF/ PACF, concept of stationarity – its importance, testing for stationarity and converting a non-stationary signal to stationary, ARMA and ARIMA modelling, Discrete Markov chains, Markov chains with Absorbing States, expected duration to reach a state from other states, computing the retention probability and customer lifetime value, confounding variables.

13 Hours

Unit 4: Recommender Systems
Collaborative filtering, knowledge-based filtering: constraint-based and case-based (with a mention of knn, decision trees, agglomerative clustering, and DBSCAN), a brief introduction to text classification/ clustering for content-based filtering, ensemble methods (bagging and boosting) to improve accuracies, interpreting ML models, mining of association rules and evaluation of recommender systems, A/B testing.

15 Hours

Tools & Languages: R and Python Programming Languages.

Text Book(s):

1. “Business Analytics, The Science of Data-Driven Decision Making”, U. Dinesh Kumar, Wiley 2022, second edition.
2. “Recommender Systems: The Textbook”, Charu C. Agarwal, Springer 2016.

Reference Book(s):

1. Data Mining: Concepts and Techniques by Jiawei Han, Micheline Kamber and Jianei, The Morgan Kaufmann Series in Data Management Systems, Elsever publications, 3rd Edition, 2012.
2. The Elements of Statistical Learning, Trevor Friedman, Robert Tibshirani and Jerome Hastie, Data Mining, Inference and Prediction, Springer 2001.
3. Practical Data Science with R, Nina Zumel and John Mount, Manning Publications, 2014.

UE21CS342AA3 : Internet of Things (4-0-0-0-4)

The Internet of Things is already changing the way people live and interact with humans and machines. Businesses in every vertical have already started to leverage the power of IoT platforms to increase their efficiency and performance. This course introduces the students to the three layer architecture of the IoT exploring all the major connectivity options and application layer protocols. It also encourages the students to develop interesting applications using different development boards, sensors and actuators. The course introduces the students to cloud platforms and analytics including security aspects of IoT Application development.

Course Objectives:

- Learn the fundamentals of the Internet of Things.
- Learn the concepts of smart objects & IoT architecture.
- Compare different application and network layer protocols for Internet of Things.
- Appreciate the role and importance of Data Analytics and Security in IoT.
- Apply IoT for real word problems and Know the role of IoT in various verticals.

Course Outcomes:

At the end of the course the student will be able to

- Recognize the principles of smart objects and the potential for IoT and comprehend IoT architecture from sensors to the cloud, including edge gateways.
- Determine the static and dynamic performance metrics for core sensors and to understand the architecture of smart sensors for IoT system design.
- Learn how to use various wired and wireless technologies to provide connectivity for IoT applications and select a suitable application-layer protocol among the ones used for IoT applications.
- Recognize how analytics and artificial intelligence algorithms are used in the IoT ecosystem.
- Recognize IoT security and privacy concerns.

Desirable Knowledge: UE21CS151A-Python for Computational Problem Solving, UE21CS151B-Problem Solvig with C, UE21EC141A - EPD, UE21CS252B- Computer Networks.

Course Content:

Unit I : Introduction to IoT & Architecture:Introduction

IoT Traffic Model, IoT Connectivity, IoT Verticals, Use Cases & Applications, IoT Value Chain. Examples of IoT Use Cases & Applications, IoT Project Implementation, IoT Standards. **IoT Architecture:**Introduction, Factors Affecting an IoT Architectural Model, IoT Architectural Model, IoTWF Model, Data Center & Cloud, Computing (Cloud, Fog & Edge).

14 Hours

Unit 2: IoT Sensors

Introduction, Sensors & Its Performance Metrics, Smart Sensors, MEMS, Sensor Fusion, Self Calibration, Sensors of the Future.

14 Hours

Unit 3 : IoT-Protocols IoT Wired Connectivity

Introduction, Ethernet, Ethernet TSN, Power Line Communications. **Unlicensed-band Wireless Connectivity:** Introduction, Zigbee, BLE, WiFi, LoRaWAN. **Cellular IoT Technologies:** Introduction, GSM-IoT, LTE, Practical Use Cases**. IoT Data Communication Protocols:** Introduction, HTTP, MQTT, CAOP

14 Hours

Unit 4 : IoT – Analytics, Security & Privacy

Introduction, Data Pipeline, AI, Machine Learning, Supervised Learning Technique: Classification: Decision Tree Example. **IoT Cloud-Based Services & Platforms:** Introduction to AWS, Azure, ThingSpeak for IoT. **IoT Security -**

Introduction, IoT Threats, IoT Vulnerabilities, IoT Threat Modeling & Risk, IoT Security Regulations, IoT Privacy Concerns & Regulations, IoT Security & Privacy Examples, IoT & Blockchain.

14 Hours

Tool & Languages: Python, Embedded-C, Cloud Platforms, Single Board Computers

Textbook (s):

1: “Fundamentals of Internet of Things” Dian, F. J, , Wiley Professional Development (P&T), 2022.

Reference Book(s):

1: "IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things” David Hanes, Gonzalo Salgueiro, Patrick Grossetete, Robert Barton, Jerome Henry,, 1st Edition, Pearson Education (ISBN: 978-9386873743).
2: “Building Enterprise IoT Applications”, Chandrasekar Vuppalapati, CRC Press, Taylor & Francis Group, 20203: “Internet of Things – A hands-on approach”, Universities Press, 2015

UE21CS342AA4: Applied Cryptography (4-0-0-4-4)

Cryptography is the science of securing data by using mathematical concepts. This course will present the fundamentals of cryptography, as well as its applications and issues of how cryptography is used in practice. Students will have opportunities to dwell well into problem solving and hands-on sessions.

Course Objectives:

- To enable to learn the fundamental concepts of cryptography and utilize these techniques in computing systems.
- To discuss about various symmetric encryption techniques.
- To understand the concept of public key cryptography.
- To introduce message authentication and hash function.
- To provide an overview of authentication techniques using cryptography.

Course Outcomes:

At the end of this course, the student will be able to:

- Evaluate classical ciphers and different cryptographic primitives.
- Explain the notions of symmetric encryption and sketch their formal security definitions.
- Describe and implement specifics of prominent public-key crypto systems.
- Evaluate the authentication and hash algorithms.
- Discuss authentication applications.

Course Contents:

Unit 1: Classical Ciphers

Introduction to Cryptography, Cryptanalysis and brute-force attack, Basic cryptographic primitives, Classical ciphers: Substitution - Caesar, Playfair and Hill cipher, Transposition cipher - Rail fence, Columnar and Double columnar, Cryptanalysis of classical ciphers, Mathematical background for cryptography: Modulo arithmetic, GCD, Euclidean algorithm, Introduction to probability, Conditional probability, Law of total probability, Shannon's theorem and Perfect secrecy, One-time pad encryption and its limitations.

14 Hours

Unit 2: Symmetric Key Cryptography

Introduction to symmetric key cryptography, Pseudo Random Number Generator (PRNG), The Feistel cipher, Data Encryption Standard (DES), Analysis, Multiple DES, Security and the avalanche effect, Mathematics- I: Galois fields, Polynomials- Advanced Encryption Standard (AES), Transformation functions, Key expansion, Analysis of AES, Block and Stream ciphers, Block cipher modes of operation, Drawback of symmetric key cryptosystem, Need for asymmetric key cryptography.

14 Hours

Unit 3: Public Key Cryptography and Hashing techniques

Introduction, Mathematics- II: Prime number, Primitive root, Prime factorization, The RSA algorithm, Diffie-Hellman key exchange, Elgamal cryptographic system, Elliptic curve cryptosystem, Digital Certificates, Cryptographic hash functions: Introduction, Applications, Collision resistance, Secure Hash Algorithm (SHA), Message Digest algorithm (MD5)**.**

14 Hours

Unit 4: Hashing Techniques and key management

Key-Distribution Centre (KDC), Kerberos, Public Key Infrastructure (PKI) and standard (PKCS), Birthday attack, Entity authentication methods: Password, Challenge-Response, Zero knowledge protocols, Biometrics. One-way authentication, Mutual authentication, Centralized authentication, Authentication using cryptography use cases: Electronic money, factory setup, web application, WhatsApp, Use Case of Asymmetric and Symmetric Encryption: HTTPS, Introduction to Post-Quantum cryptography.

14 Hours

Tools / Languages: SEED labs, gpg, Python Programming Language

Textbook (s):

1: Understanding Cryptography: A Textbook for Students and Practitioners, Christoff Paar and Jan Pelzl, Springer 2010.
2: Cryptography and Network Security: Principles and Practice, William Stallings, 7th Edition, Pearson, 2017.
3. Cryptography and Network Security, Behrouz A. Forouzan, 3rd Edition, Tata McGraw Hill, 2017.

Reference Book (s):

1: Introduction to Modern Cryptography", Jonathan Katz, Yehuda Lindell, 2nd Edition, CRC Press, 2015.

UE21CS342AA5 : Augmented and Virtual Reality (4-0-0-4-4)

This course presents an introduction to virtual and augmented reality technologies, with an emphasis on designing and developing interactive virtual and augmented reality experiences using blender and Unity-3D.

Course Objectives:

- Introduce the use of geometric transformations on graphics objects, their application in composite form and its implementation.
- Impart the basics of computer graphics and Introduce graphics programming using OpenGL and Graphics Pipeline.
- Introduce Virtual and Augmented Reality essentials.
- Understanding human physiological aspects with respect to virtual reality applications and user interfaces.
- Understanding of use of Artificial Intelligence in the field of Virtual and Augmented Reality and applying the concepts to create creative virtual experiences.

Course Outcomes:

At the end of this course, the student will be able to:

- Apply techniques and methods of augmenting virtual objects in real space.
- Demonstrate the fundamentals of computer graphics and display pipeline systems and use OpenGL for complex 3D graphical visualization and demonstrate its applications.
- Apply techniques and tool to design a immersive virtual reality experience.
- Apply graphics in greater depth to more complex aspects of Image Processing, Tracking and Human Computer Interfaces, etc.
- Apply AI models and methods for creating dynamic interactive and adaptable virtual spaces.

Desirible Knowledge : UE21CS252A- Data Structures and its Applications.

Course Content:

Unit 1: Geometric Objects and Transformations

Scalars, Points and Vectors, Three-Dimensional Primitives, Coordinate Systems and Frames, Modelling a Coloured Cube, Overview of 2D Transformations: Rotation, Translation and Scaling, Affine transformations, Transformation in Homogeneous Coordinates, Concatenation of Transformations, OpenGL Transformation Matrices, Interfaces to Three Dimensional Applications, Quaternion's.

14 Hours

Unit 2: Graphical System and Programming and 3D Modelling.

The Programmer's Interface, Graphics Architectures, Programmable Pipelines. Graphics Programming: Programming Two Dimensional Applications. The OpenGL: The OpenGL API, Primitives and Attributes, Colour, Viewing, Control Functions, Polygons, Viewing, Control Functions, the gasket Program, Polygon and Recursion, The Three-dimensional gasket, adding interaction, adding menus.

14 Hours

Unit 3: Augmented and Virtual Reality.

Introduction to Augmented Reality: Definition and Scope, A Brief History, Examples, Requirements and Characteristics: Methods of Augmentation, Spatial Display Models, Visual Display, Stationary Tracking Systems, Mobile Sensors. Introduction: What is Virtual Reality, Modern VR Experience. Bird's Eye View: hardware, Software. Eye movement and its implications or VR. Tracking: 2D and 3D orientation, Tracking Position and Orientation, Tracking Attached bodies.

14 Hours

Unit 4: IO modalities, AI, and Behaviour in VR

Computer Vision and Augmented Reality; marker tracking, Multiple-Camera Infrared Tracking, Natural Feature Tracking by Detection, Incremental Tracking, Simultaneous Localization and Mapping, Outdoor Tracking, Interaction: Output and input modalities, Haptic interaction and Multimodal interaction. 3D Scanning of environments. Reactive AI: Adaptability, Complexity and Universality, Feasibility, More Intelligence in the System: Deliberative AI, Reinforcement learning through interaction, Imitation Learning through human demonstration.

14 Hours

Tools/ Languages: C/ C++/ JAVA/ Python using OpenGL.

Text Book(s):

1: “Interactive Computer Graphics - A top-down approach with shader-based OpenGL”, Edward Angel and Dave Shreiner, Pearson Education, Sixth edition, 2012.
2. “Steven M. LaValle”, Virtual Reality. Cambridge University Press, 2017, http://vr.cs.uiuc.edu/ (Links to an external site.) (Available online for free)
3. “Creating Augmented and Virtual Realities”, Erin Pangilinan, Steve Lukas, Vasanth Mohan, O'Reilly Media, Inc., 2019.

Reference Book(s):

1: “Interactive Computer Graphics: A Top -Down Approach with WebGL”, Edward Angel, Pearson Education, 7th Edition, 2015.
2: Unity Game Development in 24 Hours , Geig, Mike. Sams Teach Yourself . Pearson Education, 2014.
3. “OpenGL Programming Guide”: Mason Woo, Jackie Neider, Tom Davis, Dave Shrenier: 3rd Edition, openGl version 1.2, Addision Wesley, 1999.
4. Blender 3D Basic, Gordon Fisher, PACKT Publishing, 2nd Edition (Note: For working with recent version the course material for UNIT 4 can be substituted with appropriate web content.
5. D. Schmalstieg and T. Höllerer. Augmented Reality: Principles and Practice. Addison-Wesley, Boston, 2016, ISBN-13 978-0-32-188357-5

UE21CS342AA6: Human Computer Interaction (4-0-0-4-4)

This course provides an overview and introduction to the field of human-computer interaction, with an emphasis on what HCI methods and HCI-trained specialists can bring to design and development teams for all kinds of products. The course will introduce students to tools and techniques for creating or improving user interfaces.

Course Objectives:

- Get Familiar with the psychology underlying user-interface and usability design guidelines keeping in mind human behavioral and perceptual capabilities and limitations that affect interface design.
- Get Familiar with the basic principles of Goal-directed user interface design and standard patterns and key modeling concepts involved in Visual interface design for software interfaces.
- Aware of development methodologies and life cycle models for building user interfaces and prototyping in user interface design and how to test them.
- Get insights with the impact of usable interfaces in the acceptance and performance utilization of information systems.
- To be aware of the importance of working in teams and the role of each member within an interface development phase

Course Outcomes:

- Develop and Incorporate a Conceptual Vocabulary for Analysing Human Interaction with software affordance, conceptual model, feedback.
- Interpret and Exemplify how User-Centered design complements other software process models.
- Apply low fidelity (lo-fi) prototyping techniques to gather and report user responses
- Analyze and Define a User-Centred design process that explicitly takes into account the fact that the user is not from the Ecosystem of the Developer.
- Evaluate and Pick appropriate methods to support the development of a specific UI.

Course Content:

Unit 1 : Basics of Design

The Human: I/O Channels , Memory, Reasoning and Problem Solving, The Computer, Devices, Memory, Processing and Networks; Interaction: Models, Frameworks, Ergonomics, Styles, Elements, Interactivity, Paradigms, Design Focus, Interaction Design basics.

14 Hours

Unit 2 : Design Thinking

Process, Scenarios, Navigation, Screen Design, Iteration and Prototyping, HCI in software process, Software Life Cycle, Usability Engineering, Prototyping in Practice, Design Rationale, Design Rules, Principles, Standards, Guidelines, Universal Design, Cognitive model.

14 Hours

Unit 3 : Design, Analyzing and Models

Socio-Organizational issues and Stakeholder Requirements, Communication and Collaboration models, GUI Design and Aesthetics, Task Analysis, Task Decomposition, Knowledge based Analysis Technique, Dialog notations and design.

14 Hours

Unit 4 : Notation and Multimedia Design

Petri nets and State Charts, Models of the system, Modeling rich interaction, Case study - A chosen Mobile APP as an Interactive System, Groupware, Ubiquitous Computing, Virtual and Augmented Reality, Hypertext, Multimedia and World Wide Web, Augmented Reality Toolkit practice session.

14 Hours

Tools/ Languages: Figma, Adobe Creative Cloud.

Text Book(s):

1: “Human Computer Interaction”, Dix A., Finlay J, Abowd G. D. and Beale R., 3rd Edition, Pearson Education, 2005.

Reference Book(s):

1:“Designing the User Interface”, B. Shneiderman; Addison Wesley 2000 (Indian Reprint).
2: “About Face: The Essentials of Interaction Design by Alan Cooper”, Robert Reimann, David Cronin. Christopher Nooessel, 4th Edition, WILEY, 2014.
3: “Don’t Make Me Think, Revisited: A Common Sense Approach to Web and Mobile Usability”, Steve Krug (Author), 3rd Edition.(e-Book)
4: “The Design of Everyday Things”, Don Norman, 2nd Edition, 2013.

UE21CS343AB1: Unix System Programming (4-0-0-4-4)

This course provides a deep understanding of the operating system architecture and low-level interfaces (principally, system calls and library functions) that are required to build system-level, multithreaded, and network applications on Linux and UNIX systems.

Course Objectives:

- Understand the basic concepts of System Programming.
- Understand the concepts Unix File systems and Process control.
- Gain expertise on UNIX kernel data structures and system calls.
- Write sophisticated applications to manipulate system resources such as files, processes and system information.
- Understand the concepts of signals and programming with signals.

Course Outcomes:

At the end of this course, the student will be able to:

- Explain basic concepts in systems programming.
- Create, modify and maintain nested directories and file systems and administer permissions.
- Manage Processes, Threads and Shared Libraries and establish synchronized execution.
- Implement Inter-Process Communication and asynchronous Signal mechanisms.
- Utilize Unix OS, efficiently and seamlessly through Shell command interface and System calls.

Course Content:

Unit 1 : Unix System Overview:

Introduction, UNIX Architecture, File I/O: Introduction, File Descriptors, open Function, create Function, close Function, lseek Function, read Function, write Function, I/O Efficiency, File Sharing, Atomic Operations, dup and dup2 Functions, fcntl Function, ioctl Function.

12 Hours

Unit 2 : Files & Directories & Process Environment

Introduction, stat, fstat, & lstat Functions, File Types, Set–User–ID and Set–Group–ID, File Access Permissions, Ownership of New Files and Directories, access Function, umask, chmod, fchmod Functions, Sticky bit ,chown, fchown, & lchown Functions, File Size, File Truncation, File Systems, link, unlink, remove, & rename Functions, Symbolic Links, symlink & readlink Functions, File Times, utimeFunction, mkdir, & rmdir Functions, Reading Directories, chdir, fchdir getcwd Functions.

Process Environment :Introduction, main Function, Process Termination, Command–Line–Arguments, Environment List, Memory Layout of a C Program, Shared Libraries, Memory Allocation, Environment Variables, setjmp and longjmp Functions, getrlimit and setrlimit Functions

16 Hours

Unit 3: Process Control, Process Relationships & Signals Process Control

Introduction, Process Identifiers, fork, vfork, exit Functions, wait, waitpid, Functions, Race Conditions, exec Functions, system Function. Introduction, Process Groups, Sessions, Controlling Terminal, Job Control, Shell Execution of Programs, /proc file system

Signals: Introduction, Signal Concepts, signal Functions, Unreliable Signals, Interrupted System Call, Reentrant Functions, SIGCLD Semantics, kill, raise Functions, alarm, and pause Functions, Signal Sets, sigprocmask Function, sigpending, sigaction Functions, sigsetjmp, siglongjmp Functions, sigsuspend, abort Functions, system, sleep Functions.

16 Hours

Unit 4 : Threads and Interprocess Communications

Interprocess Communication: Introduction, Pipes, popen, & pclose Functions, Coprocesses, FIFOs, XSI IPC, Message Queues, Semaphores, Shared Memory, Client – Server Properties. Threads: Introduction, Thread Concepts, Thread Identification,Thread Creation, Thread Termination, Thread Synchronization, Mutexes, Deadlock Avoidance, Condition Variables.

12 Hours

Tools/ Languages : C , C++ Programming Language , Unix OS

Text Book(s):

1. “Advanced Programming in the UNIX Environment”, W. Richard Stevens, Stephen. A. Rago, Addison- Wesley Professional, 3 rd Edition, 2013.

Reference Book(s):

1. “UNIX system programming using C++”, Terrence Chan, Pearson Education India, 1st Edition, 2015.
2. “UNIX Systems Programming: Communication, Concurrency, and Threads”, Kay A. Robbins, Steven Robbins, Prentice Hall Professional, 2003.

UE21CS343AB2 : Big Data (4-0-0-4-4)

The course introduces various Big Data technologies that are used to analyze large amounts of data either in batch mode or streaming mode. It focuses on both processing and storage technologies and looks at how algorithms need to be modified to work with large amounts of data. This course requires the student to have a desirable knowledge of Data Structures and its Applications and Design and Analysis of Algorithm.

Course Objectives:

- Provide an introduction to Big Data.
- Introduce computational and storage technologies for Big Data.
- Introduce algorithms for processing Big Data and programming tools focussing on practical issues in working with Big Data.
- Learn application of Big Data techniques to various real-life problems

Course Outcomes:

At the end of this course, the student will be able to:

- Explore various characteristics of Big Data Problems.
- Evaluate principles and design alternative computational/storage technologies for Big Data.
- Design Big Data applications using available infrastructure for Big Data through practical assignments.
- Applying Big Data techniques in real life problems through a group-based project.

Desirable Knowledge : UE21CS252A-Data Structures and its Applications, UE21CS241B- Design and Analysis of Algorithms.

Course Content:

Unit 1 : Introduction

Big Data definition, Challenges and opportunities with Big Data, Data intensive scientific discovery and the role of Big Data, History, Map Reduce – Storage (HDFS), Computation model, Map Reduce architecture, Overview of Hadoop Ecosystem, YARN introduction, Case Study: HIVE.

14 Hours

Unit 2 : Big data infrastructures (Compute/Storage)

Introduction to sample Big Data Algorithms – matrix multiplication and pagerank, Issues with Hadoop, Relational operators on Map-reduce, Complexity of Big Data algorithms – Communication Cost complexity model. case study: HBase.

14 Hours

Unit 3 : In memory computation

Spark and Scala/PySpark programming model, Transformations and Actions, Spark SQL, Spark architecture – RDD, DataFrames, Wide and Narrow dependencies, Streaming Algorithms –sampling, set membership – Bloom Filters counting, counting unique elements – Flajolet Martin Algorithm.

14 Hours

Unit 4 : Streaming analysis and advanced analytics on Big Data

Streaming analytics use cases, Streaming Spark, Kafka – use cases, architecture, Clustering algorithms - k means and Collaborative filtering, Scaling Neural Networks for Big Data, Case Study MLLib.

14 Hours

Tools/ Languages: Hadoop, HDFS, Spark, Streaming Spark, HIVE, HBase, Mllib.

Text Books:

1: “Big Data Analytics”, Rajkamal, Preeti Saxena, 1st Edition, McGraw Hill Education, 2019.
2: “Big Data Simplified”, Sourabh Mukherjee, Amit Kumar Das, Sayan Goswami, 1stEdition, Pearson, 2019.

Reference Book(s):

1: “Mining of Massive Datasets”,Anand Rajaraman,Jure Leskovec,Jeffrey D. Ullman, Cambridge Press, 2014.

2: “Big Data Analytics Beyond Hadoop: Real-Time Applications with Storm, Spark, and More Hadoop Alternatives”,Vijay Srinivasa Agneeswaran, Pearson Education, 2014.

UE21CS343AB3: Graphs Theory and its Applications (4-0-0-4-4)

This course focuses on mathematical structure to model the relations between the objects. It also discusses about the basics of graph theory together with a wide range of applications to different branches of Science and Technology, and to real-world problems. The course includes principles of Graph database and social networks.

Course Objectives:

- Familiarize with concepts and abstraction of Graph Theory.
- Up skill students with computer representation of graphs and algorithms.
- Introduce students with advanced concepts in graph theory and applications.
- Introduce students to graphs in Graph Database.
- Up skill students with principles and tools for social network analysis.

Course Outcomes:

At the end of this course, the student will be able to:

- Understand the graph theory concepts, abstractions and results to model real-world problems.
- Implement high performance computer representation and graph algorithms.
- Understand various applications of graph theory in varied discipline.
- Introduce students to Graph Databases.
- Apply principles of graph to analyze social networks.

Desirable Knowledge: UE21CS151B – Problem Solving with C, UE21CS252A – Data Structures and its Applications.

Course Content

Unit 1: Introduction, paths, cuts and planar Graphs

Introduction – Review of Representation and Traversals, Walks, Paths, Circuits Euler graphs Hamiltonian paths and circuits – Directed graphs, Digraphs and binary relations, Trees – Properties of trees, Rooted and binary trees, spanning trees, cut sets – Properties of cut set – All cut sets – Fundamental circuits and cut sets – Connectivity and Separability – Network flows, Isomorphism– Combinational and geometric graphs, Planar graphs –Different representation of a planar graph,

14 Hours

Unit 2: Graph Coloring, Graph Applications

Chromatic number – Chromatic partitioning – Chromatic polynomial, Matching, Covering, Four Colour problem, Register Allocation using graph coloring.Minimum Spanning Trees, Shortest Path Problem, Articulation Points, Network Flow Algorithms, Chinese Postman Problem, Time Table Problem, Strongly Connected Components

14 Hours

Unit 3: Graph Databases and Graphs in Social Networks

Graph Database: RDBMS vs Graph Database, Neo4j Basics, Neo4j Graph Data Model, Nodes, Properties, Relationships, Labels, Neo4j CQL Commands and Functions. Introduction to Graphs in Social Networks:Graphs in Social Networks: Centrality measures: Degree, Betweenness, Closeness, Eigen vector, Katz and PageRank Centrality, Measures of cohesiveness: Degree distribution, Diameters, Transitivity and Reciprocity, Clustering coefficient, Groups and substructures: top-down view – weak and strong component, in-component, out-component, giant component.

14 Hours

Unit 4: Graphs in Social Networks and social graph dynamics

Bottom-up view: clique, N-clique, N-clan, N-club, N-core. Community detection: cluster vs. community, Overlapped Communities by CPM, Girvan Newman algorithm, Louvain algorithm **, Generative Models -** Small World: Milgram's small world experiment. Scale-free networks Random Graph Models, Barabasi and Albert's model, Watts-Strogatz small world model, Myopic Search. **Dynamics on Graph** : Herd behaviour, Diffusion-Diffusion of innovation, Bass model of diffusion – epidemic modelling, Cascades and Contagions, Percolation and Robustness of Networks, Effects of communities and centralities on diffusion

14 Hours

Tools / Languages: Python, Neo4j, NetworkX

Text Book(s):

1: “Graph Theory: With Application to Engineering and Computer Science”, Narsingh Deo, Prentice Hall of India, 2017.
2: “Graph Databases: New Opportunities for connected data”, 2nd Edition, Ian Robinson, Jim Webber, Emil Eifrem, Oreilly Publications, 2015.
3: “Networks, Crowds, and Markets: Reasoning about a Highly Connected World”, David Easley, Jon Kleinberg, Cambridge University Press, 2010
4: “Social Media Mining”, Reza Zafarani, Mohammad Ali Abbasi, Huan Liu, Cambridge University Press, 2014
5: “Network Science”, Albert-Laszlo Barabasi, e-book

Reference Book(s):

1: “Graph Theory”, F. HARARY, Addison-Wesley,1969.
2: “Discrete and Combinatorial Mathematics”, Ralph P.Grimaldi&B.V.Ramana ,5th Edition, PHI/Pearson education.
3: “Graph Theory with Applications”, J A Bondy and U. S. R Murthy, Elsevier Science Publishing Co. ., Inc.
4: “Graph Representation Learning”, William L Hamilton, Morgan and Claypool Publishers, 2020.

UE21CS343AB4: Bio-Inspired Computing (4-0-0-4-4)

The field of natural computing has been the focus of a substantial research effort in recent decades. These bio inspired computing algorithms have proven to be successful problem solvers across domains as varied as management science, telecommunications, business analytics, bioinformatics, finance, marketing, engineering, architecture and design, to name but a few. This course provides a comprehensive introduction to bio inspired computing algorithms. This course requires the student to have a desirable knowledge of Design and Analysis of Algorithms.

Course Objectives:

- Introduce fundamental topics in bio-inspired computing.
- Build up their proficiency in the application of various algorithms in real-world problems.
- Provide an understanding of a range of features from the biological world that have influenced the world of computing.
- Foster a basic understanding of the nature biological inspiration for AI and Computing - the goals and motivations.
- Provide experience of collaborative work that develops biologically inspired solutions to practical problems.

Course Outcomes:

At the end of this course, the student will be able to:

- Understand some of the essential features of biologically inspired systems.
- Develop an understanding of simple computer modelling of biological systems.
- Develop a foundation for biological learning models and self-organisation.
- Understand the strengths, weaknesses and appropriateness of nature-inspired algorithms.
- Apply nature-inspired algorithms to optimization, design and learning problems.

Desirable Knowledge: UE21CS241B – Design and Analysis of Algorithms.

Course Content:

Unit 1 : Introduction

Natural Computing Algorithms: An Overview, Evolutionary Computing: Introduction to Evolutionary Computing, Evolutionary Algorithms: Genetic Algorithm: Canonical Genetic Algorithm, Design Choices in Implementing a GA, Choosing a Representation, Initialising the Population, Measuring Fitness, Generating Diversity, choosing Parameter Values Extending the Genetic Algorithm: Dynamic Environments, Structured Population Gas, Constrained Optimisation, Multi objective Optimisation, Memetic Algorithms, Linkage Learning, Estimation of Distribution Algorithms.

14 Hours

Unit 2 : Evolution Strategies and Evolutionary Programming

The Canonical ES Algorithm, Evolutionary Programming, Differential Evolution: Canonical Differential Evolution Algorithm, Extending the Canonical DE Algorithm, Discrete DE, Genetic Programming: Genetic Programming, Bloat in GP, More Complex GP Architectures, GP Variants, Semantics and GP.

14 Hours

Unit 3 : Social Computing: Particle Swarm Algorithms, Other Foraging Algorithms

Search, Particle Swarm Optimisation Algorithm, Comparing PSO and Evolutionary Algorithms, Maintaining Diversity in PSO, Hybrid PSO Algorithms, Discrete PSO, Evolving a PSO Algorithm. Ant Algorithms: A Taxonomy of Ant Algorithms, Ant Foraging Behaviours, Ant Algorithms for Discrete Optimisation, Ant Algorithms for Continuous Optimisation, Multiple Ant Colonies, Hybrid Ant Foraging Algorithms, Ant-Inspired Clustering Algorithms, Classification with Ant Algorithms, Evolving an Ant Algorithm. Honeybee Dance Language, Honeybee Foraging, Designing a Honeybee Foraging Optimisation Algorithm, Bee Nest Site Selection, Honeybee Mating Optimisation Algorithm; Non-uniform Oscillators and Firefly, the model and optimization

14 Hours

.

Unit 4: Other Social Algorithms and Immuno computing: Artificial Immune Systems

Other Social Algorithms: Glow Worm Algorithm, Bat Algorithm, Fish School Algorithm, Locusts.

Artificial Immune Systems: The Natural Immune System, Artificial Immune Algorithms, Negative Selection Algorithm, Dendritric Cell Algorithm, Clonal Expansion and Selection Inspired Algorithms, Immune Programming,The Future of Natural Computing Algorithms, Looking Ahead, Open Issues.

14 Hours

Tools/Languages: Matlab.

Text Book(s):

1: "Natural Computing Algorithms", Anthony Brabazon, Michael O'Neill, Seán McGarraghy, Springer, Natural Computing Series,2015.

Reference Book(s):

1:"Fundamentals of Natural Computing: Basic Concepts, Algorithms, and Applications",Nunes de Castro, Leandro , Chapman & Hall/ CRC, Taylor and Francis Group, 2007

2: "Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies",Floreano D. and Mattiussi C., MIT Press, Cambridge, MA, 2008.

UE21CS343AB5 : Advanced Computer Networks (4-0-0-4-4)

Advanced Computer Networks course is designed for students who have basic knowledge of networking (OSI model, TCP/IP, ARP etc.). The course builds its contents by considering a mixed approach of involving theory and practical. From this perspective, the course introduces advanced topics that are basic blocks of networking and covers fundamentals that are used to develop few of the advanced technologies, including routing protocols and switching. The course aims to provide information so that at the end of the course, the students find themselves comfortable in taking either of the direction- industrial job or further research in networking.

Course Objectives:

- Provide in-depth study of dynamic and distance vector routing protocols.
- To learn about Layer 2 Switching in detail.
- To get an insight of Link State routing protocols.
- To explore the new approach of Networking, SDN and NFV.

Course Outcomes:

At the end of this course, the student will be able to:

- Designing a network topology by demonstrating the ability to configure routers with different routing techniques.
- Understand the concepts of Layer 2 switching, VLANs and its usage..
- Designing networks with one or multiple routing protocols and investigating its behaviour/features.
- Comprehend features of Software Defined Networking (SDN) for next generation systems.

Desirible Knowledge : UE21CS252B – Computer Networks.

Course Content:

Unit 1: Dynamic and Distance Vector Routing, Access Lists

IP Routing Process, Static routing – IPv4/IPv6, Floating Static Routes, Load Sharing, Fast and Process Switching
Distance Vector Routing – RIPv2 packet format, classless Routing, compatibility with RIPv1, Metric Calculation, Route Summarization, Standard IP and Extended IP access lists, Calling access lists, Named access lists.

14 Hours

Unit 2: Exterior Gateway Routing Protocol (EIGRP), Layer 2 Switching, Spanning Tree Protocol (STP), Virtual LAN (VLAN)

EIGRP – Neighbor discovery, Reliable Transport Protocol (RTP), Diffusing Update Algorithm (DUAL), VLSM and Summary Routes, Metrics – Maximum Paths & Hop Count, Load Balancing with EIGRP, Discontiguous Networks, Auto summarization, passive interface
Layer 2 switching, Bridging vs LAN Switching, Forwarding Table, Spanning Tree Protocol and its operations, VLAN Basics, Static and Dynamic VLANs, VLAN Trunking Protocol (VTP), Routing between VLANs, configure VLANs and Inter-VLAN routing, MAC VLANs, VXLAN.

15 Hours

Unit 3: Link State Routing Protocols – OSPF & BGP

Open Shortest Path First (OSPF)v2 – OSPF packet format, Network Types, Designated Routers and Backup Designated Routers (DR & BDRs), OSPF Neighbors, Areas – Intra, Inter, External, Router Types, Virtual Links, Link State Database, LSA Types, Metric Calculation, Summary Routes, Route redistribution
Border Gateway Protocol (BGP) – BGP Packet Format, External and Internal BGP

14 Hours

Unit 4: Introduction to Software Defined Networks (SDN) and Network Function Virtualization (NFV)

Segment Routing, SDN Control and Data Plane, SDN Controller and Network-control Applications, OpenFlow Protocol, Data and Control Plane Interaction. Network Virtualization architecture, Management, Control and Data Planes, Virtual Switches, Micro-segmentation.

13 Hours

Tools / Languages: GNS3, Cisco Packet Tracer.

Text Books:

1. “Routing TCP/IP volume 1”, 2nd Edition, Jeff Doyle, Cisco Press 2005.
2. “Computer Networking: A Top-Down Approach”, James F. Kurose, Keith W. Ross, 7th Edition, Pearson Publication, 2017 [For SDN]

Reference Books:

1: “Software-Defined Networks: A Systems Approach”, Peterson, Cascone, O’Connor, Vachuska, and Davie, https://sdn.systemsapproach.org/index.html [For NFV]

2: “TCP IP Protocol Suite”, Behrouz Forouzan, 4th Edition, McGraw-Hill, 2010 [For BGP]

Labs/Assignments:

1. Applying IP addressing scheme to design a network
2. Setting up Dynamic Static Routing for the designed network and observing its behaviour
3. Setting up using different routing protocols. Configuring using EIGRP/OSPF/BGP, observing the metrics, route redistribution
4. For the setup in 3, apply VLAN configuration and access lists and observe its behaviour.

UE21CS343AB6: Computer Network Security (4-0-0-4-4)

This course will give an overview and conceptual understanding of network related security aspects. Students will have opportunities to dwell well into technical "how to" with hands-on sessions and case study discussions.

Course Objectives:

- To provide an overall view of Network Security and introduce the concept of packet analysis.
- To understand the security problems in the design and implementation of the TCP, IP/ICMP, ARP protocols.
- To learn the vulnerabilities in DNS protocol and to implement and experiment with Firewall rules.
- To provide an overview of network management techniques and implementation of VPN.
- To understand concepts of risk management and security aspects of wireless networks.

Course Outcomes:

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

- Sniff packets from clients and analyse them to extract important info such as headers, passwords etc.
- Launch DoS and MITM attacks using various protocol vulnerabilities and mitigate them.
- Configure firewalls on Linux machines and exploit vulnerabilities on DNS protocol.
- Design and implement VPN for a secure connection over the internet.
- Master in wireless network security systems in depth and perform effective network management.

Desirable Knowledge: UE21CS252B – Computer Networks

Course Contents:

Unit 1: Introduction, Packet Sniffing & Spoofing, MAC Layer Attacks, Network Layer Attacks

CIA principles, Attack surface and types, Assets, Vulnerabilities and Threats, Countermeasures, Privacy, General Data Protection Regulation, Security vs Privacy, Data Breaches. Real Life Examples of Cyber Crime, Security framework, Job outlook. Packet Sniffing and Spoofing: Introduction, Sending packets: Network Interface Card (NIC), BSD packet filter (BPF). Packet sniffing: Receiving packets using sockets, Packet sniffing using Raw sockets, Packet sniffing using PCAP API, Processing captured packets. Packet spoofing: Sending normal packets using sockets, Constructing spoofed raw ICMP packets and UDP packets. Sniffing and then spoofing, Python vs Scapy, Endianness. MAC layer and attacks: The MAC layer, ARP protocol, ARP cache poisoning attacks, MITM using ARP cache poisoning, Demo, Countermeasure. Network layer: IP, ICMP and attacks: IP protocol, IP fragmentation, Attacks using IP fragmentation: Problem and solution, Routing, and spoofing prevention, ICMP protocol, ICMP redirect attack, Smurf and other ICMP attacks.

16 Hours

Unit 2: TCP Attacks and DNS Attacks

Attacks on the TCP protocols: TCP overview, Send and receive buffers, SYN flood attack: TCP 3-way handshake, the SYN flooding attack, Launching the attack using Netwox and C, Countermeasure. TCP reset attack: TCP reset attack on Telnet, SSH and video streaming connections. TCP session hijacking attack: TCP session and session hijacking, Launching the attack, Hijacked TCP connection. Reverse shell: working, redirecting IO to TCP connection, Creating reverse shell. Countermeasure. DNS Attacks: DNS hierarchy, zones and servers, DNS query process, Constructing DNS request and response using Scapy, DNS attacks: Overview, Local DNS cache poisoning attack, Remote DNS cache poisoning attack (Kaminsky attack), Reply forgery attacks from malicious DNS servers, Countermeasure against DNS spoofing attacks, DoS attacks on DNS servers. Case Study – 1.

12 Hours

Unit 3: Firewall, IDS, IPS, Honeypots

Firewall: Introduction, Requirements of a firewall, Firewall characteristics and Access policy, Types of firewalls, NG firewall, Shortcomings, Firewall location and configuration: DMZ networks, Firewall topologies. Introduction, Build a simple firewall, Netfilter, iptables firewall in Linux, Stateful firewall and connection tracking, Application/Proxy firewall and Web proxy, Evading firewalls. Intrusion Detection and Prevention: Intruders, Intrusion detection, Analysis approaches, Host-based intrusion detection, Network-based intrusion detection, Distributed or hybrid intrusion detection, Honeypots, Example system: Snort.

14 Hours

Unit 4: Virtual Private Networks

Virtual Private Network: Introduction, Why VPN, analogy, and tunnelling. Overview of TLS/SSL VPN: Establishing a tunnel, Forwarding, and releasing IP packets, TLS/SSL VPN details. Building, Setup and Testing VPN. Bypassing Firewall using VPN. Case Study – 2. The Heartbleed Bug and Attack: Introduction and the Heartbeat protocol, Launching the attack, Fixing the Heartbleed bug. Wireless Security: Communications and 802.11 WLAN standards: Wired Equivalent Privacy (WEP), Wireless Protected Access (WPA), IEEE 802.1x, 802.11i/ WPA2, Wireless Network Threats. SOC and SIEM.

14 Hours

Note: Hands-on experience for relevant topics in the form of Lab and/or Assignment are given. Relevant cyber security case study for undergraduate students is discussed.

Tools / Languages: SEED Ubuntu VM, Wireshark, Snort, Netwox, Scapy.

Text Book (s):

1: "Computer & Internet Security: A Hands-on Approach", Wenliang Du, 2nd Edition, 2019.

Reference Book (s):

1: "Computer Security: Principles and Practice", William Stallings and Lawrie Brown, Pearson Education, 3rd Edition, 2010.

UE21CS351B: Cloud Computing (4-0-2-5-5)

The cloud computing course introduces not only the various technologies that go into building a cloud native application, but also how cloud systems are designed. The student is introduced to various tools and design techniques/tradeoffs. It also gives a flavour for the business relevance/ethics of using cloud computing. This course requires the student to have a desirable knowledge of Computer Networks and Operating System.

Course Objectives:

- Introduce the rationale behind the cloud computing revolution and the business drivers
- Introduce various models of cloud computing.
- Introduce differences between traditional monolithic applications and cloud native application architectures
- Introduction on how to design cloud native applications, the necessary tools and the design tradeoffs and how to design distributed systems for scalability and expose the student to various tradeoffs in designing cloud architectures.

Course Outcomes:

At the end of this course, the student will be able to:

- Comprehend the technical and business rationale behind cloud computing.
- Decide the model of cloud computing to use for solving a particular problem.
- Build and deploy applications for the cloud and understand the security implications.
- Apply the fundamentals of distributed systems design to cloud computing and Demonstrate design tradeoffs while designing cloud applications.

Desirable Knowledge: UE21CS252B- Computer Networks, UE21CS241B- Operating Systems.

Course Content:

Unit 1 : Cloud Programming Models

Parallel computing, Grid computing, Introduction to Cloud Programming Models and service Models, Introduction to technology challenges with Distributed & Cloud computing, Business Drivers - deployment models, Cloud architecture and IaaS programming model, Web Services and REST, PaaS Programming Model, Communication using Message queues- Pub Sub model, SaaS Programming model – Microservices and differences with the traditional monolithic model; challenges of migrating monolithic applications**.**

14 Hours

Unit 2 : Virtualization

Hypervisor - Types, Para virtualization and Transparent virtualization, Software - Trap and Emulate virtualization, Software - Binary translation, Goldberg Popek principles for Virtualization, Hardware - AMDv/Intel, Memory - Shadow page tables, Memory - Nested page tables, IO, VM Migration, Lightweight Virtualization - Containers and Namespaces, Deployment of cloud native applications through Docker – Unionfs, DevOps, Orchestration and Kubernetes.

14 Hours

Unit 3 : Distributed Storage

Types of Cloud storage - Block, Object stores, Replication, lag, multileader replication, Leaderless replication, Partitioning - key-value data, Consistent hashing, Partitioning - rebalancing partitions, Request routing, Consistency Models, CAP Theorem, Transactions, Two-phase commit.

14 Hours

Unit 4 : Cloud Controller, Performance, Scalability and Security

Master-slave v/s p2p models, Resource allocation, Scheduling algorithms, Cluster coordination – consensus, Fault Tolerance - faults and partial failures, Failure detection - checkpointing and application recovery, Unreliable communication, Cluster coordination - leader election, distributed locking, Case Study: Zookeeper - distributed consensus infrastructure.

Scaling computation - reverse proxies, Scaling computation - hybrid cloud and cloud bursting, Multitenancy, Multitenant databases

Cloud security requirements - physical/virtual security, Risk management, security design patterns, Authentication in the cloud: Keystone, Cloud Threats – DoS, Economic Denial of Sustainability**.**

14 Hours

Tools/Languages: Amazon AWS (or equivalent), AWS Skill Builder, AWS Educate, Qwiklabs, Docker, Kubernetes, Jenkins, Zookeeper, Github, NoSQL database, Flask, Python, GoLang

Text Book(s):

1: "Distributed and Cloud Computing", Kai Hwang, Jack Dongarra, Geoffrey Fox.ISBN: 978-0-12-385880-1, Morgan Kaufmann, 2012.
2:"Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems", Martin Kleppmann. O'Reilly, 2017.

Reference Book(s):

1: "Docker in Action", Jeff Nickoloff, Manning Publications, 2016.
2: "Cloud Native DevOps with Kubernetes", John Arundel and Justin Domingus, OReilly, 2019.
3: "Moving to the clouds: Developing Apps in the new world of cloud computing", Dinkar Sitaram and Geetha Manjunath. Syngress, 2011.

UE21CS352B: Object Oriented Analysis and Design using Java (4-0-2-4-4)

In this course students will learn to perform Analysis on a given domain and come up with an Object-Oriented Design (OOD). Various techniques will be discussed and practiced which are commonly used in analysis and design phases in the software industry. Unified Modelling Language (UML) will be used as a tool to demonstrate the analysis and design ideas and an object-oriented programming language such as Java would be used to implement the design. The theory is supplemented with implementations which are demonstrated/ practiced in class which provides the hands-on experiences of implementing the patterns.

Course Objectives:

- Familiarize students with static and dynamic models of object-oriented analysis and modelling using the unified modelling language (UML)
- Introduce students to object oriented programming concepts in Java
- Make students appreciate the importance of system architecture design in software development
- Introduce the students to understand the importance of GRASP and SOLID design principles.

Course Outcomes:

At the end of this course, the student will be able to:

- Construct static models, use cases and class models, followed by analysing the dynamics of the system using activity, sequence, state and process models. Depict the architecture of a software system by using component and deployment models
- Use the concepts of classes and objects of object-oriented programming in Java to model a complex system
- Use GRASP and SOLID principles in the design of software application and apply Creational software design patterns for variety of application scenarios
- Apply Structural and Behavioural software design patterns for variety of application scenarios. Understand Anti-patterns

Course Content:

Unit 1: Advanced OO, Object Oriented Analysis and Static Models and Diagrams

OOA: Requirements, Modelling and Analysis, Introduction to UML, Use Case Modelling: Use Cases Diagrams. Class Modelling: UML Class Diagrams, OO relationships, CRC Diagrams, Component model, Deployment model, Activity Modelling: UML Activity Diagrams and Modelling, Guidelines. Behaviour Modelling: Sequence Diagram, UML State Machine Diagrams and Models, Advanced State Models

Introduction to Object Orientated Programming

Object-oriented Programming: JVM, Abstraction, Encapsulation, Composition, Class Attributes, Behaviour, Objects, and Methods

14 Hours

Unit 2: Object Orientated Programming and Architecture design

Interface and Implementation: Role of Constructors and Destructors, Garbage Collector, Parameter Passing, Value Type and Reference Type, Overloading of Methods, Class Attributes and Behaviour: Difference between Class Methods and Instance Methods, Inheritance: Concepts of Single Rooted Hierarchy and Interface, Abstract Class in Programming Languages, Object Class in Java, Collection, Array, List and Stack; OO Development process, System Design and Frameworks, Architectural patterns, MVC architectural pattern

14 Hours

Unit 3: Design principles

GRASP and its application to Object Design, Creator, Information Expert, Low Coupling, Controller, High Cohesion, Polymorphism, Pure Fabrication, Indirection and Protected Variations

SOLID: Single Responsibility, Open-Closed, Liskov Substitution, Interface Segregation, Dependency Inversion

OO Design Principles and Sample Implementation of Patterns in Java

Introduction to Design Patterns, selection and usage of a design pattern, Creational Design Patterns Theory and Implementation in Java: Singleton, Factory, Builder and Prototype

14 Hours

Unit 4: OO Design Patterns & Anti-Patterns

Structural Patterns – Adapter, Façade, Proxy and Flyweight
Behavioural Patterns – Chain of Responsibilities, Command, Interpreter, Iterator
Anti-patterns – Introduction and classification, Project Management, Architecture and Development anti-patterns (1 anti-patterns of each type)

14 Hours

Hands-on/Assignment/Laboratory/Project

- Lab assignment on Use case diagram.
- Lab assignment on Class diagram.
- Lab assignment on Activity and State diagrams.
- Lab assignment on Java fundamentals.
- Lab assignment on Java Advanced features (Inheritance, Composition, etc.).
- Self Learning Assignment on Java Serialization and Multithreading
- Hands-on Assignment on MVC Framework.
- Assignment on Design Patterns.
- Mini Project using MVC Framework and incorporating all learning of the course.

Tools / Languages: Star UML, Java Programming Language

Text Book(s):

1: "Java the Complete Reference", Herbert Schildt ,McGraw-Hill ,11th Edition, 2018.
2: "Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development", by Craig Larman, 3rd Edition, Pearson 2015.

Reference Book(s):

1: "Object-Oriented Modelling and Design with UML", Michael R Blaha and James R Rumbaugh, 2nd Edition, Pearson 2007.
2: "Design Patterns: Elements of Reusable Object-Oriented Software" by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, 1st Edition, Pearson 2015.

UE21CS341B : Compiler Design (4-0-0-4-4)

Language design and implementation is an active topic in programming, and will likely always be. How we program and the tools we use, changes constantly. We try new ideas and come up with better or alternative approaches frequently. Any language that doesn't continue to adapt will fall into disuse, and any tool chain that remains stagnant will be forgotten. Hence knowledge of compilers in order to tweak these changes in the language design is a must for a Computer Science Engineer. This course requires the student to have a desirable knowledge of Data Structures and its Applications and Automata Formal Languages and Logic.

Course Objectives:

- Introduce the major concept areas of language translation and compiler design.
- Develop a greater understanding of the issues involved in programming language design and implementation.
- Provide practical programming skills necessary for constructing a compiler. Develop an awareness of the function and complexity of modern compilers.
- Provide an understanding on the importance and techniques of optimizing a code from compiler's perspective.

Course Outcomes:

At the end of this course, the student will be able to:

- Use the knowledge of patterns, tokens and regex for solving the problems in the field of data mining
- Analyze and design the semantic behaviour of a compiler. Choose the appropriate compiler internal representation for different kinds of compiler tasks.
- Translate a source-level language into a low-level compiler internal representation.
- Optimize the performance of a program in terms of speed and space using new code optimization techniques.

Desirable Knowledge: UE21CS252A- Data Structures and its Application, UE21CS243A- Automata Formal Languages & Logic.

Course Content:

Unit 1: Compilers : Introduction, Lexical Analysis, Top Down Parsers

The Language Processing System, The Phases of a Compiler, The Grouping of Phases into passes.
Lexical Analysis: The Role of the Lexical Analyzer, Input Buffering, Specification of Tokens, Recognition of Tokens, Design of a Lexical Analyzer Generator. The role of the parser, Syntax Error Handling, Error-Recovery Strategies. Top-down parsing: Recursive Descent Parser (RDP) with Backtracking, LL(1) Parser.

14 Hours

Unit 2: Syntax Analysis : Bottom up Parsers, Syntax-Directed Translation

Bottom-up parsing: Shift-Reduce Parsing, LR (0), SLR, viable prefixes, CLR, LALR. Syntax-directed definitions, Evaluation orders for SDD's : S-attributed SDD, L-attributed SDD, Applications of Syntax-Directed Translation - SDD for Syntax Trees, Expressions, Basic Types and flow control statements

14 Hours

Unit 3: Implementation of Syntax-Directed Translation Schemes and Intermediate Code Generation

Syntax-directed Translation Schemes – Parser Stack Implementation of Postfix SDT's, SDT's with actions inside Productions, SDT's for L-Attributed Definitions. Implementing L-Attributed SDD's: Bottom-Up Parsing. Variants of Syntax Trees – Directed Acyclic Graphs for Expressions, Three-Address Code – Addresses and Instructions, Quadruples, Triples, Indirect Triples, SSA Form, Control Flow Graph.

12 Hours

Unit 4: Machine Independent Code Optimization, Code Generation and Run Time Environment

Machine Independent Optimization: Different Optimizations, Optimization of Basic Blocks. Data Flow Analysis: Live-variable analysis, Next-use algorithm. Storage Organization, Different Allocation Strategies, Stack Allocation of space, Access to Non local Data on the stack. Code Generation: Issues in the design of a code generator, the target language, addresses in the target code, static allocation, stack allocation, run-time addresses for names. A Simple Code generator - The Code generation algorithm.

16 Hours

Tools/Languages: Lex and Yaac.

Text Book(s):

1: “Compilers–Principles, Techniques and Tools”, Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffery D. Ullman, 2nd Edition, Pearson Education, 2009.

Reference Book(s):

1: “Modern Compiler Design”, Dick Grune, Kees van Reeuwijk, Henri E. Bal,Ceriel J.H. Jacobs, Koen Langendoen, 2nd Edition, 2012.

UE21CS342BA1: Generic Programming in C++ (4-0-0-4-4)

This intensive course explores advanced C++ programming techniques with a focus on generic programming and the latest C++20 standard. Students will gain a deep understanding of modern C++ concepts while working on practical, hands-on projects.

Course Objectives:

- Develop a strong foundation in C++ and generic programming concepts.
- Master advanced C++ techniques and the latest language features from C++20.
- Learn performance optimization and interfacing with external systems.
- Gain experience in real-world applications such as networking and game development.
- Apply advanced C++20 features and compile-time metaprogramming to practical projects.

Course Outcomes:

At the end of this course, the student will be able to:

- Ability to design and implement generic data structures and algorithms using C++ templates.
- Proficiency in applying advanced C++ features and techniques to real-world problems.
- Knowledge of performance optimization and memory management in C++ applications.
- Skills to interface C++ applications with external systems, such as web services and databases.
- Expertise in utilizing C++20 features and compile-time meta-programming for complex projects.

Desirable Knowledge: UE21CS151A – Python for Computational Problem Solving, UE21CS151B- Problem solving with C, UE21CS252A- Data Structures and its Applications, UE21CS252B- Computer Networks, UE21CS241B- Design and Analysis of Algorithms.

Course Content:

Unit 1: C++ Foundations and Advanced Language Features

C++ history, features, and standards (including C++20). Basic syntax, variable types, operators, and control structures.Functions, function overloading, and recursion. Object-Oriented Programming in C++. Classes, objects, and member functions. Constructors, destructors, and the Rule of Three/Five. Inheritance, polymorphism, and encapsulation. Generic Programming and Templates. Function templates and class templates. Specialization and partial specialization. Template argument deduction and type traits. Advanced C++20 Language Features. Concepts and requires expressions in-depth, Ranges, projections, and custom range adaptors. New standard attributes and language features. Introduction to Networking with C++, fundamentals: sockets, protocols, and services.

14 Hours

Unit 2 : Advanced C++ Techniques, Concepts and Game Development

Advanced Template Features, Variadic templates and fold expressions. Template metaprogramming and compile-time computation. Expression SFINAE and type constraints (concepts). C++20 Ranges and Concepts. Ranges overview, views, and adaptors. Concepts: definition, usage. Implementing a custom range and using concepts for constraints. Game Development with C++. Game development concepts: game loop, input handling, and rendering. Game engines and libraries: Unreal Engine. Simple game development project in C++.

14 Hours

Unit 3: Performance and Optimization in C++ and High-Performance Computing

Memory Management and Optimizations. Memory layout and the stack/heap. Memory allocation, deallocation, and optimization techniques. Cache-aware and cache-oblivious programming. Multithreading and Concurrency. Thread creation, synchronization, and data sharing. Mutexes, locks, condition variables, and atomics. Profiling and Debugging Tools. Performance profiling and benchmarking tools. Debugging tools and static analyzers. SIMD and vectorization in C++. GPGPU programming with CUDA or SYCL.

14 Hours

Unit 4 : Interfacing with External Systems and Advanced Networking

Interfacing with C and System Libraries. C and C++ interoperability, extern "C". Using system libraries and platform-specific features. Dynamic loading of libraries. Interfacing with Python. Python-C++ bindings: pybind11,

Boost.Python. Calling Python from C++ and vice versa. Creating Python modules and packages in C++. Interfacing with Web Services. HTTP clients and servers in C++. RESTful API design and implementation. JSON parsing and serialization in C+Advanced Networking with C++. Asynchronous I/O with boost.asio or standalone_asio. Implementing network protocols and applications.

14 Hours

Tools / Languages: Compiler Explorer (godbolt.org), Cpp Insights (cppinsights.io), Quick-bench (quick-bench.com), wandbox (wandbox.org), Visual Studio Code (code.visualstudio.com), Clang-Tidy, Clang-Format, Cppcheck, Valgrind, Conan (conan.io)

Text Book(s):

1: “Programming: Principles and Practice Using C++”, Stroustrup, Bjarne, Addison-Wesley Professional, 2014.
2: “C++17 - The Complete Guide”, Josuttis, Nicolai M., and Sutter, Herb..Leanpub, 2018. ISBN: 978-1983321745.

Reference Book(s):

1: “C++ Templates: The Complete Guide”, Vandevoorde, David, and Nicolai M. Josuttis.. Addison-Wesley Professional, 2017.

UE21CS342BA2: Algorithms for Information Retrieval and Intelligence Web(4-0-0-4-4)

This course covers the basic and advanced algorithms and techniques for Information retrieval and web applications. This course focuses on Index building, document ranking, use of machine learning in Information retrieval , recommendation algorithms and design of intelligent web applications. This course requires the student to have a desirable knowledge of Design and Analysis of Algorithms.

Course Objectives:

- Understand the architecture, models and algorithms used in Information Retrieval.
- Understand the basic principles and implementation of Indexing and Search.
- Understand the use of machine learning in Information Retrieval and Web Applications.
- Understand multimedia and multimodal information retrieval.
- Understand use of neural networks for information retrieval.

Course Outcomes:

At the end of this course, the student will be able to:

- Implement an efficient index for a document collection.
- Perform searches on a document collection, rank and evaluate results.
- Apply suitable techniques for multimodal information retrieval
- Apply Machine Learning techniques in Information Retrieval Systems and Web Applications.

Desirable Knowledge: UE21CS241B - Design and Analysis of Algorithms, UE21MA241B- Linear Algebra, UE21CS352A- Machine Intelligence.

Course Content:

Unit 1: Basics of Information Retrieval

Introduction to Information Retrieval Background, Architecture and Strategies of Information Retrieval (IR) Systems, IR Models, Boolean and Extended Boolean Models, Dictionary, Vocabulary, Positional Postings, Phrase Queries and Tolerant Retrieval, Indexing and Vector Space Model, Evaluation of IR Algorithms for Indexing and Index Compression, Vector Space Model for Scoring, tf-idf and Variants

14 Hours

Unit 2: Ranking, Web Search Basics

Efficient Scoring and Ranking, Parametric and Zone Indexes, Tiered Indexes, Query Term Proximity, Query Parser, Aggregating Scores, Performance Measurement, Web Applications and Search Algorithms, Relevance Feedback, Query Expansion, Other IR Models, Web Search Basics, Economic Model of Web Search, Improving Search Results, Link Analysis, The Page Rank Algorithm, Other Search Algorithms, Scalability Issues in Search. Search User Experience

14 Hours

Unit 3: Link Analysis, Multimodal Information Retrieval

Web Crawling and Indices, Link Analysis, Building a Complete Search System, Lucene as a Search Engine, Other search engines like Solr, Everything, Google. Multimodal Information Retrieval content-based visual information retrieval: meta data searching, Query by Example, Semantic Retrieval, Machine Learning Approaches, content comparison using image distance measures. Multimedia and Multimodal Information Retrieval: Introduction, Requirements, Applications, challenges, Architecture, Metadata, Techniques for content Processing

14 Hours

Unit 4: Question Answering, neural models for IR

Question Answering, Neural models for IR QA as an Information Retrieval task, Factoid QA models, Entity Linking models, Knowledge based QA, Pretrained models for QA. Neural models for IR: Research Papers (1. Neural Models for Information Retrieval, Bhaskar Mitra, Nick Craswell, 2017. 2. Neural ranking models for document retrieval, Mohamed Trabelsi, Zhiyu Chen, Brian D. Davison, Jef Hefin, 2021)

14 Hours

Tools/Languages: Scikit, Tensor flow, Solr, Lucene search engines / Python Programming Language.

Text Book(s):

1: “Introduction to Information Retrieval”, Christopher D. Manning, Prabhakar Raghavan, Hinrich Schutze, ISBN: 9781107666399, Cambridge University Press, 2009.
2. “Search Computing”, Challenges and Directions: chapter 8: Multimedia and Multimodal Information Retrieval.
3. “Speech and Language Processing”, Third Edition, Daniel Jurafsky, James H. Martin, Chapter 14: Question Answering and Information Retrieval, 2023.

Reference Book(s):

1: “Algorithms of the Intelligent Web”, Haralambos Marmanis, Dmitry Babenko,Manning Publishers, 2011.

UE21CS342BA3 : Image Processing and Computer Vision I (4-0-0-4-4)

Digital Image Processing deals with processing images that are digital in nature. Improving the quality of images for human perception and understanding, extracting useful information for decision making and efficient storage are some of the driving factors behind image processing techniques/algorithms. The course on Digital Image Processing introduces the learner to various image processing techniques, algorithms and their applications.

Course Objectives:

- Understand the principles underlying imaging
- Assess the quality of an image based on its visual content, histogram and statistical measures
- Gain an insight to image enhancement techniques in the spatial and frequency domains
- Gain insight into various morphological operations and segmentation techniques
- Learn the nuances of various color spaces and image compression techniques
- Learn different feature extraction techniques for classification

Course Outcomes:

At the end of this course, the student will be able to:

- Apply image fundamentals for various applications
- Assess the quality of an image and identify the appropriate processing technique that needs to be applied
- Apply image enhancement techniques in both the spatial and frequency (Fourier) domains
- Design and evaluate methodologies for image segmentation
- Use appropriate feature extraction techniques for tasks such as classification
- Use both feature-engineering based approaches and data-drive approaches for classification

Desirable Knowledge: UE21CS241B – Design and Analysis of Algorithms.

Course Content:

Unit 1: Introduction and Enhancement in the Spatial Domain

What is Digital Image Processing, examples of fields that use DIP, Fundamental Steps in Digital Image Processing, elements of visual Perception, Basic Concepts in Sampling and Quantization, Representing Digital Images, Spatial and Gray-level Resolution, Zooming and Shrinking Digital Images, Some Basic Relationships Between Pixels, Linear and Nonlinear Operations. Image Enhancement in the Spatial Domain: Some Basic Gray-level Transformations, Histogram Processing, Enhancement Using Arithmetic/Logic Operations, Basics of Spatial Filtering, Smoothing Spatial Filters, Sharpening Spatial Filters, and Combining Spatial Enhancement Methods.

14 Hours

Unit 2: Image Enhancement in the Frequency Domain, Transforms and Compression

Image Enhancement in the Frequency Domain: Introduction to the Fourier Transform and the Frequency Domain, Smoothing Frequency-Domain Filters, Sharpening Frequency Domain Filters, Homomorphic filtering. Image transforms (DCT, KLT, DWT) for applications. Image Compression: Fundamentals - Image Compression Models, Some encoding techniques (including, block processing and transform-based techniques).

14 Hours

Unit 3: Morphological Image Processing, Preliminaries of segmentation and Color Image Processing

Dilation and Erosion, Opening and Closing, the Hit-or-Miss Transformation, Some Basic Morphological Algorithms. Image Segmentation: Detection of Discontinuities, Edge Linking and Boundary Detection, Thresholding, Region-Based Segmentation. Color Models, Pseudocolor Image Processing, Basics of Full-Color Image Processing, Color Transformations, Smoothing and Sharpening, Color Segmentation, Noise in Color Images.

14 Hours

Unit 4: Feature extraction, pattern classification

Scale Image Feature Transform, Image pattern classification using minimum distance classifier and prototype matching, deep convolutional neural networks and some of its recent variations for computer vision applications and state-of-art techniques and applications in computer vision.

14 Hours

Tools / Languages: Matlab, Python Programming Language.

Text Book(s):

1: “Digital Image Processing”, Rafael C Gonzalez and Richard E. Woods, Prentice Hall, 4th Edition, 2018.

Reference Book(s):

1: “Digital Image Processing and Analysis”, Scott E. Umbaugh, CRC Press, 2014.
2: “Digital Image Processing”, S. Jayaraman, S. Esakkirajan, T. Veerakumar, McGraw Hill Ed. (India) Pvt. Ltd., 2013.
3: “Digital Signal and Image Processing”, John Wiley, 2003.
4. “Computer Vision A Modern Approach”, D. A Forsyth and J. Ponce, Pearson Education, 2003.
5. “Computer Vision: Algorithms and Applications”, Richard Szeliski, Springer, 2nd Edition, 2022.

UE21CS342BA4: Natural Language Processing (4-0-0-4-4)

The goal of this course is to focus on processing of text data as found in natural language usage. The key problem discussed in this course is that of understanding the meaning of text by various types of learning models including the recent approaches using deep learning and the significance of the NLP pipeline in that meaning disambiguation process. The course also discusses disambiguation of syntax as a step of meaning disambiguation process. This course requires the student to have a desirable knowledge of Machine Intelligence.

Course Objectives:

- Learn the phases of NLP and the building of Language Models.
- Learn how lexical and distributional semantics can be used for semantic disambiguation in NLP.
- Focus on various learning models related to sequence labelling that is the basic building block in NLP.
- Learn how syntactic disambiguation is done in NLP.
- Introduce the deep learning techniques and its applications in Natural Language Processing.

Course Outcomes:

At the end of this course, the student will be able to:

- Have a very clear understanding of the phases of NLP and the building of Language Models.
- Analyze and apply comfortably appropriate branch of semantics depending on the problem being solved.
- Understand various sequence labelling approaches and applications in NLP.
- Understand how syntactic ambiguity removal can contribute in overall disambiguation process.
- Design and implement neural language model, NLP applications using neural techniques and utilize various transfer learning approaches in NLP.

Desirable Knowledge: UE21CS352A-Machine Intelligence.

Course Content:

Unit 1:Introduction to NLP, Words, Corpora

Introduction, Types of ambiguity in natural language processing, Text normalization,Morphological parsing of words – Porter stemmer, Lemmatization and Stemming, Sentence segmentation. **Noisy Channel model:** Real world spelling error, Minimum edit distance algorithm, Concept of noisy Channel Model. **Language Model**:N-grams,n-gram language model, smoothing, discounting and back-off, Kneser-Ney smoothing, interpolation, perplexity as an evaluation measure .

14 hours

Unit 2: Lexical and Vector Semantics

Word senses and relations between word senses, WordNet: A Database of Lexical Relations; Word sense disambiguation , Semantic relatedness ,Lexicons for sentiment and affect extraction: available sentiment and emotion lexicons. Vector Semantics and Embeddings: Words and vectors, TF IDF, Pointwise Mutual Information, Measuring similarity, Using syntax to define a word's context, Evaluating vector models, Dense vectors via SVD Distributional Hypothesis, **Neural Embedding:** skip gram and CBOW Pre-trained word representations: Word2Vec , Improving Word2vec, Limitation of distributional methods.

Self-Learning component:FastText, Glove

14 Hours

Unit 3: Handling sequences of text and Parsing - Disambiguating Structure

Sequence labelling: Viterbi algorithm and Hidden Markov Model, POS Tagging example, Discriminative Sequence labelling with features-Conditional Random Field. Other sequence labelling applications – Named Entities and Named Entity Tagging. Sequence labelling using RNNs and LSTMs

Self-Learning component:POS Tagging using discriminative models i.e. Maximum Entropy Markov Model (MEMM)

Constituency parsing: Ambiguity presented by parse trees, CKY parsing, Span-based Neural Constituency Parsing, CCG Parsing, Partial parsing – chunking. **Statistical Parsing :** Probabilistic Context Free Grammar, Probabilistic CKY parsing of PCFG, Problems with PCFG, Probabilistic Lexicalized CFG **Introduction to dependency parsing:** Dependency relations, Dependency Formalisms, Dependency Tree Banks. Evaluating parsers.

14 Hours

Unit 4: Coreference resolution,Transformers and Pretrained Language Models

Coreference resolution: Forms of referring expression, algorithms for coreference resolution – mention pair and mention ranking model, mention detection, classifiers using hand-built features.

Self-Attention Networks: Transformers, Transformers as Language Models, Sampling, Pretraining Large Language models, Fine-Tuning and Masked Language Models: Bidirectional Transformer Encoders, Training Bidirectional Encoders, Transfer Learning Through Fine-tuning, Training Corpora, BERT ,RoBERTa. ChatGPT and its Architecture.

14 Hours

Tools / Languages : Tensorflow, Scikit Learn, Python 3.x. CoreNLP, Natural Language Toolkit (NLTK), TextBlob,Gensim, SpaCy,PyTorch-NLP , OpenNLP.

Text Book:

1. "Speech and Natural Language Processing", Daniel Jurafsky and James H. Martin, 3rd edition online,2023. The more up to date 3rd edition draft is available at http://web.stanford.edu/~jurafsky/slp3/.

Reference Book(s):

1. "Introduction to Natural Language Processing", Jacob Eisenstein, MIT Press, Adaptive computation and Machine Learning series, 18th October, 2019.
2. The open source softcopy is available at githubhttps://github.com/jacobeisenstein/gt-nlp class/blob/master/notes/eisenstein-nlp-notes.pdf.

UE21CS342BA5 : Blockchain (4-0-0-4-4)

Blockchain having wide impact and potential growth for change around the world. It is changing how business is executed. It is important to understand why Blockchain is different and how it works in comparison with technologies of the past. This course requires the student to have a desirable knowledge of Data Structures and its Applications.

Course Objectives:

- Learn a conceptual view of Blockchain for the new applications that they enable.
- Apply the Blockchain for various applications to provide a secure way of data access using cryptographic functions.
- Learn various consensus mechanisms to implement for various real time applications.
- Familiarize with the Blockchain deployment tools.
- Learn various vulnerabilities and security mechanisms of Blockchain.

Course Outcomes:

At the end of this course, the student will be able to,

- Analyze how the traditional databases can be replaced with Blockchain for the real time applications.
- Integrate various cryptographic algorithms in to Blockchain.
- Apply various consensus mechanisms to the real world Blockchain applications.
- Evaluate the setting where a Blockchain based structure may be applied, its potential and its limitation.
- Identify the threats of Blockchain and deploy security mechanisms.

Desirable Knowledge: UE21CS252A-Data Structures and its Applications.

Course Contents:

Unit 1: Introduction and Data flow and Integrity

Key Blockchain Concepts, Nodes, Cryptocurrency, tokens, Public Ledger, Peer to peer Network, Types of Blockchain, Permissioned Blockchain model, Permission-less Blockchain model, Demonstration of Blockchain Construction steps, Demonstration of ether scan and Goerli/Sepoliaetherscan**.**

Cryptography- Need, history, features, Private and public keys, Types of cryptography, Digital signatures, Hash functions, SHA-256, Patterns of hashing, Hash Pointer, Markle tree, Ledgers, Transactions and trade, the public witness, Computers that witness.

14 Hours

Unit2 : The structure of the network: consensus algorithm

Case study: Bitcoin Blockchain Network, Creation of metamask wallets and performing transaction.

Introduction to distributed consensus: What, why, Challenges, Proof of Work, Proof of Stake, Delegated Proof of Stake, Proof of Authority, Proof of Elapsed Time, Proof of Scope, Proof of Space, Proof of Burn, RAFT, PAXOS, Byzantine Fault Tolerance System, PBFT.

14 Hours

Unit 3: Second generation applications of Blockchain technology

Smart contracts: origins and how they function, Creating and deploying smart contracts, Decentralized applications, Dapps construction, Decentralized Autonomous Organizations (DAOs)- Need, Principal agent dilemma, components, The DAO Story, Legality of DAPPs and DAOs.

Solidity- Variable, Functions, modifiers, view, pure, fallback, overloading, in-built mathematical and cryptography functions, Withdrawal pattern, Restricted Access Hyperledger Fabric: Blockchain-as-a-service (BaaS), Architecture and core components, Hyperledger fabric model, Creation of a simple DAPP.

14 Hours

Unit 4: Blockchain Security and use cases

Blockchain vulnerabilities, Smart contract vulnerabilities, Blockchain on CIA security triad, Blockchain based DNS security platform, deploying blockchain based DDOS protection.
Use cases: Public Sector, Finance, Supply Chain. Research Aspects in Blockchain

14 Hours

Tools / Languages: Solidity, Remix, Ganache, Metamask.

Text Book(s):

1: Introduction to Blockchain Technology, Tiana Laurence, 1st edition, Van Haren Publishing, 2019. 2. 2. Blockchain Technology From Theory to Practice, Sudeep Tanwar, 1st edition, Springer, 2022.

Reference Book(s):

1. Hands-On Cyber security with Blockchain: Implement DDoS protection, PKI-based identity, 2FA, and DNS security using Blockchain, Rajneesh Gupta, 1st edition, 2018.
2: Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction, Narayanan, Bonneau, Felten, Miller and Goldfeder, Princeton University Press, 2016.

UE21CS342BA6: Digital Forensics and Incident Response (4-0-0-4-4)

Digital forensics is the core set of principles and processes necessary to produce usable digital evidence and uncover critical intelligence. Digital Forensics course provides a deep understanding of the techniques to gather, protect and report the digital confirmations

Course Objectives:

- Introduces the history and evolution of Digital forensics, principles of Digital Forensics and its challenges, Incidence response types, Industry standards.
- Provide an understanding of the various sources of evidence and the procedures for acquiring the evidence
- Learn about Computer memory, Storage devices, file structures and file forensics and practice with applicable tools.
- Focus on Network Forensics, Browser forensics and mobile device forensics and Learn about the email investigations and anti-forensics
- Learn on the procedure for final report writing as per the court of law

Course Outcomes:
At the end of the course, the students will be able to

- Understand the Principles of Digital Forensics, Evolution of Computer Forensics, Scope of Cyber Forensics Investigation process and Incident response
- Know the various sources of Digital evidence and acquire the evidence by applying the standard acquisition procedures
- Understand the various operating system file systems and able to perform a better forensic analysis of the same
- Detect network intrusions and unauthorized access, Perform e-mail investigations such as e-mail crimes with help of e-mail server logs and Perform procedures for acquiring data from mobile devices using appropriate tools
- Write professional reports for Legal consumption.

Course Content:

Unit 1 : Introduction to Digital Forensics and Incident Response With Essential Technical Concepts
Introduction to forensics scienceand digital forensics,understanding digital forensics,Locard's exchange principle, Scientific method, Organization of note , International standards and practices, Role of examiner in judicial system, Digital forensic goals, Digital forensics categories and Challenges,,Digital forensics investigation types, Forensics readiness, Digital evidences - Evidence types,Location of electronic evidences, Challenge of acquiring digital evidence,Digital forensics examination process(life cycle), Digital forensics Vs other computing domain
Incident Response: Attack Lifecycle, What is Incident Response?, Incident Response Team, Incident Response Plan (IRP), Incident Response Lifecycle, Incident Response Methodology, Goals of Incident Response, Incident Response Tools

Essential Technical Concepts: Data representation, File structures, Memory types, types of computer storage, understanding the hard disks-HDD,SSD, How magnetic hard drive store data,, Data recovery considerations and file systems- introduction Allocated and unallocated space, Data Persistence, Page file(Swap space)

14 Hours

Unit 2 : Acquiring-Analyzing Digital Evidence, and Filesystems - Windows Systems
Acquiring-Analyzing Digital Evidence
Initial Response and First Responder Tasks- Search and Seizure, First responder tool kit and tasks, Documenting digital crime scene, Packing and transporting digital evidences, Conducting interview,Acquiring digital evidence: Acquiring volatile memory(Live acquisition- Tools), Acquiring Non-volatile Memory (Static Acquisition-Tools), Analysing Digital Evidence, autopsy data source types, how to recover deleted files from the supplied data source
Filesystems - Windows file systems- File Systems- examining FAT, NTFS, (HFS+), File allocation table(FAT), New Technology File System(NTFS), windows system artifacts, Introduction Deleted Data Hibernation File

(Hiberfile.Sys) Registry Print Spooling Recycle Bin Metadata Thumbnail Cache Most Recently Used (MRU) Restore Points and Shadow Copy, Windows forensic analysis- Timeline Analysis, file recovery, Windows registry analysis, Deleted registry key recovery.

14 Hours

Unit 3 : Filesystems - Mac OS X Systems and Linux Systems , Network Forensics and Web browser Forensics

Linux Systems and Artifacts- Linux file systems (Ext2/Ext3), file system layer, filename layer, metadata layer, data unit layer, deleted data, Linux logical volume manager, Linux boot process and services, Linux System Organization and Artifacts, Unix/Linux Forensic Investigation: Unix/Linux forensics, investigation steps and technologies, Principles of file carving **Mac OS X Systems and Artifacts**- OS X File System Artifacts- HFS+ Structures, OS X System Artifacts. **Network Forensics**: Networking Fundamentals and Types of Networks, Network Forensics Overview, Securing A Network,Developing Procedures for Network Forensics Network Security Tools ,Network Attacks,Forensic Footprints,Network Evidence and Investigations, Seizure of Networking Devices,**Applying Forensic Science To Networks**

Web browser forensics- Internet overview, IE, Microsoft web browser, Firefox, Google Chrome, web browser investigation tools.

14Hours

Unit 4: E-Mail forensics, Mobile Device Forensics,Anti-forensicsand Report Writing

Email Forensics- Steps in email communications, List of E-mail protocols, E-Mail header examination, email forensics, recovering emails

Mobile device forensics- Cellular Networks and How They Work, Overview of Cell Phone Operating Systems, Potential Evidence Found on Cell Phones, Collecting and Handling Cell Phones as evidence, Cell Phone Forensic Tools, mobile device forensic investigation, storage location, acquisition methods

Anti-forensics: Introduction to Antiforensics, Classification of antiforensics techniques,antiforensics Practices-Data Wiping and Shredding, Trail Obfuscation, Encryption, Data Hiding, stenography techniques and tools, Introduction to malware analysis and malware forensics

Report Writing: Prep work for report Writing, Structure of the report, Characteristics of a good report, Document design and good writing practices, Legal Acceptance , digital forensics report writing for High-Tech Investigations

14 Hours

Tools / Languages :Open source tools on Forensics.

Text Book(s):

1: “The basics of digital forensics: the primer for getting started in digital forensics”, Sammons, J. Elsevier, 2012.
2: “Digital Forensics Basics: A Practical Guide Using Windows OS”, Hassan, N. A. Apress 2019.
3: “Practical Cyber Forensics- An Incident-Based Approach to Forensic Investigations”, by Niranjan Reddy, A Press, 2019.
4: “Introductory Computer Forensics-A Hands-on practical Approach”, by Xiaodong Lin, Springer, 2018.

Reference Book(s):

1: “Guide to Computer Forensics and Investigations”, Bill Nelson, Amelia Phillips and Christopher Steuart, Course Technology, Cengage Learning, 2010.
2: “Digital forensics with open source tools”, Altheide, C., & Carvey, H. Elsevier, 2011.
3: “Digital Forensics Workbook_-Hands-on Activities in Digital Forensics”, Michael K Robinson ,CreateSpace Independent Publishing Platform, 2015.

UE21CS342BA7: Digital Twin and eXtended Reality (4-0-0-4-4)

This course is an in-depth exploration of the concepts, technologies, and applications of DigitalTwin and Extended Reality. Through a combination of theoretical lectures and practical projects,students will gain the knowledge and skills needed to develop intelligent systems, analyzesecurity threats, and apply Digital Twin and Extended Reality solutions to various industries.

Course Objectives:

- Understand the fundamental concepts and technologies of Digital Twin and Extended Reality, including AR and VR, 3D transformation, AI/ML, and data visualization.
- Develop the skills needed to design and implement Digital Twin and Extended Reality solutions, including the ability to create mock, functional, and executable twins.
- Analyze security threats to Digital Twin and Extended Reality systems, and apply trust modeling techniques to ensure the security and integrity of these systems.
- Apply Digital Twin and Extended Reality solutions to various industries, including industrial IoT, critical infrastructure, Agri infrastructure, and connected vehicles.
- Collaborate with peers on practical projects to gain hands-on experience in developing intelligent systems and applying Digital Twin and Extended Reality solutions to real-world problems.

Course Outcomes:

At the end of this course, the student will be able to:

- Students will be able to demonstrate a deep understanding of the fundamental concepts and technologies of Digital Twin and Extended Reality, including AR and VR, 3D transformation, AI/ML, and data visualization.
- Students will be able to design and implement Digital Twin and Extended Reality solutions, including the ability to create mock, functional, and executable twins, and apply generative AR techniques.
- Students will be able to analyze security threats to Digital Twin and Extended Reality systems, and apply trust modeling techniques to ensure the security and integrity of these systems.
- Students will be able to apply Digital Twin and Extended Reality solutions to various industries, including industrial IoT, critical infrastructure, Agri infrastructure, and connected vehicles, and assess the benefits and limitations of these solutions.
- Students will be able to collaborate effectively with peers on practical projects, and demonstrate the ability to apply Digital Twin and Extended Reality solutions to real-world problems, such as improving operational efficiency, enhancing safety, and reducing costs.

Desirible Knowledge : UE21CS252A- Data Structures and its Applications.

Course Content:

Unit 1:Introduction to AR and VR

Overview of AR and VR technologies, Introduction to the OpenGL graphics pipeline, 3D transformation and Quaternions math, Basic coding in OpenGL, Developing a simple AR or VR applications.

14 Hours

Unit 2: Digital Twin Essentials

Introduction to Digital Twin concepts, Types of Digital Twin, Metaverse and Metaversity, Examples of mock, functional, and executable twins, Developing a Digital Twin using a simulation platform.

14 Hours

Unit 3: Intelligent Systems for IIoT.
Methods and models for developing intelligent systems, AI/ML techniques for IIoT, Generative AR and data visualization, Analytics for IIoT, implementing an intelligent system using a chosen AI/ML framework.

14 Hours

Unit 4: Security and application studies
Security aspects of Digital Twin, Trust modelling for Digital Twin, Overview of XR-based attacks, Conducting a security analysis of a Digital Twin, BlockChain for DT, Digital twin in Brain Computing Interface, Digital Twin in Digital Transformation, Application of Digital Twin and Extended Reality in various industries, Case studies in industrial IoT, critical infrastructure, Agri infrastructure, connected vehicles.

14 Hours

Tools/ Languages: C/ C++/ JAVA/ Python using OpenGL.

Text Book(s):
1: "Interactive Computer Graphics - A top-down approach with shader-based OpenGL", Edward Angel and Dave Shreiner, Pearson Education, Sixth edition, 2012.
2: Digital Twin a Complete Guide for the complete Beginner by Santanu deb barma& Vijay Raghunathan

Reference Book(s):
1: Building IndustrialDigital Twins by Shyam Varan Nath & Pieter van Schalkwyk, by Packt Publishing Ltd.

UE21CS343BB1: Heterogeneous Parallelism (4-0-0-4-4)

This course focuses on parallel heterogeneous architectures as well as programming models and imparts pragmatic skills to program using parallel programming languages and frameworks trending in the industry.

Course Objectives:

- Familiarize with various parallel heterogeneous architectures, associated techniques and programming models.
- Acquaint with Memory Consistency & Coherence.
- Acquaint with Concurrency Bugs and Resolution Techniques.
- Familiarize with opportunities and challenges in Parallel Programming using popular frameworks.
- Familiarize with opportunities and challenges in Parallel Programming using popular languages.

Course Outcomes:

At the end of this course, the student will be able to:

- Understand the underpinnings of Parallel Heterogeneous Architectures and Parallel Computing Techniques.
- Understand Memory Consistency Models and Coherence Techniques.
- Understand Techniques for Parallelism Bugs' Resolution.
- Program using popular Parallel Programming Frameworks and Languages trending in the industry.
- Engineer high performance migration of varied applications appreciating and assimilating varieties of parallelism.

Desirable Knowledge: UE21CS151B – Problem Solving with C, UE21CS251B – Microprocessor and Computer Architecture.

Course Content:

Unit 1: Fine Grained Parallelism

Review on Parallelism and Performance, Instruction Level Parallelism and Enhancement Techniques, Prediction and Speculation, Vectorization and Predication, Dependency Analysis, Code Optimization, Cache optimized Programming. Laws of Parallelism, Data, Task and Pipeline Parallelism, Pthreads.

14 Hours

Unit 2: Coarse Grained Parallelism & GPU Architecture & Programming

Multithreaded and Multi-Core architectures, GPUs and GPGPUs, GPU Architectures, CUDA Programming Frameworks, OpenCL, Many-Core Heterogeneous Architectures.

14 Hours

Unit 3: Parallelism Frameworks

OpenMP, Race Conditions, Deadlocks & Debugging, Memory Models for Parallel Programming, Memory Consistency Models.

14 Hours

Unit 4: Parallel Algorithms & Programming Languages

Parallel Algorithms, Task Decompositions and Mapping. Hardware acceleration platforms (FPGAs and TPUs), Architecture and organization of modern FPGAs, Case Studies, Concurrency in Main stream Languages.

14 Hours

Tools/Languages: pthread, OpenMP, CUDA, openCL.

Text Book(s):

1: "Programming Massively Parallel Processors", David Kirk and Wen-mei Hwu, 3rd Ed, Morgan Kaufmann, 2016.

Reference Book(s):

1: "Computer Architecture: A Quantitative Approach: John Hennessy, David Patterson", 6th Edition, Morgan Kaufmann, 2017.

2: “Computer Systems: A Programmer's Perspective”, Randal E. Bryant, David R. O'Hallaron, Pearson, 2nd Ed, 2016.
3: “Parallel Programming for FPGAs”, Ryan Kastner, Janarbek Matai, and Stephen Neuendorffer, Creative Commons, 2018.
4: Latest Web Resources and Papers.

UE21CS343BB2: Topics in Deep Learning (4-0-0-4-4)

Deep Learning has received a lot of attention over the past few years and has been employed successfully by companies like Google, Microsoft, IBM, Facebook, Twitter etc. to solve a wide range of problems in Computer Vision and Natural Language Processing. In this course we will learn about the building blocks used in these Deep Learning based solutions. At the end of this course students would have knowledge of deep architectures used for solving various Vision and NLP tasks. This course requires the student to have a desirable knowledge of Machine Intelligence.

Course Objectives:

- Introduction to Deep Neural Network and programming with Tensor Flow and Keras tools.
- Introduction to Deep Learning(CNN, RNN)
- Introduction to Transfer Learning Techniques.
- Introduction to Generative Adversarial Networks and Graph Neural Network.
- Introduction to Deep Reinforcement Learning.

Course Outcomes:

At the end of this course, the student will be able to:

- Implement a Neural Network using Tensor Flow and Keras
- Classify images using CNN.
- Solve time-series related problems with RNN.
- Generate data in the form of images using GAN.
- Develop a simple game engine using Reinforcement Learning.

Desirable Knowledge: UE21CS352A : Machine Intelligence.

Unit 1: Introduction to Deep Learning: Introduction**,** Activation functions, Loss functions, Batch Normalization, Regularization and Optimization.**Convolutional Neural Network(CNN):** Introduction, Filters, FeatureMaps, Max-Pool Layers, Other Pooling Types, Back Propogation. Convolution Architectures - Alexnet, ZFNet, VGGNet, GoogleNet, ResNet. RCNN, FRCNN, Faster RCNN, YOLO V5. **Transfer Learning**: Introduction, Motivation, Variations, TL Architecture of CNNs. Hands-on: Assignment on CNN & TL.

14 Hours

Unit 2: Recurrent Neural Networks (RNN): Introduction-Recurrent Neurons, Memory Cells, Variable-Length Input-Output Sequences, RNN Architecture, Sequence learning problem, BPTT-Back Propagation Through Time, truncated BPTT, Vanishing and Exploding Gradient, Bidirectional RNN, LSTM Cell and GRU Cell, Text Classification with RNN, Encoder/Decoder architecture, Seq2Seq model with Attention , Transformer model and BERT architecture, Transformer Attention and its implementation.
Hands-on: Problem using RNNs, Transformers and TL / predefined model **14 Hours**

Unit 3: Generative Models & Meta Learning: Introduction to Autoencoders , Regularization in autoencoders, Denoising autoencoders, Sparse autoencoders, Contrastive autoencoders , Variational Auto Encoders(VAEs). **Generative Adversarial Networks**(GANs)- Architecture and Training Methods, Image Generation, DCGAN, Style GAN, WGAN, Applications.**Graphical Neural Networks(GNN)**: Introduction to GNNs, Graph Convolution Networks, Applications. **Meta Learning**: Introduction to Meta Learning, MAML, FOMAML, Adaptive Neural inductive Learning, Siamese Networks.
Hands-on: Problem on Fashion **MNIST** dataset for generating apparels.

14 Hours

Unit 4: Reinforcement Learning, **Diffusion Models, Federated Learning and Overview of Latest Deep Learning Models:**
Introduction, Basic Framework of RL, Learning to Optimize Rewards, Credit Assignment Problem, Temporal Difference, Learning and Q Learning. **Deep RL:** Deep Q Learning, Training and Testing.
Diffusion Model, Stable diffusion architectures, Introduction to Vision Transformers, GPT Architecture. **Federated Learning:** Horizantal, Vertical and FTL (Federated Transfer Learning).

Hands-on : Flower Architecture for MIMC3 dataset **OR** Learning to play a simple game using Double Deep (DD) Q-learning implementation on OpenAI gym.

14 Hours

Tools/ Languages: Pytorch.

Text Book(s):

1: “Advanced Deep Learning with Python” - Ivan Vasilev, Packt Publishing, 2019.
2: “Neural Network and Deep learning” by Charu C Agarwal. Springer International Publishing 2018.

Reference Book(s):

1: “Hands-on Machine Learning with Scikit-Learn and TensorFlow”, Aurelian Geron, O’REILLY, 1st Edition, 2017.
2: “Deep Learning with Keras”, Antonio Gulli and Sujit Pal, Packt Publishing, 1st Edition, 2017.
3: “Pattern Recognition and Machine Learning”, Christopher Bishop, Springer, 1st Edition, 2011 (Reprint).
4:Handouts: Transfer Learning / Latest Deep Learning Techniques / Vision Transformers/ GPT / FL

UE21CS343BB3: Database Technologies (4-0-0-4-4)

The last decade saw enormous advancements in the design of large-scale data processing systems due to the rise of Internet services. This course covers the architecture of modern data management systems. Topics include storage management, query optimization, distributed and parallel processing, data streaming with a focus on the key design ideas shared across many types of data intensive systems.

Course Objectives:
The objective(s) of this course is to make the student learn.

- Data Storage Techniques, Indexing Mechanisms.
- Query Processing and Query Optimization Techniques.
- Parallel and Distributed Databases.
- Data Streaming Systems.
- Current Trends in Design and Implementation of Database Systems.

Course Outcomes:
At the end of the course, the student will be able to:

- Design and Deploy Storage Solutions and Indexing mechanisms for optimized performance of databases.
- Perform Query Optimization.
- Apply Parallel and Distributed Database approaches to solve problems of large databases.
- Select an approach to manage "Data Streams".
- Apply techniques learnt to Develop or Select Specialized Database Applications.

Desirable Knowledge: UE21CS351A – Database Management Systems.

Course Content:
Unit 1: Relational Data Model and Storage Formats and Indexing
Review of Relational Design Theory – Functional Dependencies, Normalization. **Secondary Storage Management:**The Memory Hierarchy, Flash Storage and Database Buffer, Architecture, Accelerating Access to Secondary Storage, Disks and Disk Failures, RAID, Arranging Data on Disk, Block and Record Addresses, Variable-Length Data and Records – Column Stores, Record Modifications. **Index Structures**: Concepts, Types of Indexes, B+ Tree Indexes, Hash Tables, Multidimensional Data and Indexes and It's Applications, Multiple-key Indexes and R-trees, Bitmap Indexes, Indexes in SQL.

14 Hours

Unit 2: Query Processing and Optimization
Query Execution: Physical-Query-Plan Operators, One-Pass Algorithms and Two-Pass Algorithms (based on Sorting and Hashing), Clustered and Non-clustered indexes, Buffer Management. **The Query Compiler:** Parsing and Preprocessing, Algebraic Laws for Improving Query Plans, From Parse Trees to Logical Query Plans, Cost based Plan Selection, Choosing an Order for Joins.

14 Hours

Unit 3: Parallel and Distributed Databases
Avenues for Parallelism: Parallel Database Architectures, Models of Parallelism, Intra-query and Inter-query Parallelism, Parallel Algorithms on Relations. **Distributed Databases** – Distributed Data Storage, Types, Advantages, Data Fragmentation and Replication, Distributed Transactions, Distributed Query Processing, Distributed Commit – Two-phase Commit, Distributed Locking, Distributed Catalog Management, Concurrency Control, Peer-to-Peer Distributed Search – Chord Circles.

14 Hours

Unit 4: Data Stream Management &Trends inDesign and Implementation of Database Systems
Introducing **Stream Processing**, Stream Processing Model, Streaming Architectures, Apache Spark as a Stream Processing Engine, Spark's Distributed Processing Model, Spark's Resilience Model, Apache Kafka, Amazon Kinesis.

Current Trends in Design and Implementation of Database Systems and Decision Support Systems, Data Warehousing and Data Mining Applications, Introducing Data Lake, Data Mesh and Data Fabric, Multi-model Databases.

14 Hours

Tools & Languages: MySQL, postgres, Oracle, Apache Spark, Apache Kafka, Amazon Kinesis.

Text Book(s):

1: "Database Systems: The Complete Book", H Garcia-Molina, JD Ullman and J Widom, 2nd Ed., Pearson, 2018.
2: "Stream Processing with Apache Spark", Gerard Maas & Francois Garillot, O'Reilly, June 2019.

Reference Book(s):

1: "Fundamentals of Database Systems", Elmasri and Navathe, Pearson Education, 7th Ed., 2016.
2: "Streaming Systems" byTyler Akidau, Slava Chernyak, Reuven Lax, O'Reilly, July 2018.

UE21CS343BB4: Meta-Learning for Graph Abstractions (4-0-0-4-4)

`

UE21CS343BB4: Meta-Learning for Graph Abstractions (4-0-0-4-4)

This course introduces the students to the three meta landscapes of representation, optimization, and objective for the learning tasks on graph data. Graphs are ubiquitous in almost everything today –in social graphs, large-scale biological systems, road networks, polypharmacy, and user-item relations. Graphs, being non-Euclidean, require different machine-learning approaches. This course starts by exploring graph features for traditional machine learning on graphs and discussing a few noteworthy classical graph ML approaches. The second unit focuses on shallow embedding methods for graphs in general and then the same for knowledge graphs. The third unit focuses on representation learning using graph neural networks and discusses key algorithms along with extensions like temporal graphs and multi-relational graphs. The fourth unit discusses the deep generative models on the graph and model-building techniques using graph neural networks. The theory input concludes with a discussion on the interdisciplinary applications of Graph Neural Networks, interpretability, and meta-learning perspectives of graph AI.

Course Objectives:

- Learn graph feature-based traditional machine learning.
- Learn shallow embedding approaches for graphs and multi-relational knowledge graphs.
- Learn neural methods of representation learning on graphs.
- Learn GNN modeling techniques and deep graph generative models with interdisciplinary applications.

Course Outcomes:

At the end of this course, the student will be able to:

- Should be able to do feature engineering-based graph ML tasks.
- Should be able to do downstream machine learning tasks on graphs using shallow embedding.
- Should be able to solve a downstream machine-learning task using a graph neural network.
- Should be able to use graph modeling techniques and appreciate deep generative models on graphs.

Desired Knowledge:UE21CS352A – Machine Intelligence, UE21CS343AB3 -Graph Theory and its applications

Course Content:

Unit 1:Traditionalmachine learning on graph

Types of complex Graphs and computational tasks on Graphs. Graph Features for traditional machine learning– Node level features. Graph kernel features- Graphlet & Motif, Weisfeiler-Lehman kernel. Measures for neighborhood overlap. Graph clustering and spectral methods –Graph Laplacians, Graph Cuts and clustering, Spectral clustering. Semi-supervised Learning on Graph - Label Propagation Algorithm and Label Spreading Algorithm.

14 hours

Unit 2:Graph Representation Learning Based on shallow embedding methods

Encoder-Decoder perspective – encoder, decoder, and optimization of the encoder-decoder model. Factorization-based approaches - Random Walk-based embedding –DeepWalk and Node2Vec. Limitations of shallow embedding. Shallow embedding in multi-relational Knowledge Graph- embedding as reconstruction task, loss function, and decoders.

14 Hours

Unit 3:Graph Learning using graph neural network

Vanilla GNN - Neural message passing framework. Generalized neighborhood aggregation. Generalized update methods. Graph Convolution Networks (GCN), GraphSAGE, Multi-relational GCN and Graph Attention Networks

(GAT), Graph Transformer. Graph pooling. Applications and loss functions. Efficiency issues in GNN Modelling– GNN layer optimization, Stacking GNN layers, GNN modeling pipeline with different prediction heads, Graph augmentation, and Setting up graph datasets for a learning task.

14 Hours

Unit 4:Deep Generative Models, GNN on complex graphs and interdisciplinary applications

Deep Generative models on graph – Variational autoencoder on the graph, Generative adversarial network on the graph, Auto-regressive methods. GNN on complex graphs – Heterogeneous GNN, Dynamic GNN, Hypergraph GNN. Interdisciplinary Applications of GNN: Unstructured data – Text and Image. Structured Data - Social Network, Recommender System, Cyber Security. An introduction to trustworthy graph AI. Meta-Learning perspectives of graph ML.

14 Hours

Tools/Languages: NetworkX for statistical features of graphs, Tensorflow, Keras, and Scikit Learn for traditional graph ML, and Pytorch Geometric for Graph Neural Networks.

Text Book(s):

1: “Graph Representation Learning”, William L Hamilton, Morgan and Claypool Publishers, 2020.
2: “Deep Learning on graphs”, Yao Ma and Jiliang Tang, Cambridge University Press, 2021

Reference Book(s):

1: “Introduction to Graph Neural networks”, Zhiyuan Liu and Jie Zhou, Synthesis Lectures on Artificial Intelligence and Machine learning, Morgan and Claypool Publishers, 2020.
2: “Social Media Mining”, Reza Zafarani, Cambridge University Press, 2015 (Asian Economic edition available)
3: “ Network Science” by Albert Barabasi , 2016 .

UE21CS343BB5: Wireless Mobile Networking (4-0-0-4-4)

Wireless Mobile Networking is a dynamic field that has spurred tremendous excitement and technological advances. This course will cover topics in wireless networking and mobile computing. The objective of the course is to introduce students to recent advances in mobile networking, with an emphasis on practical design aspects of mobile systems.

Course Objectives:

- Introduce the different trends of wireless network technologies
- Applications of various wireless communication fundamentals based on its usage
- Provide cellular systems and standards for different generations, and explore cellular concept of design fundamentals such as Frequency Reuse, Handoff etc.
- Discuss design parameters and applications of computing and architecture for different mobile standards

Course Outcomes:

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

- Develop knowledge on wireless LANs and its evolution
- Understand the concepts of wireless communications and wireless networking
- Learn cellular concepts, e.g. frequency reuse and multiple access technologies
- Understand cellular evolution from 1G to 5G and associated mobility management

Desirable Knowledge : UE21CS252B – Computer Networks.

Unit 1 : Overview of Wireless communication

Introduction, Wireless Local Area Networks: IEEE 802.11, 802.11 Frame Format, Basic Access Methods – CSMA/CA, ALOHA, P-persistent CSMA, Non-persistent CSMA, Distribution Coordinating Function (DCF), DIFS, SIFS, Point Coordination Function (PCF), Light Weight Access Point Protocol (LWAPP), IEEE 802.1x, 802.11a,h,k,p, WPAN Technologies: Bluetooth, Bluetooth Protocol Stack, Piconets, NFC, 6LOWPAN, LPWAN-LORA, Wireless Local Loop (WLL)-Local Multipoint Distribution System (LMDS), MMDS, WiMAX - QDMA, Adhoc Networks and Routing, MANET & VANET, Wireless Sensor Networks, Self-Organizing Networks, Wireless Mesh Networks, RFID: Concept, frequency band, classification of RFID tags, applications.

14 Hours

Unit 2 : Wireless Communication Fundamentals

Multiple Access for Wireless Systems – FDMA, TDMA, CDMA, Multiple-Input and Multiple-Output (MIMO), Coding Techniques – Pulse Code Modulation (PCM), Differential PCM (DPCM), Adaptive PCM (ADPCM), Delta Modulation (DM), Wireless Modulation Schemes – Binary Phase-Shift Keying (BPSK), Quadrature Phase-Shift Keying (QPSK), Gaussian Minimum-Shift Keying, Frequency Spectrum, Long Range Communication- Satellite Communication, Smart Antennas, Wireless networking and security issues in 802.11 – Static filtering based on MAC address, Wired Equivalent Privacy (WEP), 802.11e QoS issues – DCF & HCF, Scanning, Increased Bit Error Rate (BER), Multipath Propagation, Authentication, Path Loss, RF signal interference, traffic and resource allocation – flow control, error control, mobility, routing, Channel Allocation Scheme, Power Management.

14 Hours

Unit 3 : Fundamentals of Cellular System

Evolution of Wireless Network generations (1G, 2G, etc), Mobile Radio standards – AMPS, N-AMPS, GPRS, GSM, UMTS, CDMA 2000, roaming, Cellular Concept – cell structure, cluster, frequency reuse, basic cellular system: mobile terminal, base station (BS), mobile switching center (MSC), home location register (HLR) and visitor location register (VLR), traffic and control channel (forward and reverse).

Handoff Strategies: Concept of handoff, Types of handoff – hard and soft handoff, Queued Delay, MAHO (Mobile Assisted Handoff), Improper handoff, Umbrella cell approach, Improving coverage and capacity in cellular systems: Cell splitting, Microcell Zone concept, Repeaters for range extension.

Unit 4 : Digital Cellular Mobile Standards

Global System for Mobile Communication (GSM) – features, architecture, GSM call routing, stages of call processing in GSM, Signaling System No. 7 (SS7), Need for 3G, 4G and 5G technologies, UMTS/W-CDMA standard – features, architecture, specifications and other procedures
Next generation mobile standards – features of 4G & 4G LTE, VoLTE, 5G .

14 Hours

Tools and Languages : Wireshark, Claynet, Cisco Packet Tracker.

Text Book(s):

1: Wireless and Mobile Network Architecture", Lin Yi-Bang and Clamtac Imrich, John Wiley & Sons 2001.

Note : There is no fixed text book. Lectures will be drawn from several sources (books, articles, papers) as most available texts do not cover all of the course material. The text book listed covers some of the topic that are relevant to the material presented.

Reference Book(s):

1: "Wireless Communications: Principles and Practice", Theodore S. Rappaport.
2: "Wireless and Mobile Networks: Concepts and Protocols", Dr. Sunil Kumar S. Manvi, Mahabaleswar S. Kakkasageri, Wiley India.
3: Wireless Communication Networks and System", by Cory Beard and William Stallings,1st edition, Pearson, 2015.
4: "Wireless Communication and Networks: 3G and Beyond", ITI Saha Mishra, McGraw-Hill Education.
5: "Wirless and Mobile Network Architectures", Lin Yi-Bang and Clamtac Imrich, John Wiley & Sons 2001.
6: "Wireless Network Security: A Beginner's Guide"; by Tyler Wrightson, McGraw-Hill Education.

UE21CS343BB6 : Information Security (4-0-0-4-4)

This course will present security aspects from a secure software life cycle process – requirement, architecture, design, coding, and testing. Students will have opportunity to dwell well in to technical "how to" with hands-on sessions, assignments, and some case study discussions.

Course Objectives:

- To understand various cyber threats and attacks and secure software development process.
- To learn attack and defence mechanisms for buffer overflow, shellshock attack, etc.
- To understand the concept of threat modelling and its application.
- To learn about the most common web application security vulnerabilities.
- To understand and apply various penetration testing techniques and tools.

Course Outcomes:

At the end of this course, the student will be able to:

- Identify possible misuse cases in the context of software development.
- Defend against various attacks and how to write secure code.
- Apply threat modelling techniques to expose inherent vulnerabilities in applications.
- Design and develop secure web applications.
- Exploit software vulnerabilities and launch attacks.

Course Content:

Unit 1: Introduction and Privilege Escalation Attacks

Software Threats, Attacks and Vulnerabilities, CIA Triad, OWASP Top 10, CVE, Security and reliability, Security vs. privacy, Cyberattack Types, Anatomy of an Attack, Security Concepts and Relationships. Use cases and Misuse cases, Misuse case legend, Security use case vs Misuse case, Secure Software Development Life Cycle (SDL). Case Study: Target case study. Set-UID program: Need for privileged programs, Set-UID mechanism, Superman story, Attack surfaces, Invoking other surfaces, Principle of least privilege. Environment variables and attacks: Environment variables, Attack surface, Attacks via Dynamic linker, External program, and Library. Lab: Set-UID program & Environment variables and attacks. Shellshock vulnerability, Shellshock attack on Set-UID and CGI programs.

14 Hours

Unit 2: Software Vulnerabilities and Malicious Software

Buffer overflow attack: Program memory layout, Stack and function invocation, Stack buffer-overflow attack, Attacks with Unknown address and Buffer size, Shellcode, Countermeasures & Defeating it. Return-to-libc attack: Introduction, Launch the attack part I & part II. Format string vulnerability: Introduction to functions and format string, Vulnerable program, Exploiting the vulnerability, Code injection attack, Countermeasures.**Case study: Target case study**. Malware and its Types, Malware analysis: Conifer, Morris, Stuxnet worm, Ransomware.

14Hours

Unit 3: Threat modelling and Basic Web Security

Threat Modelling, Trust Boundaries, Attack Surfaces, Brainstorming, Modelling Methods, STRIDE model and variants, Defensive tactics, and Technologies, Privacy Threats, Taxonomy and Types. Web security basics, Cross Site Request Forgery (XSRF/CSRF): Cross-site requests and its problems, CSRF attacks, Attacks on HTTP GET and POST services, Countermeasures. Case study: Apple - Privacy vs Safety.

14 Hours

Unit 4: Web application security and Penetration Testing

(XSS/CSS) Attack: CSS attack, CSS attacks in action, Self-propagation, Preventing CSS attacks.SQL injection attack: Introduction to SQL, interacting with database in web, Launching SQL injection attacks, Countermeasures. Apple Case Study, Static analysis, Penetration testing: Introduction, Benefits, Drawbacks, Penetration testing tools and Fuzzing.

14 Hours

Note: Hands-on experience for relevant topics in the form of Lab and/or Assignment is given. Relevant cyber security case study for undergraduate students is discussed.

Hands-on exercises:

1: Set-UID program & Environment variables and attacks
2: Shellshock attack.
3: Buffer overflow attack .
4: Return-to-libc attack.
5: Format string vulnerability.
6: CSS attack
7: CSRF attack.
8: SQL injection.
9: Intetration testing on live websites

Tools / Languages: SEED Labs VM, Scapy, Burp Suite, Metasploit, Nmap, etc.

Text Book(s):

1: "Computer & Internet Security: A Hands-on Approach", Wenliang Du, 2nd Edition, 2019.

Reference Book(s):

1: "Computer Security: Principles and Practice", William Stallings and Lawrie Brown, Pearson Education, 3rd Edition, 2014.
2: "Secure Programming with Static Analysis", Brian Chess and Jacob West, Pearson Education, 2007

UE21CS343BB7:Mobile and Autonomous Robots (4-0-0-4-4)

This course introduces students to the field of autonomous mobile robots. Students will learn the basic concepts and techniques necessary for designing, building, and programming autonomous robots, as well as explore the latest developments in this exciting field. The course will include lectures, hands-on activities, and projects.

Course Objectives:

- Understanding the past, present, and future of autonomous mobile robotics. Getting exposed to the current state of-the-art of scientific literature.
- The course aims to teach the theoretical and practical fundamentals involved in designing and operating autonomous robots or intelligent agents.
- The introductory discussions cover subtopics like robot perception, planning, and control.
- Other major topics include designing robot parts, integrating sensors, analyzing motion kinematics, simulation testing using ROS/ROS2, handling unmodeled environmental and social factors, and preparing for field deployment.
- The course covers both terrestrial and aerial robotic systems.

Course Outcomes:

At the end of this course, the student will be able to:

- Students will gain a comprehensive understanding of autonomous robots and their applications.
- Understand the principles of robot manipulators, end-effectors, and mobile robots, and identify the sensors and actuators used in robot control.
- They will be able to interface with robot hardware using ROS/ROS2 programming language.
- Students will understand the ROS architecture and communication protocols and use ROS packages for robot software control.
- They will be able to use path planning and obstacle avoidance algorithms for robot navigation.
- Ultimately, students will be able to apply this knowledge to design and develop their own autonomous robot systems.

Course Content:

Unit 1: Introduction to Autonomous Robots

Overview of autonomous robots, History and current state of autonomous robots, Basic concepts and terminology, Applications of autonomous robots, Robot Hardware and Software: Sensors and actuators, Motors and controllers, Power sources and management, Robot kinematics and dynamics, Overview of robot sensors: range finders, encoders, vision.Programming languages and environments for robots, Robot operating systems, Control algorithms and architectures, Robot perception and decision-making.ROS overview: ROS/ROS2 for robotics, ROS architecture and communication protocols, ROS packages for robot hardware control.

14 Hours

Unit 2: Locomotion Mechanisms for Robots

Ground robots (UGVs) 2-DOF and 3-DOF robots, Forward and inverse kinematic, The basics of wheel types and arrangements. Aerial robots (UAVs): 5-DOF and 6-DOF robots, Dynamics of thrusters and propellers, The basics of SOTA locomotion systems.

14 Hours

Unit 3: Perceptionin Autonomous Robots

Visual and inertial measurements: Gyroscope, accelerometers, IMU, GPS, Range sensors, camera vision and LiDAR. Visual perception: Robot vision basics - UGVs / AUVs / UAVs, Image processing and filtering, Object detection and tracking, Stereo vision and 3D perception, The basics of scene segmentation and parsing.

14 Hours

Unit 4: Navigation and Localization for Robots

Path planning: motion planners and path planners,Dynamic programming, UGV and UAV planning: social and behavioural aspects,the basics of AUV planning.SLAM: Simultaneous localization and mapping,2-DOF, 3-DOF, and 6-DOF robots, Acoustic and optical localization, Obstacle avoidance, Sensor fusion, Reinforcement Learning using AI/ML, Applications and Social Implications.

14 Hours

Tools and Languages : C, C++, Python, ROS.

Recommended Materials
1: Robot Operating System (ROS): http://wiki.ros.org/ROS/Tutorials
2: ROS2 tutorials: https://docs.ros.org/en/foxy/Tutorials.html

Text Book(s):
1: "Introduction to Autonomous Mobile Robots (Intelligent Robotics and Autonomous Agents series)", Roland Siegwart, Illah Reza Nourbakhsh, Davide Scaramuzza 2nd Edition
2: "Probabilistic Robotics", Sebastian Thrun, Wolfram Burgard and Dieter Fox.ISBN-13: 978-0262201629, ISBN-10: 0262201623.Intelligent Robotics and Autonomous Agents series; 1st Edition

Reference Book(s):
1: "Introduction to Autonomous Robots:", Nikolaus Correll, Magellan Scientific, 2016.
2: "ROS Robot Programming, ROBOTIS Co., Ltd.From the basic concept to practical programming and robot application", YoonSeokPyo, HanCheol Cho, RyuWoon Jung, and TaeHoon Lim.
3: "Introduction to Robotics: Mechanics and Control", 4th Edition, John Craig, ISBN-13: 978-0133489798, Pearson; 4th edition.

UE21CS343BB8 - Security for Internet of Things

This course enables the learner to understand the security threats associated with Internet of Things and perform security testing on connected devices within our homes and enterprise to build a better model for protecting ourselves.

Course Objective:

- Understand the threats of IoT devices
- Understand vulnerabilities, possible attacksand learn the security testing methodology for networks, IoT hardware and radio protocols.
- Enhance knowledge with hands-on experience on IoT security tools and attack analysis.
- Understand the various Security measures to be adopted to secure IoT devices

Course Outcome:

- Identify and describe the variety of IoT systems architectures, essential components and challenges specific to IoT systems.
- Analyze various network and hardware security mechanisms for IoT devices
- Gain hands-on experience on different tools to target IoT ecosystem.
- Analyze and apply appropriate security and privacy solutions for real-world applications.

Unit 1: Introduction to IoT Security

Introduction to IoT Security, Traditional security vs IoT security,Basic concepts of IoT Architecture from security perspective, Challenges of IoT Security, OWASP Top 10 security risks and consumer IoT security guidance, Threat Modelling for IoT attack, Common IoT threats, Network Hacking: VLAN hopping in IoT networks.

14 Hours

Unit 2: Network Security and Hardware Security for IoT devices

MQTT authentication, Analysing Network protocols: Wireshark dissector and Nmap Scripting Engine module for the DICOM protocol, exploiting zero-configuration Networking: UPnP, mDNS, DNS-SD, and WS-Discovery, Hardware Hacking: UART, JTAG, and SWD Exploitation- Hacking an STM32F103 microcontroller using UART and SWD.SPI and I2C, Firmware Hacking.

14 Hours

Unit 3: Radio Hackingand Smart IoT devices hacking

Radio Hacking:Short Range Radio: Abusing RFID, Bluetooth low energy, Medium Range Radio: Hacking Wi-Fi, Long Range Radio: LPWAN, Smart home, Hacking the smart home: Gaining physical entry to a building, cloning a keylock system's RFID Tag, Jamming the Wireless alarm.

14 Hours

Unit 4: Targeting the IoT Ecosystem and Secure Design of IoT Devices

Playing back an IP Camera stream, Analysing IP Camera Network traffic, Extracting the video stream, attacking a Smart treadmill, Secure design goals: Mitigate automated attack risks, Secure points of integration, Hardware protection measures, IoT IAM infrastructure: PKI for IoT, Revocation support: OCSP, SSL pinning, Authorization and access control with OAuth 2.0 Cryptographic Controls for IoT Protocols.

14 Hours

Tools and Languages : Wireshark, Yersinia, VoIP Hopper, Bettercap,aircrack-ng

Textbook(s):

1: "Practical IoT Hacking", Fotios Chantzis, Ioannis Stais, Paulino Calderon, Evangelos Deirmentzoglou, Beau Woods, March 2021, No Starch Press Publishers, ISBN: 9781718500907

Reference(s):

1: "Practical Internet of Things Security" , Brian Russell, Drew Van Duren, Packt Publishers, 2nd Edition