# B. TECH IN COMPUTER SCIENCE AND ENGINEERING

## PROGRAM EDUCATIONAL OBJECTIVES

- Prepare and train students in theoretical foundations to work with cutting edge computing technologies and design solutions to complex engineering problems, making them ready to work in industrial environment.
- Develop all round skills such as team building, inter-personal skills, and leadership qualities in order to effectively communicate with engineering community and with society at large.
- Promote research culture through internships, research assistantships, research-oriented projects, sponsored and collaborative research and enable them to pursue higher studies in computer science and related fields.
- To inculcate social concern meeting the requirements of prospective employers and to develop an ability to innovate efficient computing solutions for a better society.
- Create professionally superior and ethically strong globally competent employees and entrepreneurs.

## PROGRAM OUTCOMES

- Apply mathematical and theoretical principles in the modelling and design of high-quality computer-based systems using state-of-the-art computer technology.
- Conduct in-depth study of research literature in the area of Computer Science, analyse problems in order to arrive at substantiated conclusions using first principles of mathematics, and allied sciences.
- Design, implement and evaluate Computer Systems, programs and processes that meet partial/ complete specifications with concern for society, environment and culture.
- Design and conduct experiments, collect data, analyze and interpret the results to investigate complex engineering problems in the field of Computer Science.
- Apply state-of-the-art techniques and modern computer-based tools in prediction, comparison and modelling of complex engineering activities.
- Have a sound understanding of professional, legal, security and social issues and responsibilities in engineering activities involving Computer Science.
- Understand societal and environmental concerns and demonstrate responsibility in sustainable development of computer-based solutions.
- Be aware of ethical and professional responsibilities in engineering situations; make informed judgments regarding intellectual property and rights in relation to computer-based solutions in global, economic, environmental and societal contexts.
- Able to function effectively in teams to establish goals, plan tasks, meet deadlines, manage risk and produce high-quality technical solutions.
- Contribute and communicate effectively with the society, be able to write effective reports and design documents by adhering to appropriate standards, make effective presentations, give and receive clear instructions.
- Apply skills in clear communication, responsible teamwork and time management by, for example, managing a team or project and communicating with external stakeholders.
- Recognize the need for and demonstrate an ability to engage in continuing professional development in its broadest sense.

## III SEMESTER (2020-24 BATCH)

| Sl. No. | Course Code | Course Title | Hours per week | | | | Credits | Tools / Languages | Course Type |
|---|---|---|---|---|---|---|---|---|---|
| | | | L | T | P | S | C | | |
| 1 | UE20CS201 | Digital Design and Computer Organization | 4 | 0 | 0 | 4 | 4 | | CC |
| 2 | UE20CS202 | Data Structures and its Applications | 4 | 0 | 0 | 4 | 4 | C | CC |
| 3 | UE20CS203 | Statistics for Data Science | 4 | 0 | 0 | 4 | 4 | Python | CC |
| 4 | UE20CS204 | Web Technologies | 4 | 0 | 0 | 4 | 4 | HTML, CSS, JavaScript, MERN Technologies. | CC |
| 5 | UE20CS205 | Automata Formal Languages and Logic | 4 | 0 | 0 | 4 | 4 | JFLAP | CC |
| 6 | UE20CS206 | Digital Design and Computer Organization Laboratory | 0 | 0 | 2 | 1 | 1 | Icarus, Verilog Simulator, GTKWave waveform viewer | CC |
| 7 | UE20CS207 | Data Structures and its Applications Laboratory | 0 | 0 | 2 | 1 | 1 | In house plagirisam tool / C | CC |
| 8 | UE20CS208 X | Special Topic I | 0 /2 | 0 | 0/4 | 0/4 | 2 | | ST |
| 9 | UE21MA101D | Bridge Course Mathematics –I (Applicable to Lateral Entry Students) | 2 | 0 | 0 | 0 | 2 | | FC |
| **Total** | | | **20/22** | **0** | **4/8** | **22/26** | **24/26** | | |
| **Note : Desirable Knowledge – None** | | | | | | | | | |

## IV SEMESTER (2020-24 BATCH)

| Sl. No. | Course Code | Course Title | Hours per week | | | | Credits | Tools / Languages | Course Type |
|---|---|---|---|---|---|---|---|---|---|
| | | | L | T | P | S | C | | |
| 1 | UE20MA251 | Linear Algebra | 4 | 0 | 0 | 4 | 4 | SciLab, Python | CC |
| 2 | UE20CS251 | Design and Analysis of Algorithms | 4 | 0 | 0 | 4 | 4 | C-Language, GCC Compiler | CC |
| 3 | UE20CS252 | Microprocessor and Computer Architecture% | 4 | 0 | 0 | 4 | 4 | ARM Simulator | CC |
| 4 | UE20CS253 | Computer Networks | 4 | 0 | 0 | 4 | 4 | Wireshark, Python | CC |
| 5 | UE20CS254 | Operating Systems@ | 4 | 0 | 0 | 4 | 4 | Pthread,C,Linux/Unix OS | CC |
| 6 | UE20CS255 | Computer Networks Laboratory | 0 | 0 | 2 | 1 | 1 | Wireshark, Claynet, Cisco packet tracer | CC |
| 7 | UE20CS256 | Microprocessor and Computer Architecture Laboratory | 0 | 0 | 2 | 1 | 1 | ARM Simulator, Ardino microcontroller kit, MIPS pipeline simulator, ParaCache simulator. | CC |
| 8 | UE20CS257 X | Special Topic II | 0 /2 | 0 | 0 /4 | 0 /4 | 2 | | ST |
| 9 | UE21MA151D | Bridge Course Mathematics – II (Applicable to Lateral Entry Students) | 2 | 0 | 0 | 0 | 2 | | FC |
| **Total** | | | **20/22** | **0** | **4/8** | **22/26** | **24/26** | | |

**Note : Desirable Knowledge - %UE20CS201, @UE20CS202.**

## V SEMESTER (2019-23 BATCH)

| Sl. No. | Course Code | Course Title | Hours per week | | | | Credits | Tools / Languages | Course Type |
|---|---|---|---|---|---|---|---|---|---|
| | | | L | T | P | S | C | | |
| 1 | UE19CS301 | Database Management System | 4 | 0 | 0 | 4 | 4 | PostgreSQL 13.3, ERwin | CC |
| 2 | UE19CS302 | Software Engineering | 4 | 0 | 0 | 4 | 4 | GitHub, MS Project, Jenkins. | CC |
| 3 | UE19CS303 | Machine Intelligence* | 4 | 0 | 0 | 4 | 4 | Tensorflow 1.15, Keras 2.3.1, Python 3.7. | CC |
| 4 | UE19CS304 | Database Management System Laboratory | 0 | 0 | 2 | 1 | 1 | Oracle, MySQL,SQL Server, PostgreSQL. | CC |
| 5 | UE19CS305 | Machine Intelligence Laboratory | 0 | 0 | 2 | 1 | 1 | Python(3.7x), sk learn(v0.23), Keras(v2.2.4), Tensorflow(v1.14). | CC |
| 6 | UE19CS31X | Elective I | 4 | 0 | 0 | 4 | 4 | | EC |
| 7 | UE19CS32X | Elective II | 4 | 0 | 0 | 4 | 4 | | EC |
| 8 | UE19CS306X | Special Topic-III | 0 | 0 | 4 | 2 | 2 | | ST |
| **Total** | | | **20** | **0** | **8** | **24** | **24** | | |
| **Elective – I** | | | | | | | | | |
| 9 | UE19CS311 | Advanced Algorithms% | 4 | 0 | 0 | 4 | 4 | C or C++ | EC |
| 10 | UE19CS312 | Data Analytics& | 4 | 0 | 0 | 4 | 4 | Python and R | EC |
| 11 | UE19CS313 | Internet of Things | 4 | 0 | 0 | 4 | 4 | Python or C | EC |
| 12 | UE19CS314 | Applied Cryptography | 4 | 0 | 0 | 4 | 4 | Seed virtual machine environment, gcc, python. | EC |
| 13 | UE19CS315 | Fundamentals of Virtual and Augmented Reality!!! | 4 | 0 | 0 | 4 | 4 | C, C++, Java, Python using OpenGL | EC |
| 14 | UE19CS316 | Human Computer Interaction | 4 | 0 | 0 | 4 | 4 | | EC |
| **Elective – II** | | | | | | | | | |
| 15 | UE19CS321 | Principles of Programming Languages | 4 | 0 | 0 | 4 | 4 | GCC /g++, Ada, Python, Prolog, Haskell, gdb ,pdb,Ruby,Java | EC |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 16 | UE19CS322 | Big Data[$] | 4 | 0 | 0 | 4 | 4 | Hadoop, HDFS Spark, Streaming spark, HIVE, Hbase, Mllib. | EC |
| 17 | UE19CS323 | Graph Theory and Its Applications[!] | 4 | 0 | 0 | 4 | 4 | C. | EC |
| 18 | UE19CS324 | Bio-inspired Computing[%] | 4 | 0 | 0 | 4 | 4 | Matlab | EC |
| 19 | UE19CS325 | Advance Computer Networks[%%] | 4 | 0 | 0 | 4 | 4 | Claynet, Cisco packet tracer | EC |
| 20 | UE19CS326 | Computer Network Security[%%] | 4 | 0 | 0 | 4 | 4 | Seed Ubuntu VM, Wireshark, Snort, NetwoX, Scapy. | EC |

**Note: Desirable Knowledge - Core : [*]- UE19CS203, UE19MA251, UE19CS251.**

**Desirable Knowledge – Elective I : [%]- UE19CS251, [&]- UE19CS203, [!!!]- UE19CS202.**

**Desirable Knowledge – Elective II : [$]-UE19CS202, UE19CS251, [!]- UE19CS151, UE19CS202, [%]- UE19CS251, [%%]- UE19CS253.**

**ELECTIVES TO BE OPTED FOR SPECIALIZATION**

| **Sl. No.** | **SPECIALIZATION** | **ELECTIVE – I** | **ELECTIVE – II** |
|---|---|---|---|
| A | System and Core Computing(SCC) | UE19CS311, UE19CS315. | UE19CS321, UE19CS322, UE19CS323. |
| B | Machine Intelligence and Data Science(MIDS) | UE19CS312, UE19CS313, UE19CS315, UE19CS316. | UE19CS322, UE19CS323, UE19CS324. |
| C | Network and Cyber Security(NWCS) | UE19CS313, UE19CS314. | UE19CS325, UE19CS326. |

## VI SEMESTER (2019-23 BATCH)

| **Sl. No.** | **Course Code** | **Course Title** | **Hours per week** | | | | **Credits** | **Tools / Languages** | **Course Type** |
|---|---|---|---|---|---|---|---|---|---|
| | | | **L** | **T** | **P** | **S** | **C** | | |
| 1 | UE19CS351 | Compiler Design[!] | 4 | 0 | 0 | 4 | 4 | Lex and Yaac. | CC |
| 2 | UE19S352 | Cloud Computing[@@] | 4 | 0 | 0 | 4 | 4 | Amazon AWS, Docker, Kubernetes, Github, NoSQL, databases, Flask. | CC |
| 3 | UE19CS353 | Object Oriented Analysis & Design with Java | 4 | 0 | 0 | 4 | 4 | Star UML, Java. | CC |
| 4 | UE19CS354 | Cloud Computing Laboratory | 0 | 0 | 2 | 1 | 1 | Amazon AWS, Docker, Kubernetes, Github, NoSQL, databases, Flask. | CC |
| 5 | UE19CS355 | Object Oriented Analysis & Design with Java Laboratory | 0 | 0 | 2 | 1 | 1 | Eclipse, Star UML | CC |
| 6 | UE19CS33X | Elective III | 4 | 0 | 0 | 4 | 4 | | EC |
| 7 | UE19CS34X | Elective IV | 4 | 0 | 0 | 4 | 4 | | EC |
| 8 | UE19CS390A | Capstone Project Phase-1 | 0 | 0 | 8 | 2 | 2 | | PW |
| | | **Total** | **20** | **0** | **12** | **24** | **24** | | |
| **Elective – III** | | | | | | | | | |
| 9 | UE19CS331 | Generic Programming[#] | 4 | 0 | 0 | 4 | 4 | C, C++, C# | EC |
| 10 | UE19CS332 | Algorithms for Intelligence Web and Information Retrieval[**] | 4 | 0 | 0 | 4 | 4 | Scikit, Tensorflow, Solr, Lucene Search Engines/ Python | EC |
| 11 | UE19CS333 | Image Processing and Computer Vision[**] | 4 | 0 | 0 | 4 | 4 | Matlab, Python. | EC |
| 12 | UE19CS334 | Natural Language Processing[##] | 4 | 0 | 0 | 4 | 4 | Tensorflow, Spacy , NLTK, SCIKIT , Python 3.x. | EC |
| 13 | UE19CS335 | BlockChain[!] | 4 | 0 | 0 | 4 | 4 | Claynet, Python | EC |
| 14 | UE19CS336 | Digital Cyber Forensics | 4 | 0 | 0 | 4 | 4 | Open source Forensics Tools | EC |
| 15 | UE19CS337 | Hardware Accelerated Computing[?] | 4 | 0 | 0 | 4 | 4 | | EC |
| **Elective – IV** | | | | | | | | | |
| 16 | UE19CS341 | Design Patterns[**] | 4 | 0 | 0 | 4 | 4 | C, C++, Java, C#. | EC |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 17 | UE19CS342 | Heterogeneous Parallelism!!! | 4 | 0 | 0 | 4 | 4 | pthread, OpenMP CUDA, openCL, Chapel, UPC. | EC |
| 18 | UE19CS343 | Topics in Deep Learning## | 4 | 0 | 0 | 4 | 4 | Tensorflow 1.15, Keras 2.3.1/ Python 3.7 | EC |
| 19 | UE19CS344 | Database Technologies*** | 4 | 0 | 0 | 4 | 4 | MS SQL Server, Oracle, Apache Spark, Storm, Kafka, Flink, Amazon Kinesis. | EC |
| 20 | UE19CS345 | Network Analysis and Mining%%% | 4 | 0 | 0 | 4 | 4 | Gephi, NetworkX, Pytorch/Tensorflow, Python. | EC |
| 21 | UE19CS346 | Information Security | 4 | 0 | 0 | 4 | 4 | Seed Labs, Scapy, Burp-Suit,N-Map, 'C' | EC |
| 22 | UE19CS347 | Wireless Network Communication%% | 4 | 0 | 0 | 4 | 4 | Wireshark,Claynet, Cisco Packet Tracer | EC |

**Note: Desirable Knowledge – Core : !UE19CS202, UE19CS205,@@UE19CS253,UE19CS254.**

**Desirable Knowledge – Elective III : #UE19CS151, UE19CS202, UE19CS251 **- UE19CS251, ##UE19CS303, !UE19CS202, ? UE19CS201.**

**Desirable Knowledge – Elective IV : ** UE19CS251, !!!-UE19CS151, UE19CS252, ##-UE19CS303, *** - UE19CS301, %%% - UE19MA251,UE19CS303, %% - UE19CS253.**

**ELECTIVES TO BE OPTED FOR SPECIALIZATION**

| **Sl. No.** | **SPECIALIZATION** | **ELECTIVE – III** | **ELECTIVE – IV** |
|---|---|---|---|
| A | System and Core Computing(SCC) | UE19CS331, UE19CS332, UE19CS337. | UE19CS341, UE19CS342 UE19CS344. |
| B | Machine Intelligence and Data Science(MIDS) | UE19CS332, UE19CS333, UE19CS334. | UE19CS343, UE19CS345. |
| C | Network and Cyber Security(NWCS) | UE19CS335, UE19CS336. | UE19CS346, UE19CS347. |

## UE20CS201: Digital Design and Computer Organization (4-0-0-4-4)

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 Design**

Introduction, Boolean Functions, Truth tables, Boolean Algebra, Identities, Logic minimization, K-Maps, Adder/Subtractor, Overflow.- **12 Hours**

**Unit 2 : Combinational and Sequential Logic Design**

Muxes, Decoders, Shifters, Gate/Wire delays, Timing, Latches, Flip-flops, Synchronous logic design, Finite State Machines. - **12 Hours**

**Unit 3: Sequential and Arithmetic Circuits**

FSM examples, Counters, Memory arrays, Carry-look ahead and Prefix adders. - **10 Hours**

**Unit 4: Arithmetic Circuit and Architecture**

Shift/add multiplier/divider, Wallace tree multiplier, Floating point, Assembly Language, Machine Language. - **10 Hours**

**Unit 5 : Micro Architecture**

Addressing Modes, Performance analysis, Single-cycle, Multi-cycle processor data path and control, Systolic array matrix multiply, PC I/O Systems.- **12 Hours**

**Text Book:**

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.

## UE20CS202: Data Structures and its Applications (4-0-0-4-4)

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.
- 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.
- Apply skills required to implement any data structure.
- Minimal data structure that supports all operations needed.
- Appropriate data structure in competitive programming.
- Design and develop efficient software systems with good knowledge of data structures.

**Course Content:**

**Unit 1 : Overview**

Recursion,User defined data type, pointers, pointer to structures.-Static and Dynamic Memory Allocation. Linked List: Doubly Linked List, Circular Linked List – Single and Double, Multilist: Introduction to sparse matrix (structure). Application: Case Study -Text Editor, Assembler - Creation of a Symbol Table. **-12 Hours**

**Unit 2 : Stacks and Queues**

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. 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). **- 12 Hours**

**Unit 3 : Trees and Heaps**

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. 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). **-12 Hours**

**Unit 4 : Balanced Trees and Graphs**

Balanced Trees: definition, AVL Trees, Rotation, 2,3 Trees.
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. 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. **-10 Hours**

**Unit 5 : Suffix Tree and Hashing**

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.- **10 Hours**

**Tool/ Languages:** C- Language

**Text Book:**
1.“Data Structures using C / C++” , Langsum Yedidyah, Moshe J Augenstein, Aaron M Tenenbaum Pearson Education Inc, 2nd edition, 2015.

**Reference Book:**
**1:** “Data Structures and Program Design in C”, Robert Kruse, Bruce Leung, C.L Tondo, Shashi Mogalla, Pearson, 2nd Edition, 2019.

## UE20CS203: 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**

Introduction, Sampling: Sampling Methods, Sampling Errors. Getting and Analyzing Data: Reading Files, Scraping the Web, Need for Data Cleaning and its Basics. Statistics: Introduction, Types of Statistics. Data Visualization and Interpretation: Histogram, Bar Charts, Box Plots, Scatter Plots, Heat Maps, Good vs. Bad Visualization, Case Study. **- 12 Hours**

**Unit 2 : Random Variables and Probability Distributions**

Random Variables, Expectation, Functions of Random Variables. Probability Distributions: Discrete Distributions (Bernoulli, Binomial, Poisson), Continuous Distributions (Normal), Derivation of Distributions, Generation of Random Variates, Case Study. **- 12 Hours**

**Unit 3 : Confidence Intervals**

Principles of Point Estimation - Mean Squared Error, Maximum Likelihood Estimate, The Central Limit Theorem and Applications, Normal Probability Plots. Confidence Intervals: Using Simulation to Construct Confidence Intervals, Interval Estimates for Mean of Large and Small Samples, Interval Estimates for Proportion of Large Samples, Confidence Intervals for the Difference between Two Means, Interval Estimates for Paired Data. Factors affecting Margin of Error, Case Study. **- 10 Hours**

**Unit 4 : Hypothesis and Inference**

Hypothesis Testing for Population Mean and Population Proportion of Large Samples, Drawing conclusions from the results of Hypothesis tests, Large sample tests for a Population proportion, Large - Sample tests for Difference between two means, Distribution Free Tests, Chi-squared Test, Fixed Level Testing, Type I and Type II Errors, Case Study.- **12 Hours**

**Unit 5 : Power of Test and Simple Linear Regression**

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, Case Study. **-10 Hours**

**Tools / Languages:** Python.

**Text Book:**
1: “Statistics for Engineers and Scientists”, William Navidi, McGraw Hill Education, India, 4th Edition, 2015.

**Reference Book:**
1: “The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling”, 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.

## UE20CS204: 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 and JavaScript
- The core concepts of HTML5, Advanced JavaScript, 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 the MongoDB through NodeJS
- 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 and JQuery
- Understand MERN stack layers (MongoDB, ExpressJS, ReactJS and NodeJS) and Create rich User Interface using React JS
- Integrate the UI with MongoDB database through NodeJS
- Create RESTful Web services using ExpressJS

**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), JavaScript objects.- **12 Hours**

**Unit 2 : HTML5, JQuery and Ajax**

DOM Manipulations, Events and Event Handling in JavaScript, HTML5 (APIs), JQuery Introduction, Callbacks and Promises, Introduction to Single Page Application, XML Vs JSON, Asynchronous Communication using AJAX and fetch API.- **12 Hours**

**Unit 3 : ReactJS**

MERN Introduction, React Classes and Components, JSX, Rendering of elements, Properties, State, Context, Component lifecycle methods, Refs & Keys, Event Handling, Stateless components. **-10 Hours**

**Unit 4 : MongoDB and NodeJS**

Understanding Node JS Architecture, Set up Node JS app, Node Modules, callbacks, buffers, streams, File system Module, HTTP Module, Handling HTTP Requests, MongoDB-Documents, Collections, Reading and Writing to MongoDB, MongoDB NodeJS Driver, Running a react application on NodeJS, React Router. **- 12 Hours**

**Unit 5 : 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.- **10 Hours**

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

**Text Books:**

1: "Learing 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.
3: https://reactjs.org/docs/
4: https://www.kirupa.com/react

## UE21CS205 : 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.
- To familiarize students to construct Regular Expressions, Regular Grammars and to identify Non – Regular Languages.
- 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 familiarize students with concepts like Recursively Enumerable languages, Recursive Languages, Undesirable Problems.
- 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.
- 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.- **10 Hours**

**Unit 2 : Regular Languages and Grammars**

Regular Expressions Connection between Regular Expressions and Regular Languages Regular Grammars. Properties of Regular Languages: Closure Properties of Regular Languages, Elementary Questions about Regular Languages, Identifying Non Regular Languages. - **10 Hours**

**Unit 3 : Pushdown Automata and Context Free Languages**

Definitions of PDA and CFL, Deterministic Pushdown Automata, Non-Deterministic Pushdown Automata, Pushdown Automata and Context Free Languages, Context Free Grammars, 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. - **12 Hours**

**Unit 4 : Properties of Context-Free Languages**

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. 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- **12 Hours**

**Unit 5 : Propositional Logic**
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. **- 12 Hours**

**Tools / Languages**: JFLAP.

**Text Books:**
1: "An Introduction to Formal Languages and Automata", Peter Linz, Jones and Bartlett, New Delhi, India, 5th Edition, 2011.
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.

## UE20CS206: Digital Design and Computer Organization Laboratory (0-0-2-1-1)

Digital Design and Computer Organization Laboratory essentially consists of implementation and simulation of a series of digital building blocks in an HDL (Hardware Description Language). The developed blocks are finally composed to yield a simple microprocessor.

**Course Objectives:**

- Explain the elements of digital system abstractions such as digital representations of information, Digital Logic, Boolean algebra, State Elements and Finite State Machine (FSMs).
- Design simple digital systems based on these digital abstractions, using the "Digital Paradigm" including discrete sampled information.
- Use the "Tools of the Trade" - Basic Instruments, Devices and Design Tools.
- Work in a design team that can propose, design, successfully implement and report on a digital systems project.
- Communicate the purpose and results of a design project in written and oral presentations.

**Course Outcomes:**

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

- Achieve knowledge and awareness of various components to design stable digital circuits.
- Analyze and design combinational circuits.
- Design and develop sequential circuits.
- Design and develop a basic microprocessor.
- Translate real world problems into digital logic formulations using Verilog.

**Course Content:**

1. Hardware Kit Assignment.
2. Verilog Basics – I.
3. Verilog Basics – II.
4. Mux, Adder Design.
5. ALU.
6. Register File.
7. Datapath.
8. Program Counter.
9. Control Logic.
10. Microprocessor.
11. Project Assignment (microprocessor based).

**Tools / Languages**: Icarus Verilog Simulator, GTKWave Waveform Viewer.

**Reference Book:**

1. Laboratory Manual prepared by the Department of Computer Science and Engineering, PES University.

## UE20CS207: Data Structures and its Applications Laboratory (0-0-2-1-1)

This laboratory focuses on concrete implementations of basic data structures such as Stacks, Queues, Linked lists, trees and graphs to be implemented in variety of ways. This allows the student to appreciate the choices and tradeoffs which face a programmer in a real situation.

**Course Objectives:**

- Basic approaches and mindsets for analyzing and designing data structures.
- 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, Graph.
- Implement a given application using the available data structure.

**Course Outcomes:**

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

- Relevant data structures for any given application.
- Apply skills required to implement any data structure.
- Minimal data structure that supports all operations needed.
- Appropriate data structure in competitive programming.

**Course Contents:**

1. Implement Singly Linked list and perform all types of insert & delete, display operations.
2. Implement Doubly Linked list and perform all types of insert & delete, display operations.
3. Implement Singly Linked Circular list and Doubly linked Circular list and perform insert, delete, display operations.
4. Implement operations of on Stack & its applications.
5. Implement operations of a Queue, Circular queue.
6. Implement Binary search tree and perform the operations of insert, delete and traverse the BST.
7. Implement priority Queue using heap( both min and max).
8. Implementation of TRIE tree.
9. Implementation of HASH function, Hash table. Use rehashing and double hashing techniques to resolve collision
10. Week #10 to Week #14: Implementation of case studies.

**Tools / Languages**: In House Plagiarism Tool, C – Language.

**Reference Book:**

1. Laboratory Manual prepared by Department of Computer Science and Engineering, PES University.

## UE21MA101D: Bridge Course Mathematics - I (2-0-0-2-2)

This course is designed to give a comprehensive coverage at an introductory level to the subject Of Differential Calculus, Partial Differentiation, Integral Calculus and Ordinary Differential Equations.

**Course Objectives:**

- To understand the concepts of Polar Curves, their Pedal forms, Curvatures, Radius of Curvatures and Taylor's, Maclaurin's Series expansion using differentiation.
- To understand the concept of Partial Differentiation, Total Differentiation, Homogeneous functions and Euler's Theorem.
- To understand the concepts of Double and Triple Integrals, Change of order, Change of variables, Tracing of Curves in Cartesian & Polar form.
- To study the solution of Differential Equation of first order and the Orthogonal Trajectories of a given family of curves.
- To understand Linear Differential Equations of higher order with Constant co-efficient.

**Course Outcomes:**

After completing this course, students will be able to:

- Evaluate Angle between two curves and Radius of Curvature for a given Polar Curve.
- Analyze a function of more than one independent variable to find its derivative, expand it as Taylor's & Maclaurin's Series.
- Evaluate Double and Triple Integrals by changing the order and also by changing the variables.
- Apply the concepts of Double and Triple Integrals for finding Area and Volume.
- Evaluate the solution of Linear Differential Equations using various methods.

**Course Content:**

**Unit 1 : Differential Calculus**

Polar Curves, Angle between Radius Vector and Tangent, Angle between two Curves, Pedal Equations, Radius of Curvature and its different forms (Statement only). Taylors and Maclaurin's series expansion for one variable. **- 6 Hours**

**Unit 2 : Partial Differentiation**

Introduction to Partial Differentiation, Total Derivative, Chain Rule, Partial Differentiation of Composite and Implicit Functions, Homogeneous Functions and Euler's Theorem.- **6 Hours**

**Unit 3 : Integral Calculus**

Introduction to Jacobians, Double Integrals, Change of order of Integration, Triple integrals- Simple Problems, Change of Variables (Polar, Spherical and Cylindrical coordinates). Self learning component: Tracing of Curves (Cartesian and Polar). **- 6 Hours**

**Unit 4 : Ordinary Differential Equations**

Linear and Reducible to Linear (Bernoulli as a particular case), Exact, Reducible to Exact, Orthogonal Trajectories (Cartesian and Polar Forms).Department of Science & Humanities Self Learning Component: Variable Separable. **- 6 Hours**

**Unit 5 : Higher Order Differential Equations**

Introduction to Higher Order Differential Equations, Complementary Function and Particular Integrals of standard functions. Method of Variation of Parameters.- **5 Hours**

**Text Book:**

1: Higher Engineering Mathematics, B. S. Grewal, Khanna Publishers, New Delhi, 44 th edition, 2020.

**Reference Books:**

1: Higher Engineering Mathematics, B V Ramana. Tata McGraw-Hill Education, New Delhi, 23 rd print, 2015.
2: Advanced Engineering Mathematics, Erwin Kreyszig, Wiley (India), New Delhi, 10 th Edition, 2015.

3: Calculus, James Stewart, Cengage Publications, 8 th edition, 20

## UE20MA251: Linear Algebra (4-0-0-4-4)

This course aims to familiarize with some basic techniques in Matrix Theory used for approximating solutions and to organize information about Vector Spaces in a way that makes problems involving several variables easy.

**Course Objective:**

- To understand Matrix Operations to solve systems of Linear Equations and find the Inverse of a Matrix.
- To understand the basic concepts of Vector Spaces, Linear Transformations and Fundamental Subspaces.
- To understand the concept of Orthogonality of Vectors & Subspaces and Gram Schmidt Orthogonalization to produce Orthonormal Vectors.
- To understand the concept of Eigenvalues, Eigenvectors for Diagonalization, Singular Value Decomposition which is used to compute Pseudo Inverse, Least Squares fitting of data, Multivariable control and Matrix Approximation.
- Use Scilab to solve problems related to visualize the various concepts of Linear Algebra.

**Course Outcomes:**

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

- Solve systems of Linear Equations using Matrix Transformations, Interpret the nature of Solutions, Visualize Consistency of Linear system of Equations and also compute Inverse of a Matrix.
- Demonstrate the ability to work within Vector Spaces, distil Vector Space properties and understand the concepts of the four fundamental Subspaces, Linear Span, Linear Independence , Dimension and Basis
- Analyze Linear Transformation as a mapping and calculate its Matrix Representation with respect to Standard and Nonstandard Bases.
- Understand the concepts of Orthogonal Vectors and Orthogonal Subspaces and apply the Gram-Schmidt process to find an Orthonormal Basis in a Subspace
- Analyze Eigenvalues, Eigenvectors and Diagonalization of a Matrix.
- Understand the concept of Positive Definite Matrices, Singular Value Decomposition and its Applications.

**Course Content:**

**Unit 1 : Matrices and Gaussian Elimination**

Introduction, The Geometry of Linear Equations, Gaussian Elimination, Singular Cases, Elimination Matrices, Triangular Factors and Row Exchanges, Inverses and Transposes, Inverse by Gauss -Jordan method.

Self Learning Component: Algebra of Matrices.- **12 Hours**

**Unit 2: Vector Spaces**

Vector Spaces and Subspaces (definitions only) , Linear Independence, Basis and Dimensions, The Four Fundamental Subspaces.

Self Learning Component: Examples of Vector Spaces and Subspaces.- **12 Hours**

**Unit 3: Linear Transformations and Orthogonality**

Linear Transformations, Inner Products and Cosines, Orthogonal Vectors and Subspaces, Cosines and Projections onto Lines, Projections and Least Squares. **12 Hours**

**Unit 4: Orthogonalization, Eigen Values and Eigen Vectors**

Orthogonal Bases, The Gram- Schmidt Orthogonalization, Introduction to Eigen values and Eigenvectors, Properties of Eigenvalues and Eigenvectors, Symmetric Matrices, Diagonalization of a Matrix. **10 Hours**

**Unit 5: Singular Value Decomposition**

Quadratic Forms, Tests for Positive Definiteness, Positive Definite Matrices and Least Squares, Semi Definite Matrices, Singular Value Decomposition, Applications – Curve Fitting, Covariance of matrices (2x2). **10 Hours**

**Tools/ Languages**: SciLab, Python.

**Text Book:**
1: “Linear Algebra and its Applications”, Gilbert Strang, 4th Edition, Thomson Brooks/ Cole, Second Indian Reprint 2007.

**Reference Book(s):**
1: Linear Algebra and its Applications, David .C.Lay, 4th Edition, 2012.
2: Higher Engineering Mathematics, B S Grewal, 44 th Edition, Khanna Publishers, 2020.
3: Practical Linear Algebra, Gerald Farin and Dianne Hansford, 3 rd Edition, CRC Press, Taylor & Francis Group, 2013.

**SCILAB:**
1. Introduction and Gaussian Elimination.
2. Inverse of a Matrix by the Gauss- Jordan Method.
3. The LU Decomposition.
4. The Span of Column Space of a Matrix.
5. The Four Fundamental Subspaces.
6. The Gram-Schmidt Orthogonalization.
7. A=QR factorization.
8. Projections by Least Squarse.
9. Eigen values and Eigen Vectors of a Matrix.

**Reference Book(s):**
1:“Numerical Methods: Principles, Analysis, And Algorithms”, S. Pal, Oxford University Press, 1st Edition, 2009.
2: “Numerical Methods”, E. Balaguruswamy , Tata McGraw - Hill Education, New Delhi 1st Edition, 1999.

## UE20CS251: 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.
- Analyze algorithms using quantitative evaluation.
- Design and implement efficient algorithms for practical and unseen problems.
- Analyze time efficiency over trading space.
- Understand the limits of algorithms and the ways to cope with the limitations.

**Course Content:**

**Unit 1 : Introduction**

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. **-12 Hours**

**Unit 2: Brute Force and Divide-and-Conquer**

Brute Force: Selection Sort, Bubble Sort, Sequential Search, Brute Force String Matching, Exhaustive Search. 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. **- 12 Hours**

**Unit 3: Decrease-and-Conquer & Transform-and-Conquer**

Decrease-and-Conquer: Insertion Sort, Topological Sorting, Algorithms for Generating Combinatorial Objects, Decrease-by-a-Constant-Factor Algorithms. Transform-and-Conquer: Pre-sorting, Heap Sort, Red-Black Trees, 2-3 Trees and B Trees. Problems on - Decrease by a constant factor /constant number. **- 10 Hours**

**Unit 4: Space and Time Tradeoffs & Greedy Technique**

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. **10 Hours**

**Unit 5: Limitations & Coping with the Limitations of Algorithm Power**

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 Language, GCC Compiler.

**Text Book:**
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.

## UE20CS252 : Microprocessor and Computer Architecture (4-0-0-4-4)

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 desirible 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 analyse 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:** UE20CS201- 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.- **12 Hours**

**Unit 2 : Pipeline**

3- Stage Pipelining, 5 - Stage Pipelining, Pipeline Datapath and Control, Data Hazards – Forwarding vs. Stalling, Control Hazards, Branch Prediction Mechanisms and Exceptions, Performance Metrics. - **12 Hours**

**Unit 3 : Memory Hierarchy**

Basics of Caches - Fully Associative, Direct Mapped and Set Associative, Cache Performance, Basic Cache Optimizations. **12 Hours**

**Unit 4 : Flash Storage, Connecting and interfacing devices and Bus Architecture**

Introduction to flash storage, Connecting Processors, Memory and I/O devices, Interfacing I/O Devices to the Processor, Memory and Operating System. - **10 Hours**

**Unit 5 : Advances in Architecture**

Parallel Computing : Introductory concepts and terminology-Flynn’s taxonomy, parallel computing memory architectures, parallel programming models, parallel examples: matrix multiplication, Amdahl’s Law, Gustafson Law, Hardware Multi threading, Multi-Core Architecture.- **10 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.

## UE20CS253: Computer Networks (4-0-0-4-4)

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 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 and study the IPv4 and IPv6 protocols and addressing.
- 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**

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, Introduction to Cloud Computing. **- 10 Hours**

**Unit 2 : Application Layer**

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; 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. **-12 Hours**

**Unit 3 : Transport Layer**

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; 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, TCP Congestion Control. Numerical on TCP congestion control mechanisms-TCP Tahoe, Reno. **- 12 Hours**

**Unit 4 : Network Layer and Internet Protocol**

Overview of Network Layer: Forwarding and routing, The Internet Protocol (IP) IPV4: Datagram Format, Fragmentation, Addressing, NAT; Introduction to Network layer Protocols: DHCP, ICMP; IPv6 Addressing: Address space allocation, Global unicast addresses, Auto configuration, Renumbering, IPv6 Protocol: Packet Format, Transition from IPv4 to IPv6; Introduction to Routing Algorithms: Link State and Distance Vector. - **12 Hours**

**Unit 5 : Link Layer and LAN**

Link layer – Error-Detection and Correction techniques, Parity checks, Internet checksum, cyclic redundancy check, Multiple access protocols: CSMA/CD; 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, Analog vs Digital Signals, Transmission media. Wireless LANs: IEEE 802.11 LAN architecture, 802.11 MAC Protocol, IEEE 802.11 Frame.- **10 Hours**

**Tools/ Languages**: Wireshark, Python.

**Text Book:**

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.

**2:** "Mastering Cloud Computing: Foundations and Applications Programming", Rajkumar Buyya, Christian Vecchiola, S. Thamarai Selvi, Morgan Kaufmann, 2013.

## UE20CS254: 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:**

- The course focuses on fundamental operating system concepts
- The course provides an understanding of various components of operating system.
- The course delves deeper into various algorithms and associated trade-offs for efficient resource management such as process, disk and memory management.
- The course will introduce design principles and trade-offs in the design of Operating Systems.
- The course will also introduce the concepts such as protection and virtualization.

**Course Outcomes:**

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

- Gain extensive knowledge on principles and modules of Operating Systems.
- Understand the design of various algorithms for scheduling and their relative performance.
- Design pieces of the operating systems such as process management, concurrent processes and threads, memory management, virtual memory.
- Use the tools and the interface of the operating system.
- Explore design trade-offs in designing various components of the Operating system.

**Desirable Knowledge**: UE20CS202-Data Structures and its Applications.

**Course Content:**

**Unit1: Introduction and Process Management**

What Operating Systems Do, Operating-System Structure & Operations, Kernel Data Structures, Operating-System Services, Operating System Design and Implementation. **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. - **12 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 Detection.- **12 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. **- 12 Hours**

**Unit 4 : File 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 Mounting, File Sharing. **File-System Implementation:** File-System Structure, File-System Implementation, Directory Implementation, Allocation Methods, Protection. **- 10 Hours**

**Unit 5 : Storage Management and System Protection**

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, Revocation of Access Rights, system calls for set uid- setuid()/chmod() and access control list. Case Study: Windows 10. **- 10 Hours**

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

**Text Book:**
1. "Operating System Concepts", Abraham Silberschatz, Peter Baer Galvin, Greg Gagne 9th Edition, John Wiley & Sons, 2013.
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. "Operating Systems": Three Easy Pieces, RemziArpaci-Dusseau and Andrea Arpaci Dusseau, http://pages.cs.wisc.edu/~remzi/OSTEP/.
3. "Operating Systems", Harvey Deitel, Paul Deitel, David Choffnes, 3rd Edition, Prentice Hall, 2004.
4. "Modern Operating Systems", Andrew S Tanenbaum, 3rd edition, Pearson, 2007.

## UE20CS255: Computer Networks Laboratory (0-0-2-1-1)

Computer Networks Laboratory helps students learn how to put "principles into practice," in a hands-on-networking lab course. These labs will be done in a networked lab setting with Ethernet switches, NICs, desktops and cables. Cisco Packet Tracer, a visual simulation tool designed by Cisco Systems allows students to create network topologies and imitate modern computer networks. Claynet, a network virtualization tool helps students realize large scale networks using routing and switching NFV technology.

**Course Objectives:**

- Learn the basic network utilities to analyse and troubleshoot networks.
- Sniff, filter, and analyze network traffic with Wireshark.
- Understand the header fields and their values of HTTP, TCP, UDP, IP and IEEE 802.11 protocols.
- Study how protocols and layering are represented in packets.
- Apply the client-server model in network socket programming.

**Course Outcomes:**

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

- Be familiar with the network simulator Packet Tracer.
- Assess different solutions for computer networks.
- Understand the functionality of seven layers of OSI reference model.
- Realize network communication skills through socket programming.
- Design and build a small sized wired or wireless LAN.

**Course Contents:**

1. Basic network command-line utilities to monitor/analyse/troubleshoot networks.
2. Understand the building blocks of networks and usage of Cisco Packet Tracer & Claynet network virtualization platform with reference to OSI Layer.
3. Network traffic/protocol analysis using packet sniffer – Wireshark.
4. Network performance effects of HTTP persistent and non-persistent connections.
5. HTTP protocol investigation on HTTP message formats, Cookies, Conditional Get, Authentication and Caching.
6. Setting up a DNS server to understand the functionality and its operations.
7. Extensive network socket programming for applications: Multi-threaded web server, E-mail client, UDP Pinger, Web proxy server.
8. Analyzing and troubleshooting transport and network layer protocols like TCP, UDP and DHCP using Wireshark.
9. Understand IPv4 addressing and static routing (Desktops & Claynet).
10. Understand IPv6 addressing and static routing (Claynet).
11. Understanding ICMP redirect messages, Broadcast storm and TTL expiry in L2 network (Desktops & Claynet)
12. Analyzing and troubleshooting Layer 2 protocols (MAC, Ethernet frames, ARP)
13. Building and testing a small network using Cisco Packet Tracer.
14. Investigation on 802.11 wireless network protocol.

**Tools / Languages**: Claynet, Wireshark, Cisco Packet Tracer.

**Reference Book(s):**

1: Laboratory Manual prepared by Department of CSE, PES University.

## UE20CS256: Microprocessor and Computer Architecture Laboratory (0-0-2-1-1)

This laboratory is to complement the theory course and will help in understanding the low level programming needs of a microprocessor and understanding the performance issues of system architecture in the context of a complex hardware system including pipeline and memory hierarchy. This is understood using suitable tools for Low level programming design, analysis, architecture simulation, implementation, and debugging.

**Course Objectives:**

- Implement assembly language programs and develop strong competencies in contemporary ISAs.
- Develop, edit, compile and debug assembly language programs using present - day simulators.
- Know various addressing modes that are defined in a given instruction set architecture and illustrate how machine language instructions in that architecture identify the operand(s) of each instruction.
- Practice interfacing experiments using various sensors with an Arduino board.
- Learners to imbibe the skills of formulation of a complex problem design a suitable solution using Arduino/ Raspberry Pi processors and demonstrate the end results.

**Course Outcomes:**

- Inculcate the importance of instruction set architecture and their fundamental concepts using assembly language programming.
- Demonstrate editing, compiling, executing and debugging an assembly language program of a contemporary microprocessor.
- Demonstrate the usage of subroutines and recursion supported by the ISA.
- Imbibe strong assembly language programming skills by implementing solutions to problems using simulators.
- Instilling the idea to formulate a complex problem definition, approach to solve the problem, methodology to apply and implement suitable algorithm and check for the final results.

**Course Content:**

1. Introduction to Instruction Set – ARM Processor. Sample programs using Simulator.
2. Programs on ARM/ x86 using Simulator.
3. Programs on ARM/ x86 using Simulator.
4. Demonstration of Arduino Microcontroller with various sensors, Project Discussion.
5. Project Discussion and project Title confirmation.
6. Programs on ARM/ x86 using Simulator.
7. Introduction to 3 stage Pipeline using simulator and Plug gins.
8. Introduction to 5 stages Pipeline (case study: Hazards using Simulator - RAW, WAR, and WAW).
9. Cache Memory Demonstration.
10. Mini Project.
11. Mini Project.
12. Mini Project.
13. Project Evaluation.

**Tools/ Languages:** ARM Simulator, Arduino Microcontroller kit, MIPS pipeline simulator, Para Cache simulator.

**Reference Book(s):**

1: The ARMSim User Guide.
2: 5-stage MIPS simulator online.
3: Para cache simulator online.
4: Laboratory Manual prepared by Department of CSE, PES University.

## UE21MA151D: Bridge Course Mathematics- II (2-0-0-0-2)

This course is designed to provide an understanding of the concepts and techniques of Vector Calculus, Laplace Transforms and Fourier series emphasizing their inter relations and applications to Engineering.

**Course Objectives:**

- To understand Vector Differentiation, Gradient, Divergence and Curl of Scalar and Vector valued functions.
- To understand Vector Integration, Line, Surface & Volume Integration, Green's, Stokes's and Divergence Theorem.
- To understand Laplace Transforms, Inverse Laplace Transforms and associated properties for applying to Engineering Problems.
- To understand the concept of Fourier series.

**Course Outcomes:**

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

- Apply the concepts of Vector Calculus for solving Engineering problems related to Gradient, Divergence, Curl, Directional Derivatives and Laplacian.
- Evaluate Line Integrals, Volume Integrals & Surface Integrals.
- Develop the Mathematical Models of Physical Systems using Differential Equations and solve it using Laplace Transforms.
- Evaluate Fourier series of Periodic Functions.

**Course Content:**

**Unit 1 : Vector differential Calculus**

Introduction to Vector Differentiation. Gradient of a Scalar Function, Directional Derivative, Angle between the Surfaces. Divergence, Curl, related properties and application problems.- **6 Hours**

**Unit 2 : Vector Integral Calculus**

Line, Surface and Volume integrals of Vector point functions, Green's Theorem, Stokes' Theorem and Gauss Divergence Theorem (Statements Only). **-6 Hours**

**Unit 3 : Laplace Transforms**

Definition, Laplace Transforms of Standard Functions. Laplace Transforms of derivatives and Integrals Periodic Function, Unit Step function, Unit Impulse Function and related problems. Self learning component: Proof of Laplace Transform of Periodic Function. **- 5 Hours**

**Unit 4 : Inverse Laplace Transforms**

Definition, Inverse Laplace Transforms of Standard Functions, Various Methods of finding inverse Laplace Transforms, Convolution Theorem. An application of Laplace Transforms to solve Differential Equations.
Self learning component: Proof of Convolution Theorem. **- 5 Hours**

**Unit 5 : Fourier series** Introduction to Fourier series, Dirichlet's conditions, Euler's formulae. Fourier Series of Even and Odd functions, Half Range Fourier Series, Practical Harmonic Analysis.- **6 Hours**

**Text Book:**

1: Higher Engineering Mathematics, B.S.Grewal, Khanna Publishers, New Delhi, 44th edition, 2020.

**Reference books:**

1: Higher Engineering Mathematics, B V Ramana. Tata McGraw-Hill Education, New Delhi, 23rd print, 2015.
2: Advanced Engineering Mathematics, Erwin Kreyszig, Wiley (India), New Delhi, 10th Edition,
2015.
3: Calculus, James Stewart, Cengage Publications, 8th edition, 2013.

## UE19CS301: Database Management System (4-0-0-4-4)

The course provides a theoretical foundation for DBMS and pragmatic applications of DBMS in a real world environment. The course places particular emphasis on the design of relational database systems and covers logical aspects of database design and implementation.

**Course Objectives:**
- Introduce fundamental concepts, terminology and application of relational databases.
- Teach design concepts and creation of relational databases.
- Teach basic and advanced SQL commands.
- Provide overview of database programming and procedural languages.
- Teach OLTP application design and development methodology.

**Course Outcomes:**
At the end of this course, the student will be able to:
- Construct an Entity-Relationship (E-R) model from specifications and transform it to a relational model.
- Design databases and apply normalization techniques.
- Construct queries in SQL and Relational Algebra to perform CRUD (Create, Retrieve, Update and Delete) operations on database.
- Understand and apply the concepts of database programming.
- Design and build OLTP database applications using a RDBMS.

**Course Content:**
**Unit 1 : Introduction to Database and Conceptual Design using ERD:**
Introduction to Databases, DBMS Architecture, Three-Schema Architecture, Data Abstraction and Data Independence, Database Languages and Interfaces, DBMS Modules, Conceptual Model, Conceptual Design using ERD, Entity, Weak Entity, Relationships, Attributes and Keys, Roles and Constraints **-10 Hours**

**Unit 2 : Relational Model:**
Relational Model, Constraints and Database Schemas, ER to Relational Mapping, Relational Algebra, Unary Operations - SELECT and PROJECT, Set Theory Operations, Binary Relational Operations - JOIN, DIVISION, Aggregate Functions and Grouping. - **10 Hours**

**Unit 3 : Database Design**
Functional Dependencies, Inference Rules, Closure, Equivalence, Minimal Cover, Normal Forms Based on Primary Keys (1st, 2nd and 3rd NF), General Definitions of Second and Third Normal Forms, Boyce-Codd Normal Form, Properties of Relational Decompositions, Overview of Higher Normal Forms. . **- 12 Hours**

**Unit 4 : SQL**
SQL Advanced Data Types like CLOB, BLOB, Advanced SQL Queries, Specifying General Constraints as Assertions and Triggers, Views, Database Programming, Procedural language.**- 12 Hours**

**Unit 5 : Design and Implementation of Database Systems**
Database Security, Database transactions, Concurrency control, Locking, Query processing and optimization. NoSQL databases.**- 12 Hours**

**Tools/ Languages**: PostgreSQL 13.3, Erwin.

**Text Book:**
1:"Fundamentals of Database Systems", Ramez Elamsri, Shamkant B Navathe, Pearson, 7th Edition, 2017.

**Reference Book:**
1: "Database System Concepts", Silberschatz, H Korth and S Sudarshan, McGrawHill, 7th Edition, 2019.

## UE19CS302 : 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. **- 12 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). **12 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- **10 Hours**

**Unit 4: Software Testing and Maintenance**
Fundamentals of testing, Test Objectives, Verification and Validation, Terminologies, Testing Types – Blackbox, 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. **12 Hours**

**Unit 5: Ethics, SE in Global Environment, ITSM, ITIL and DevOps**
Software Engineering in a Global Environment, Software Engineering and Hacking, Ethics in Software Engineering. Managed Support Services like ITSM and ITIL. DevOps: Introduction and terminologies, DevOps Pipeline – Continuous Integration, Build, delivery, test, Deployment and Operate and Monitor. DevOps Pillars - Collaboration, Affinity, Tools and Scaling, Exploration to a tool like Jenkins. **10 Hours**

**Tools / Languages**: GitHub, MS Project, Jenkins etc.

**Text Book(s):**
1: “Software Engineering: Principles and Practice", Hans van Vliet, Wiley India, 3rd 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: A Practitioner’s Approach”, Roger S Pressman, McGraw Hill, 6th Edition 2005
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.

## UE19CS303 : Machine Intelligence (4-0-0-4-4)

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 nd its Applicaions 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**: UE19CS203-Statistics for Data Science, UE19MA251-Linear Algebra & its Applications UE19CS251- Design and Analysis of Algorithms.

**Course Content:**

**Unit 1: Introduction, Search Algorithms ,Classification with Decision trees and Performance Metrics**

Introduction to AI and ML , Intelligent Agents and its Types, Machine Learning and its Models, Problem solving by Searching- Uninformed Search and Informed Search Methods, Perspectives and Issues, designing learning systems, Concepts of hypotheses, Version space, inductive bias, Performance metrics-accuracy, precision, recall, sensitivity, specificity, AUC, ROC, Bias Variance decomposition. Decision Trees- Basic algorithm (ID3), Hypothesis search and Inductive bias, Issues in Decision Tree Learning – Overfitting, Solutions to overfitting, dealing with continuous values.

**- 12 Hours**

**Unit 2: Supervised Learning with KNN, ANN, SVM**

Instance-based learning: k-nearest neighbour learning, Artificial Neural networks: Introduction, Perceptrons, Multi-layer networks and back-propagation, Activation Units, Support Vector Machines – margin and maximization, SVM - The primal problem, the Lagrangian dual,SVM – Solution to the Lagrangian dual.- **12 Hours**

**Unit 3: Boosting and Stochastic Models**

Improving performance with Ada-boost, combining weak learners. Bayesian Learning – Bayes theorem, Concept learning, Maximum likelihood,Bayes optimal classifier, Naïve Bayes classifier, Expectation maximization and Gaussian Mixture Models, Hidden Markov Models.- **10 Hours**

**Unit 4: Unsupervised Learning ,Dimensionality Reduction and Genetic Algorithms**

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.Genetic Algorithms – Representing hypothesis, Genetic operators and Fitness function and selection,. Introduction to PSO and application in Single Objective optimization problems. **- 12 Hours**

**Unit 5: Introduction to Deep Learning Techniques**

Overview: Introduction to Deep Learning and Multi Layer neural network, Introduction to Keras. Understanding Various optimizer's , Introduction to Convolution operation and Convolution Neural Network. Understanding Basic Image Augmentation Techniques. Introduction to Recurrent Neural Network. Building a classification model on given data set using CNN and RNN.- **10 Hours**

**Tools / Languages**: Tensorflow 1.15, Keras 2.3.1, Python 3.7.

**Text Books:**
1: "Artificial Intelligence: A Modern Approach (3rd Edition)", Stuart Russel and Peter Norvig, Pearson , 2009.
2: "Machine Learning", Tom Mitchell, McGraw Hill Education (India), 2013.

**Reference Book(s):**
1: "Machine Learning The Art and Science of Algorithms that Make Sense of Data", Peter Flach, Cambridge University Press (2012).
2: "Pattern Recognition and Machine Learning", Christopher Bishop, Springer (2nd Printing), 2011.
3. "Hands-on Machine Learning with Scikit-Learn and TensorFlow", Aurelian Geron, O'REILLY, 1st Edition, 2017.

## UE19CS304: Database Management Systems Laboratory (0-0-2-1-1)

The course provides a practical foundation for developing database applications. The course places particular emphasis on the implementation details of the topics covered under database design and implementation in the theory course.

**Course Objectives:**
- Implement fundamental concepts of application development using relational databases.
- Teach basic and advanced SQL commands.
- Provide overview of database programming and procedural languages.
- Teach OLTP application design and development methodology.

**Course Outcomes:**
At the end of the course the student will be able to:
- Implement applications using RDBMS.
- Master SQL to perform operations in databases.
- Understand and apply the concepts of database programming.
- Design and develop OLTP applications using a RDBMS.

**Course Contents:**
1. DDL – create table with check constraints, Referential integrity constraints, alter, rename, drop, truncate table, Views.
2. DML – Insert, Update, Delete, Transactions - commit, rollback, savepoint..
3. SQL – Joins: inner, outer; Sub queries: correlated and uncorrelated
4. SQL – Aggregate functions.
5. SQL - Set operators: union, intersect, minus..
6. SQL – Creating Functions and Procedures
7. SQL – Creating Triggers.
8. High level programming language using an API.
9. Key value DB – CRUD operations.
10. Document DB - CRUD operations.
11. Columnar DB - CRUD operations.
12. Graph DB - CRUD operations.

**Tools / Languages :** Oracle/MySQL/SQL Server/PostgreSQL

**Reference Book:**
1. Laboratory Manual prepared by Department of Computer Science and Engineering, PES University.

## UE19CS305: Machine Intelligence Laboratory (0-0-2-1-1)

This course provides an introduction to the fundamental methods at the core of modern Machine Learning and Deep Learning. It helps students to learn and Implement various Machine Learning algorithms with the latest tools available.

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

- Understand and Implement Supervised Learning Models
- Learn and Implement Unsupervised Learning Models
- Understand and Implement Dimensionality Reduction Models
- Understand and Implement Bio Inspired Algorithms
- Understand and Implement Deep Learning Models

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

- Apply TensorFlow, Keras and Sk-Learn Tool Kits for Machine Learning Problems.
- Identify and apply suitable Machine Learning Algorithms for a given problem
- Apply Dimensionality reduction techniques on datasets
- Analyze and evaluate the performance of machine learning models
- Apply search techniques for suitable use cases

**Course Contents:**

1. COLAB - some features of colab ; linear regression and logistic regression with a sample data set. How to use the correct version of tensorflow and keras. Etc download as .py file and submission practice.

2. EXPLORE COLAB - some features of colab ; linear regression and logistic regression with a sample data set. How to use the correct version of tensorflow and keras. Etc download as .py file and submission practice.

3. Given a sample dataset with discrete features and class labels as target. Find the entropy of the dataset, entropy of the dataset given an attribute, information gain for the attribute and the attribute to lit the dataset. The chosen attribute has the maximum information gain.

4. Implementing k-nearest neighbours algorithm from scratch for classification problem given a sample training dataset. Find the minkowski distance of a new sample to each sample in the training data given the parameter p(minkowski distance of order p), find the k nearest neighbours given k and predict the class label of the sample.

5. Given a computation graph, perform backpropagation and find the gradient at a given node. The computation graph is a Directed Acyclic graph where each node represents an operation and the edges represent inputs and outputs to this operation. Implement the derivatives of a few simple operations(nodes) and find the gradient at any given node by moving backwards through the graph.(chain-rule).

6. Given two sample datasets where one is linearly separable and the other is not linearly separable perform classification using SVC(Support Vector Classifier) from the scikit-learn package. Visualize the dataset for separability and use appropriate kernels for solving the problem. (Bonus: Use any preprocessing available in scikit-learn package to improve the accuracy)

7. Given a sample dataset implement key features of Ada-Boost algorithm. Given set weights for the samples in the training dataset, construct a decision stump, find the classification error of the stump and weight of the stump. Also find the new sample weights of the training dataset based on the performance of the stump.

8. Given starting probabilities, state-transition matrix, emission probabilities implement the Viterbi algorithm. Given a set of observation sequences, find the most likely sequence of hidden states and also

return its likelihood. Also implement the forward algorithm and find the value at a given time-step for a given set of observation sequences.

9. Implement K-Means clustering algorithm from scratch. Given a sample dataset and number of clusters, find centroids and cluster assignments of all the points by applying k-means.

10. Given a cost function which must be minimized, use Particle Swarm Optimization to find a solution that minimizes the cost function. The cost function gives a real valued score for a given solution. Find the solution that has minimum cost.

11. Introduction to Keras workflow; data loading and preprocessing, building sequential model, training the model, evaluating the same and using it for prediction. Introduction to some basic preprocessing like Normalization, some basic layers like Dense, BatchNorm, Dropout. A dataset is given to train and test the model.

12. Given an image dataset, perform image classification using Convolutional Neural Networks. Understanding and using ImageDataGenerator from keras.preprocessing for generating batches of image data with real-time augmentation. Build the model using keras, perform training and hyperparameter tuning on the given dataset.

13. Given a text dataset, perform text classification using Recurrent Neural Networks. Understanding and using Tokenizer from keras.preprocessing and  Embedding Layer for text encoding. Build the model using keras, perform training and hyperparameter tuning on the given dataset.

**Tools / Languages**: Python(3.7x), sk learn(v0.23), Keras(v2.2.4), Tensorflow(v1.14).

**Reference Book:**
1. Laboratory Manual prepared by Department of Computer Science and Engineering, PES University.

## UE19CS311: 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 Algoorithm.

**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** : UE19CS251 – Design and Analysis of Algorithms.

**Course Content:**

**Unit 1 : Basics of Complexity**

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.- **12 Hours**

**Unit 2 : String Algorithms**

Naïve String Match, Boyer–Moore, Rabin–Karp, String matching with Finite State Automata, and Knuth–Morris–Pratt Algorithms, Suffix Trees - Applications of Suffix Trees, Regular Expression Searches Using Suffix Trees. **- 10 Hours**

**Unit 3 : Maximum Flow, Polynomials and FFT**

Flow Networks, The Ford-Fulkerson method, The Edmonds-Karp algorithm, Maximum Bi-Partite Matching, Polynomials and FFT: Representation of Polynomials, Efficient Polynomial Multiplication, DFT and FFT, Efficient Implementation of FFT.- **12 Hours**

**Unit 4 : 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.- **10 Hours**

**Unit 5 : 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/C++.

**Text Book:**

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.

## UE19CS312 : 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,

- Understand and apply the following concepts pertaining to the analysis of data.
- Data pre-processing, summarization and visualization.
- Model building, evaluation and validation using various techniques.
- 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.
- Use effective data visualization techniques.
- Build regression and time series models for prediction.
- Analyze data to infer underlying patterns and formulate recommendations.
- Be able to build classification models and cluster text data.
- Develop intelligent decision support systems using analytical models .
- Use R to perform data analysis.

**Desirable Knowledge :** UE19CS203 – Statistics for Data Science

**Course Content:**
**Unit 1 : Exploratory Data Analysis and Visualization**
**I**ntroduction, Data Sources, Data Cleaning, Dimensionality Reduction, Data Summarization, Visualization – Graphics and Plotting (R Graphics and relevant packages and maps), Case Studies. **-10 hours**

**Unit 2: Regression Analysis**
Linear regression, Multivariate Regression, Ridge Regression, Lasso Regression, Non-linear regression, Logistic Regression, Case Studies. **- 12 Hours.**

**Unit 3 : Time Series Analysis**
Decomposing a time series, Simple and exponential smoothing, ACF/ PACF, concept of stationarity – its importance, testing for stationarity and converting a non-stationary signal to stationary, AR, MA, ARMA and ARIMA modelling, Case Studies.**- 12 Hours**

**Unit 4 : Recommendation Systems**
Collaborative filtering, Knowledge-based filtering: a mention of decision trees, agglomerative clustering, knn, svm and DBSCAN (including a brief introduction to text classification/ clustering), mining of association rules and evaluation of recommendation systems, Case Studies.**- 12 Hours**

**Unit 5 : Advanced Techniques**:
Latent Semantic Analysis, Discrete Markov Chains, Classification of states in a Markov Chain, Markov Chains with Absorbing States, Expected Duration to Reach a State from Other States, Confounding variables, Case Studies. **-10 Hours**

**Tools & Languages :** Python and R.

**Text Book:**
1. "Business Analytics, The Science of Data-Driven Decision Making", U. Dinesh Kumar, Wiley 2017.
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 Jian Pei, 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.

## UE19CS313 : Internet of Things (4-0-0-4-4)

The Internet of Things is transforming our physical world into a complex and dynamic system of connected devices on an unprecedented scale. In this course, student will explore various components of Internet of things such as Sensors, internetworking and cyber space. In the end they will also be able to design and implement IoT circuits and solutions.

**Course Objectives:**
- Learn the fundamentals of Internet of Things.
- Learn about smart objects and IoT network architecture.
- Compare different Application protocols for Internet of Things.
- Familiarize the Importance of Data Analytics and Security in IoT.
- Apply IoT for real word entities and role of IoT in various domains of Industry.

**Course Outcomes:**
At the end of the course the student will be able to:
- Apply the fundamentals of Internet of Things.
- Build a small low-cost embedded system using Arduino / Raspberry Pi or equivalent boards.
- Evaluate different infrastructure components and network systems and design the basic network for IoT solutions.
- Employ the need of Data Analytics and Security in IoT.
- Analyse applications of Internet of Things in real world scenario Provide lab session for each unit to help gain deeper inside to IoT

**Course Content:**
**Unit 1 : Introduction**
What is IOT? Trends in adoption of IOT, Convergence of IT and IOT, Challenges in IOT, IOT network Architecture and design, Behind New Network Architectures, A Simplified IoT Architecture, The Core IoT, IOT Design Methodology, Domain specific IOT, Functional Stack, Hierarchy of edge, cloud, and Fog..- **10 hours**

**Unit 2 : Smart Objects**
The "Things" in IoT, Sensors, Actuators, and Smart Objects, Sensor, Networks Connecting Smart Objects, Communications Criteria, IoT Access, Technologies, IoT platforms, Programming with Arduino, Programming with Raspberry Pi and Node MCU - **12 hours**

**Unit 3 : IP as IOT Network and Protocols**
The Business Case for IP, the need for Optimization, Optimizing IP for IoT, Application Protocols for IoT, The Transport Layer, IoT Application Transport Methods, Networking technologies, Common protocols, Data Analytics for IoT, Machine Learning, Big Data Analytics Tools and Technology, Edge Streaming Analytics**. - 12 hours**

**Unit 4 : Introduction to the Cloud:**
Cloud Computing Paradigm for Data Collection, Storage and Computing, everything as service and cloud service models, IoT Cloud-Based Services Using the Xively, Nimbits and other platforms.
**Securing IoT:** A Brief History of OT Security, Common Challenges in OT Security, How IT and OT Security Practices and Systems Vary, Formal Risk Analysis Structures: OCTAVE and FAIR, The Phased Application of Security in an Operational Environment, Identify and analyze IoT security, Privacy risks. . - **10 hours**

**Unit 5 : Case Study and advanced Topics**
IoT Physical Devices and Endpoints - Arduino UNO: Introduction to Arduino, Arduino UNO, Fundamentals of Arduino Programming. IoT Physical Devices and Endpoints – Raspberry Pi: Introduction to Raspberry Pi, About the Raspberry Pi Board: Hardware Layout, Operating Systems on Raspberry Pi, Programming Raspberry Pi with Python, Wireless Temperature Monitoring System

Using Pi, Connecting Raspberry Pi via SSH, Accessing Temperature from DS18B20 sensors, Remote access to Raspberry Pi, Introduction to ESP32 Dev Board , Programming ESP32 with Arduino.

Smart Cities: Smart Waste Management, Smart Street Lights, Smart Street Parking, Security Without Surveillance, Connected Vehicles, Module RecapVideo Player; Healthcare: Baby Monitoring, Elderly Monitoring, Mood Enhancing, Disease Treatment and Progression Monitoring, Enhance Adherence, ChallengesVideo Player; Agriculture: Precision Agriculture, Connected Livestock, Food SafetyVideo Player; Manufacturing and Logistics: Smart Manufacturing; Industry 4.0 - Future Scenario Production, Smart Packaging, Smart Label Animation – Thinfilm Printed Electronics**- 12 hours**

**Tools and Languages :** Python/C

**Textbook:**

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 (Cisco Press Indian Reprint). (ISBN: 978-9386873743).

**Reference Books:**
1. "Internet of Things: Architecture and Design Principles", Raj Kamal ,McGraw Hill Education, 1st Edition, 2017.
2. Internet of Things – A hands-on approach , Arshdeep Bahga, Vijay Madisetti ,Universities Press, 2015.
3. Designing the Internet of Things ,Adrian McEwen, Hakin Cassimally Publisher Wiley 2013.
4. Enterprise IoT by Dirk Slama, Frank Puhlmann, Jim Morrish, Rishi M Bhatnagar Publisher: O'Reilly 2015.

## UE19CS314 : Applied Cryptography (4-0-0-4-4)

Cryptography is the science of securing data by using mathematical concepts. Cryptography involves the authentication and verification of data in all domains by applying Cryptographic protocols.

**Course Objectives:**

- Enable to learn the fundamental concepts of cryptography and utilize these techniques in computing systems.
- Discuss about various encryption techniques.
- Understand the concept of public key Cryptography.
- Introduce message authentication and hash function.
- Provide Lab sessions for each unit to help gain deeper insight into Cryptography.

**Course Outcomes:**

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

- Classify the symmetric encryption techniques.
- Illustrate various Public key cryptographic techniques.
- Evaluate the authentication and hash algorithms.
- Discuss authentication applications.

**Course Content:**

**Unit 1 : Classical Ciphers**

Introduction to cryptography, cryptanalysis, and cryptology, Overview of cryptography, Basic Cryptographic primitives, Classical ciphers: substitution cipher – Caesar, Playfair and Hill cipher, Transposition cipher – Rail fence, Columnar and Double columnar, Cryptanalysis of classical ciphers, Introduction to probability, Conditional probability, Law of Total probability, Shannon's theorem, One-time-pad encryption, Limitations of One-Time-Pad. Algebraic structures - Rings, Fields and Groups- **12 Hours**

**Unit 2 : Symmetric Key Cryptography**

Introduction to symmetric key cryptography, Pseudo Random Numbers, Feistel Cipher, S-box and E-box, Initial and Final permutations, Data Encryption Standard (DES), Cryptanalysis and avalanche effect, AES (Advanced Encryption Standard), Key Scheduling, Side channel attacks, Block and Stream ciphers.**-12 Hours**

**Unit 3 : Public Key Cryptography**

Introduction to Public key cryptography, Modes of operation, Prime number, Primitive root, Modular arithmetic, Polynomials, Diffie Hellman Protocol(DH Protocol), Elgamal crypto systems, Prime Factorization, Rivest–Shamir–Adleman cryptosystem (RSA).**-12 Hours**

**Unit 4 : Key management Hashing Techniques**

Key management and distribution (KDC), Birthday attack, Entity authentication methods; password, challenge response, Zero knowledge protocols, message digest algorithm (MD5), One-way function, Collision resistant hash function (CRHF), Secure Hash Algorithm (SHA), Applications. **- 10 Hours**

**Unit 5 : Authentication using Cryptography**

Identification protocols, Digital Signature (DS), Elliptic Curve cryptography-based signature (ECDSA), RSA based signature, Message Authentication Code (MAC), Cipher Block Chaining MAC (CBC-MAC), Different areas where cryptography needs to be applied, gpg, and Quantum resistant cryptography.**- 10 Hours**

**Tools/ Languages** : Seed virtual machine environment, gpg, Python.

**Text Book:**

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

**Reference Book(s):**
**1.** “Cryptography and Network Security” Behrouz A.Foruzan, 3rd Edition, Tata McGraw  Hill, 2017.

## UE19CS315: Fundamentals of 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 Unity3D. This course requires the student to have a desirable knowledge of Data Structures and its Applications.

**Course Objectives:**

- Impart fundamentals of graphical display pipeline system and programming using openGL
- Introduce the use of geometric transformations on graphics objects and their application in composite form and its implementation.
- Introduce Augmented and virtual reality concepts, models and techniques.
- Building a basic application using blender and unity 3D and deploying.

**Course Outcomes:**

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

- Demonstrate the fundamentals of graphical display pipeline system and programming using openGL.
- Use OpenGL for complex 3D graphical visualization and demonstrate its applications.
- Apply techniques and methods of augmenting virtual objects in real space.
- Apply techniques and tools to design an immersive virtual reality experience.
- Use Unity3D to develop complex graphical applications including 3D interactive games.
- Apply graphics in greater depth to more complex courses like Image Processing, Virtual and Augmented Reality, etc...

**Desirable Knowledge** : UE19CS202- Data Structures and its Applications.

**Course Content:**

**Unit 1 : Graphical System and Programming**

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.- **12 Hours**

**Unit 2: 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. **- 12 Hours**

**Unit 3: Introductiont to Augmented Reality**

**I**ntroduction 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. 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.- **12 Hours**

**Unit 4: Introduction to Virtual Reality**

Introduction: What is Virtual Reality, Modern VR Experience. Bird's Eye View: hardware, Software. Physiology of human vision: Eye movement and its implications or VR. Tracking: 2D and 3D orientation, Tracking Position and Orientation, Tracking Attached bodies, 3D Scanning of environments. Interaction: Locomotion, Manipulation, Social Interaction, Additional Interaction Mechanisms. Audio: Physiology of human hearing, Auditory Perception. **10 Hours**

**Unit 5: Visually Realistic 3D Graphics using Blender and Unity3D**

introduction and Installation, using 3D View, controlling lamps, lights, Animating Objects, Modelling with vertices, edges and faces, building a simple boat, modelling organic forms like sea, and terrain, working with camera, Rendering and Compositing Introduction to Unity, Game Objects, Models, Materials and Textures, Terrains, Environments, Lights and Camera, Game1: Amazing Racer, Scripting I, Scripting II, Collision **10 Hours**

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

**Text Books:**
1. Steven M. LaValle. Virtual Reality. Cambridge University Press, 2017, http://vr.cs.uiuc.edu/ (Links to an external site.) (Available online for free)
2. D. Schmalstieg and T. Höllerer. Augmented Reality: Principles and Practice. Addison-Wesley, Boston, 2016, ISBN-13 978-0-32-188357-5

**Reference Books**
**OpenGL**
1: "Interactive Computer Graphics - A top-down approach with shader-based OpenGL", Edward Angel and Dave Shreiner, Pearson Education, Seventh edition, 2015.
2. "OpenGL Programming Guide": Mason Woo, Jackie Neider, Tom Davis, Dave Shrenier: 3rd Edition, openGl version, Addision Wesley, 1999.
**Blender and Unity3D**
3. Blender 3D Basic, Gordon Fisher, PACKT Publishing, 2nd Edition
4: Unity Game Development in 24 Hours ,Geig, Mike. Sams Teach Yourself . Pearson Education, 2014.

***Note: For working with recent versions the course material for UNIT 5 can be substituted with appropriate web content.***

## UE19CS316 : 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:**

- Familiarize the psychology underlying user-interface and usability design guidelines keeping in mind human behavioral and perceptual capabilities and limitations that affect interface design.
- Familiarize 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.
- Apply development methodologies and life cycle models for building user interfaces and prototyping in user interface design and how to test them.
- Familiarize with the impact of usable interfaces in the acceptance and performance utilization of information systems.
- Highlight the importance of working in teams and the role of each member within an interface development phase..

**Course Outcomes:**

- Develop and use a conceptual vocabulary for analysing human interaction with software: affordance, conceptual model, feedback.
- Explain how user-cantered design complements other software process models.
- Use lo-fi (low fidelity) prototyping techniques to gather, and report, user responses.
- Define a user-centred design process that explicitly takes account of the fact that the user is not like the developer or their acquaintances.
- Choose appropriate methods to support the development of a specific UI.

**Course Content:**

**Unit 1: Foundations of HCI**

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. **12 hours**

**Unit 2: Design and Software processes**

nteractive Design basics – 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, rules. Evaluation Techniques – Universal Design **.12 hours**

**Unit 3: Models and Theories**

Cognitive models –Socio-Organizational issues and stake holder requirements –Communication and collaboration models, GUI Design and Aesthetics.**10 Hours**

**Unit 4: Task Analysis**

Task Analysis-Dialog notations and design, Models of the system, Modelling rich interaction, usability engineering, State Charts and Petri nets: case study –Coke machine.**10 Hours**

**Unit 5: Outside the Box**

Groupware, ubiquitous computing, augmented realities, hypertext, multimedia and World Wide Web, Augmented Reality Toolkit practice sessions. **12 Hours**

**Text Book:**

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.

## UE19CS321 : Principles of Programming Languages (4-0-0-4-4)

Principles of Programming Languages is a course to understand the various design choices made by Language designers and the philosophy behind these choices. The course is structured in a way that explores various programming paradigms and their features.

**Course Objectives:**

- Enable students to learn constructs in a language.
- Enable students to design a new construct/ language.
- Enable students to choose appropriate language for real world problem solving, based on the required features.
- Enable students to evaluate various language design features considering the programming paradigm.
- Introduce various paradigms and their support in language design.

**Course Outcomes:**

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

- Choose a particular language for problem solving depending on the application domain.
- Analyze and compare programming language concepts.
- Analyze the implementation issues related to a language design.
- Identify the language design features of any language and evaluate them.
- Apply various language features in solving real-world problem.

**Course Content:**

**Unit 1 : Preliminary Concepts**

Reasons for Studying, Concepts of Programming Languages, Programming Domains Language Evaluation Criteria, Influences on Language Design, Language Categories, Programming Paradigms – Imperative, Object Oriented, Functional Programming, Logic Programming, Programming Language Implementation – Compilation and Virtual Machines, Programming Environments. Names, Binding, Type Checking and Scopes: Names, Variables, Binding of Attributes to Variables, Type Bindings, Type Inferencing, Type Checking, Strong Typing. Case Study: Linux utilities and Program Debuggers for languages such as C, Python. **- 12 Hours**

**Unit 2 : Type Checking and Scopes**

Type Equivalence, Scope, Scope and Lifetime, Referencing Environments. Data types: Introduction, Primitives, Character, User Defined, Array, Associative, Record, Union, Pointer and Reference Types, Design and Implementation Issues Related to These Types, Names, Variable, Concept of Binding, Type Checking, Strong Typing, Type Compatibility, Named Constants, Variable Initialization. Expressions and Statements: Short Circuit Evaluation, Mixed Mode Assignment, Assignment Statements, and Cascading Operators Case Study: Variable scopes and lifetimes in various languages such as C, Java and Python.**- 12 Hours**

**Unit 3 : Control Structures**

Statement Level, Compound Statements, Selection, Iteration, Unconditional Statements, Guarded Commands. Subprograms and Blocks: Fundamentals of Subprograms, Scope and Lifetime of Variable, Static and Dynamic Scope, Design Issues of Subprograms and Operations, Local Referencing Environments, Parameter Passing Methods, Overloaded Subprograms, Generic Subprograms, Parameters that are Subprogram Names.

Case Study: Shallow binding and deep binding of parameters in Function/procedure calls and their implications. Suitable languages can be chosen as demonstration medium.**- 12 Hours**

**Unit 4 : Functions**

Functions: Design Issues for Functions, User Defined Overloaded Operators, Co- Routines and Function Closures. Abstract Data types: Abstractions and Encapsulation, Introduction to Data Abstraction, Design Issues, Object Oriented Concepts with Reference to Java and Python. Case Study: Object oriented concepts demonstration through suitable language. (Java/Python). Clarity on Immutable and Mutable Objects can be imparted with reference to these languages.**- 10 Hours**

**Unit 5 : Exception Handling, Logic programming and Functional Programming**
Exceptions, Specifications, Exception Propagation. Logic Programming Language: Introduction and Overview of Logic Programming, Basic Elements of Prolog, Application of Logic Programming. Functional Programmin g Languages: Introduction, Fundamentals of FPL, Applications of Functional Programming Languages and Exploration of the Features, Comparison of Functional and Imperative Languages.
Case Study: Logic Programming and Functional Programming feature demonstration with suitable languages such as Prolog/LISP and Haskell.- **10 Hours**

**Tools/ Languages**: Various compilers and Debuggers as gcc, g++, Ada, Python, Ruby, Java, Prolog, Haskell, GDB, PDB.

**Text Book:**
1: "Concepts of Programming Languages", Robert W Sebesta, Pearson Education, 10thEdition, 2012.

**Reference Book(s):**
1: "Programming Language Pragmatics", Michael L Scott, Elsevier, 3rd Edition, 2009.
"Programming Languages Design and Implementation", Pratt and Zelkowitz, Prentice
Hall/ Pearson Education, 4th Edition, 2001.

## UE19CS322 : 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.
- Introduce programming tools, 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.
- Apply and differentiate between algorithms for processing Big Data and Normal Algorithms.
- Applying Big Data techniques in real life problems through a group based project.

**Desirable Knowledge** : UE19CS202-Data Structures and its Applications, UE19CS251- 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, Case Study: Google. YARN introduction.- **12 Hours**

**Unit 2 : Big data infrastructures (Compute/Storage)**

Overview of Hadoop Ecosystem, Introduction to sample Big Data Algorithms – matrix multiplication and pagerank Relational operators on Map-reduce, case study: HIVE, Other storage – Hbase/Cassandra.- **12 Hours**

**Unit 3 : In memory computation**

Issues with Hadoop, Spark and Scala/PySpark programming model, Transformations and Actions, Spark SQL, Spark architecture – RDD, DataFrames, Wide and Narrow dependencies, Complexity of Big Data algorithms – Communication Cost complexity model.- **12 Hours**

**Unit 4 : Streaming analysis**

Streaming analytics use cases, Streaming Spark, Kafka – use cases, architecture, Streaming Algorithms – sampling, set membership – Bloom Filters counting, counting unique elements – Flajolet Martin Algorithm.- **10 Hours**

**Unit 5 : Advanced Analytics on Big Data**

Clustering algorithms - k means and Collaborative filtering, Scaling Neural Networks for Big Data, Case Study MLLib.- **10 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.

3: “Hadoop: The Definitive Guide”, Tom White, O’Reilly, 4th edition, 2009.

## UE19CS323 : Graph 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 combinatorics and its application in problem solving. This course requires the student to have a desirable knowledge of problem solving with C and  Data Structures and its Applications.

**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.
- Teach graph theoretical modelling of problems and solution.

**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.
- Apply principles of inclusion and exclusion and generating function to solve problems.
- Solve Recurrence Relations of first and second order

**Desirable Knowledge** :UE19CS151 – Problem Solving with C, UE19CS202 – 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 planer graph. **- 12 Hour**

**Unit 2 : Colouring, Covering and Partitioning**
Chromatic number, Chromatic partitioning, Chromatic polynomial, Matching, Covering, Four Colour problem,  Counting, Register Allocation using graph colouring. **- 12 Hours**

**Unit 3 : Graph Applications**
Shortest Path Problem, Finding Articulation Points, Reliable Communication Network Problem, Chinese Postman Problem, Optimal Assignment Problem, Time Table Problem, Gra ph Databases, Graphs for social network analysis. **- 12 Hours**

**Unit 4 : Inclusion, Exclusion, Generating Functions**
The principle of inclusion and exclusion, Generalizations of the principle, derangements, Rook polynomials. Generating functions: Introductory examples, Definition and examples– calculational techniques, partitions of integers, the exponential generating function, the summation operator. **- 10 Hours**

**Unit 5 : Recurrence Relations**
First Order Linear Recurrence Relation, The Second Order Linear Homogeneous Recurrence Relation with Constant Coefficients, The Non-homogeneous Recurrence Relation, Generating Functions for second order recurrence relations. **- 10 Hours**

**Tools / Languages**: C- Language.

**Text Book:**
1: “Graph Theory: With Application to Engineering and Computer Science”, Narsingh Deo, Prentice Hallof India, 2017.

**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: Latest Web based resources.

## UE19CS324: 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** : UE19CS251 – 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. **- 12 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. **- 12 Hours**

**Unit 3 : Social Computing: Particle Swarm 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.**- 12 Hours**

**Unit 4: Other Foraging Algorithms**

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.

Other Social Algorithms: Glow Worm Algorithm, Bat Algorithm, Fish School Algorithm, Locusts.**10 Hours**

**Unit 5: Immuno computing: Artificial Immune Systems**

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.**10 Hours**

**Tools/Languages**: Matlab.

**Text Book:**
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.

## UE19CS325 : Advanced Computer Networks (4-0-0-4-4)

Advanced Computer Networks course will help students design networks that meet a customer's business and technical goals. This course provides tested processes and tools to help you understand traffic flow, protocol behaviour, and internetworking technologies. Students will be equipped to design enterprise networks that meet a customer's requirements for functionality, capacity, performance, availability, scalability, affordability, security, and manageability. This course requires the student to have a desirable knowledge of Computer Networks.

**Course Objectives:**

- Identify a customer's business and technical requirements for a network design.
- Analyze the existing network and network traffic caused by applications and protocols.
- Develop a topology for a network design.
- Select the right switching, routing and management protocols for network design customer.
- Select technologies and devices for campus and enterprise networks.

**Course Outcomes:**

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

- Gather a list of customer's most important business and technical goals.
- Design an enhanced network which improves performance of existing network.
- Increase the probability of meeting a customer's goals for scalability, adaptability, and performance.
- Recommend the best switch and router products to the customer.
- Design a campus network or WAN design with all the customer requirements.

**Desirable Knowledge** : UE19CS253 – Computer Networks.

**Course Content:**

**Unit 1: Identifying Your Customer's Needs and Goals**

Analyzing Business Goals and Constraints:Using a Top-Down Network Design Methodology, Analyzing Business Goals, Analyzing Business Constraints, Business Goals Checklist. Analyzing Technical Goals and Tradeoffs: Scalability, Availability, Network Performance, Security, Manageability, Usability, Adaptability, Affordability, Making Network Design Tradeoffs, Technical Goals Checklist. **- 10 Hours**

**Unit 2: Characterizing the Existing Internetwork and Network Traffic**

Characterizing the Network Infrastructure, Checking the Health of the Existing Internetwork, Network Health Checklist.Characterizing Network Traffic: Characterizing Traffic Flow, Characterizing Traffic Load, Characterizing Traffic Behaviour, Characterizing Quality of Service Requirements, Network Traffic Checklist. **- 12 Hours**

**Unit 3 : Logical Network Design**

Designing a Network Topology: Hierarchical Network Design, Redundant Network Design Topologies, Modular Network Design, Designing a Campus Network Design Topology, Designing the Enterprise Edge Topology, Secure Network Design Topologies. Designing Models for Addressing and Numbering: Guidelines for Assigning Network Layer Addresses, Using a Hierarchical Model for Assigning Addresses, Designing a Model for Naming.**12 Hours**

**Unit 4: Selecting Switching and Routing Protocol**

Selecting Switching and Routing Protocols: Making Decisions as Part of the Top-Down Network Design Process, Selecting Switching Protocols, Selecting Routing Protocols, IP Routing. Developing Network Management Strategies: Network Management Design, Network Management Architectures, Selecting Network Management Tools and Protocols.**- 12 Hours**

**Unit 5 : Physical Network Design**

Selecting Technologies and Devices for Campus Networks: LAN Cabling Plant Design, LAN Technologies, Selecting Internetworking Devices for a Campus Network Design, Example of a Campus Network Design. Selecting Technologies and Devices for Enterprise Networks: Remote-Access technologies, Selecting

Remote-Access Devices for an Enterprise Network Design, WAN technologies, Example of a WAN design.
**10 Hours**

**Tools / Languages**: Claynet, Cisco Packet Tracer.

**Text Book:**
1: “Top-Down Network Design”, Priscilla Oppenheimer, Cisco Press, 3rd Edition, 2011.

**Reference Book(s):**
1: “Modelling and Tools for Network Simulation”, KlausWehrle, Mesut Günes and James Gross, Springer, 2010.
2:“Networking Systems Design and Development”, Lee Chao, CRC Press, 2009.
3:“The Practice of System and Network Administration”, Thomas A. Limoncelli, Christina J. Hogan and Strata R. Chalup, Addison-Wesley Professional, 2016.

## UE19CS326 : Computer Network Security (4-0-0-4-4)

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 with case studies.

**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 internet.
- Master in wireless network security systems in depth, and perform effective network management.

**Desirable Knowledge :** UE19CS253 – Computer Networks

**Course Contents:**

**Unit 1 : Introduction and Packet Analysis**

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, Hybrid approach, Endianness.**10 Hours**

**Unit-2 : OSI Protocol Attacks**

Attacks on the TCP protocols: Attacks on the TCP protocols: Introduction, 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**. MAC layer and attacks:** Introduction, The MAC layer, ARP protocol, ARP cache poisoning attacks, MITM using ARP cache poisoning, Demo, Countermeasure.

**Network layer: IP, ICMP and attacks**: Introduction, 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, Case Study – **1 – 12 Hours**

**Unit-3 : DNS Attacks and Firewall**

**DNS Attacks:** Introduction, DNS hierarchy, zones and servers, DNS query process, Experiment Setup, 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. **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. **- 12 Hours**

**Unit 4 :IDS, IPS and Virtual Private Networks**

**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, Intrusion prevention system **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. **-12 Hours**

**Unit 5 : Network Management and Wireless Network Security**
**Network Management:** IT security management overview and Risk assessment, IT security controls, Plans, and procedures, Physical and infrastructure security, Human Resources security, Security auditing. **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. **10 hours**

**Note:** Hands-on lab sessions, assignments and case study discussions will be part of few lecture hours.

**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.

## UE19CS351: 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**: UE19CS202- Data Structures and its Application, UE19CS205- Automata Formal Languages & Logic.

**Course Content:**

**Unit 1 : Compilers**

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.- **10 Hours**

**Unit 2 : Syntax Analysis**

The role of the parser, Syntax Error Handling, Error-Recovery Strategies. Top-down parsing: Recursive Descent Parser (RDP) with Backtracking, LL(1) Parser. Bottom-up parsing: Shift-Reduce Parsing, LR (0), SLR, viable prefixes, CLR, LALR.- **12 Hours**

**Unit 3 : Syntax-Directed Translation**

Syntax-directed definitions, Evaluation orders for SDD's, Applications of Syntax-Directed Translation, and Syntax-directed Translation Schemes – Postfix Translation Schemes. Parser Stack Implementation: 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. - **12 Hours**

**Unit 4 : Intermediate-Code Generation**

Variants of Syntax Trees – Directed Acyclic Graphs for Expressions, Three-Address Code – Addresses and Instructions, Quadruples, Triples, Indirect Triples, SSA Form, Control Flow Graph. Machine Independent Optimization: Different Optimizations, Optimization of Basic Blocks. Data Flow Analysis: Live-variable analysis, Next-use algorithm.- **12 Hours**

**Unit 5 : Run-Time Environments**

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. - **10 Hours**

**Tools/Languages**: Lex and Yaac.

**Text Book:**
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.

## UE19CS352: Cloud Computing (4-0-0-4-4)

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.
- Introduce and design distributed systems for scalability.
- 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.
- Demonstrate design tradeoffs while designing cloud applications.

**Desirable Knowledge**: UE19CS253- Computer Networks, UE19CS254- 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 S ub model, SaaS Programming model – Microservices and differences with the traditional monolithic model; challenges of migrating monolithic applications.**-12 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.**-12 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.**-12 Hours**

**Unit 4 : Cloud Controller**
Master-slave v/s p2p models, Resource allocation, Scheduling algorithms, Cluster coordination – consensus, Fault Tolerance - faults and partial failures, Unreliable communication, Cluster coordination - leader election, distributed locking, Case Study: Zookeeper - distributed consensus infrastructure.**- 10 Hours**

**Unit 5 : Performance, Scalability and Security in Cloud**
Scaling computation - reverse proxies, Scaling computation - hybrid cloud and cloud bursting, Multitenancy, Multitenant databases, Failure detection - checkpointing and application recovery, Cloud security requirements - physical/virtual security, Risk management, security design patterns, Security architecture,

legal and regulatory issues, Authentication in the cloud: Keystone, Cloud Threats – DoS, Economic Denial of Sustainability.**-10 Hours**

**Tools/Languages**: Amazon AWS (or equivalent), Docker, Kubernetes, github, NoSQL databases, Flask.

**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.

## UE19CS353: Object Oriented Analysis and Design using Java(4-0-0-4-4)

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:**

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

**Course Outcomes:**

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

- Use the concepts of classes and objects of object-oriented programming in UML to model a complex system.
- Construct static models using 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 an.d deployment models.
- Use GRASP and SOLID principles in the design of software application.
- Apply different software design patterns for variety of application scenarios.

**Course Content:**

**Unit 1: Introduction to Object Orientated Programming**

Introduction to object-oriented concepts, Object Based Programming: JVM, Abstraction, Encapsulation, Composition, Class Attributes, Behaviour, Objects, and Methods. Interface and Implementation: Role of Constructors and Destructors, Garbage Collector, Parameter Passing, Value Type and Reference Type, Overloading of Methods Model, Java Recursion. 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. **- 12 Hours**

**Unit 2: Advanced OO, Object Oriented Analysis and Static Models and Diagrams**

Composition: Flexibility of Composition over Inheritance.Concept of Serialization. Introduction to Generics and Collections: Generic Programming Concepts Recap of SDLC and OOAD as part of SDLC, Recap of Architecture & design characteristics/techniques, OOA: Requirements, Modelling and Analysis, Introduction to UML. Use Case Modelling: Use Cases Diagrams. Class Modelling: UML Class Diagrams, OO relationships, CRC Diagrams. UML Component Diagrams, UML Deployment Diagrams. **– 12 Hours**

**Unit 3: Dynamic Models, Diagrams and Architecture design and principles**

Activity Modelling: UML Activity Diagrams and Modelling, Guidelines. Behaviour Modelling: UML State Machine Diagrams and Models, Advanced State Models, OO Development process, System Design and Frameworks, Architectural patterns, MVC architectural pattern, analysing a simple program, designing the system and subsystems. GRASP and its application to Object Design, Creator, Information Expert, Low Coupling, Controller, High Cohesion, Polymorphism, Pure Fabrication, Indirection and Protected Variations. Sample implementation in Java to demonstrate few of GRASP Principles.**- 12 Hours**

**Unit 4: OO Design Principles and Sample Implementation of Patterns in Java**

SOLID: Single Responsibility, Open-Closed, Liskov Substitution, Interface Segregation, Dependency Inversion and their Implementation. Introduction to Design Patterns. Selection or usage of a design pattern. **Design Patterns Theory and Implementation in Java:** Creational Patterns (~2 patterns of Singleton, Builder, Factory, Abstract Factory) **– 10 Hours**

**Unit 5: OO Design Patterns & Anti-Patterns with Sample implementation in Java**
Structural Patterns (~2 of Adapter, Façade, Proxy patterns), Behavioural Patterns (~2 of Chain of Responsibility, Command, Interpreter Patterns). Anti-patterns - Architecture and Design anti-patterns (~2 patterns). **– 10 Hours**

**Tools / Languages**: Star UML, Java.

**Text Book:**
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.

## UE19CS354: Cloud Computing Laboratory (0-0-2-1-1)

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.

**Course Objectives:**

- Introduce working with a public cloud and the terminology associated with cloud services.
- Introduce different communication mechanisms.
- Introduce cloud native programming models.
- Introduce deployment tools on the cloud like Docker and Kubernetes.

**Course Outcomes:**

- At the end of this course, the student will be able to:
- Work with a public cloud and work on.
- Build and deploy a sample application on the cloud.
- Demonstrate the use of tools in building cloud applications.
- Demonstrate their learning through practical hands-on assignments.

**Course Content:**

1. Connecting to Public cloud and creating a VM on the public cloud; setup the firewalls to allow connection to the VM. Auto shutdown.
2. Setup a web server on the VM in the public cloud. Create a web page on the web server and access it from your desktop. Monitor cloud usage.
3. Introduction to REST apis.
4. Introduction to message queues for communication.
5. Docker images; deploying docker containers.
6. Dockerizing the micro service pap, network setup.
7. Load balancers on the cloud.
8. Introduction to kubernetes setup and sample deployment
9. Storing data persistently.
10. Zookeeper.

**Tools/Languages** : Amazon AWS, Docker, Kubernetes, Github, NoSQL, databases, Flask.

**Reference Book(s):**

1: “Docker in Action”, Jeff Nickoloff, Manning Publications, 2017.
2: “Cloud Native DevOps with Kubernetes”, John Arundel and Justin Domingus, OReilly, 2019.
3: Laboratory Manual prepared by the Department of Computer Science and Engineering, PES University.

## UE19CS355: Object Oriented Analysis and Design with using Java Laboratory (0-0-2-1-1)

In this laboratory course, the student will perform analysis on a given domain and come up with an Object-Oriented Design (OOD). Various techniques will be adopted which are commonly used in analysis and design phases in the software industry. Usage of some design tools using Object oriented modeling and design approach to carry out the design for the projects gives a hands on experience. Implementation of the projects using Java gives them overall experience of analysis, design and implementation. To familiarize with OO concepts, some of the real world scenarios are implemented using java.

**Course Objectives:**

- Students to learn object oriented programming concepts and object-oriented analysis and modelling.
- Students to learn to implement object oriented concepts with Java.
- Acquaint students with static and dynamic models used in UML.
- Students learn to the importance of GRASP and SOLID design principles**.**
- Expose students to adopt OO concepts for the real world scenarios.

**Course Outcomes:**

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

- Implement OO Concepts using Java.
- Analyze and Design using OOAD approach
- Design the given scenarios using UML
- Apply Design principles to get an improved code
- Work on real time projects applying OOAD approach using Java

**Course Content:**

1. Experiment covering Java concepts which includes classes, instances, constructors & destructors, Parameter passing, method overloading, array & exception handling. Implementation of Stack or Queue (also PQ) data structure covering the above concepts.
2. Experiment covering Java Concepts which include recursion & multithreading. Implementation of Merge sort algorithm (Divide & Conquer Technique) and find the complexity with varying large inputs & plot the same (Use like XGraph).
3. Implementing serialize & de-serialize of files from one node to another node.
4. Implementation of the project using java. As part of the project start with Design of use case diagram, class diagram, system architecture of the chosen project.
5. Implementation of a few GRASP design principles for the given case study.
6. Implementation of the project using java. As continuation of the project design, draw behavioral UML diagrams as per the requirement.
7. Implementation of dependency inversion design principles for the given problem statement.
8. Implementation of Design Patterns specifically Singleton & abstract factory pattern.
9. Implementation of Design Patterns, Structural Pattern (i.e, Proxy).
10. Week 10 to 12 – Implementation and Testing of the project.
11. Week 13 - Evaluation week.

**Tools / Languages**: Eclipse, Star UML

**Reference Book(s):**

1: Laboratory Manual prepared by the Department of Computer Science and Engineering, PES University.

## UE19CS331: Generic Programming (4-0-0-4-4)

Generic programming is one of the most popular paradigms of programming. This provides a type agnostic, flexible way of developing data structures and algorithms. This also provides programming at compile time in compile time artefacts. This course requires the student to have a desirable knowledge of problem Solving with C, Data Structures and its Applications and Design and Analysis of Algorithm.

**Course Objectives:**
- Understand rationale behind generic programming - appreciate generic functions.
- Understand rationale behind generic classes.
- Understand and appreciate STL philosophy.
- Understand and appreciate programming at compile time.
- Understand and appreciate generics in Java and C#.

**Course Outcomes:**
At the end of this course, the student will be able to:
- Develop programs using generic functions.
- Develop programs using generic classes.
- Develop programs using STL.
- Develop programs using compile time artefacts.
- Develop simple programs using Java and C# generics.

**Desirable Knowledge** :UE19CS151- Problem solving with C, UE19CS202- Data Structures and its Applications, UE19CS251- Design and Analysis of Algorithms.

**Course Content:**
**Unit 1 : Template Functions**
Definition, Instantiation - Implicit and Explicit, Specialization,Type and Non-Type Template Parameter.- **12 Hours**

**Unit 2 : Template Class**
Instantiation, Templates and Static Members, Templates andInheritance, Templates and Composition, Templates and Friends, Template Member Functions, Dependent Type, Default Template Parameter, Nested Templates.-**12 Hours**

**Unit 3 : STL**
STL Philosophy, Efficiency of Algorithms, Separation of Behaviour from Container Classes, Functor and Iterator, Iterator Hierarchy, Adaptors, Examples of Containers and Algorithms, Traits and Policies.- **12 Hours**

**Unit 4 : Template Meta - Programming Overview**
Compile-Time Programming Nature and Limitations of Template Meta-Programming, Values, Functions, Branching, Recursion, Compile-Time "If' Conventions for "Structured" Template Meta Programming.- **10 Hours**

**Unit 5 : Generics in Java and C#**
Generic Methods, Constructors, Type Inference, Bounded TypeParameters, Subtyping, Wildcards, Type Erasure, Overview of Generic Collection Classes, Generics in C#, Generic Constraints, Generics and Casting, Inheritance and Generics, Generic Methods, Generic Delegates, Generics and Reflection. **10 Hours**

**Tools / Languages: C**++, Java, C#.

**Text Book:**
1: "STL Tutorial and Reference", Musser, Derge and Saini, 2nd Edition, Addison-Wesley, 2001.

**Reference Book(s):**

1: “C++ Primer”, Lippman, Addison-Wesley, 2013.
2: “A tour of C++”, Bjarne Stroustrup, Addison-Wesley, 2013.
3: “Templates: The Complete Guide”, David Vandevoorde, Nicolai M Josuttis, Addison-Wesley, 2002.
4: “Java Tutorials”, Online Reference Link - https://docs.oracle.com/javase/tutor.
5: MSDN for C# generics.

## UE19CS332: Algorithms for Intelligence Web and Information Retrieval (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 the recommendation algorithms and Clustering Algorithms and their working.
- Understand the different web Applications.

**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 Machine Learning techniques in Information Retrieval Systems and Web Applications.
- Implement different recommendation algorithms and clustering algorithms.
- Design different intelligent web applications.

**Desirable Knowledge** : UE19CS251 - Design and Analysis of Algorithms.

**Course Content:**

**Unit 1: Introduction to Information Retrieval and Web Scale Computing**

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. Introduction to Web and Intelligent Web Applications. Examples of Intelligent Web Applications.- **10 Hours**

**Unit 2: Indexing and Vector Space Model, Evaluation of IR**

Algorithms for Indexing and IndexCompression, Vector Space Model for Scoring, tf-idf and Variants, Efficient Scoring and Ranking, Performance Measurement, Relevance Feedback, Query Expansion, Other IR Models.- **10 Hours**

**Unit 3 : Web Applications and Search Algorithms**

Web Search Basics, Economic Model of Web Search, Lucene as a Search Engine, Other search engines like Solr, Everything, Google. Improving Search Results, Link Analysis, The Page Rank Algorithm, Other Search Algorithms, Scalability Issues in Search. Search User Experience, Web Crawling and Indices, Link Analysis, Building a Complete Search System. **- 12 Hours**

**Unit 4: Recommendation Algorithms**

Instance based Learning, Introduction to Recommender Systems- Goals, Basic Models, Domain Specific Challenges in Recommender Systems, Neighborhood based collaborative filtering, Model based collaborative filtering, Content based Recommender System, Constraint based recommender systems. Analysis of the Paper “Cutting Edge Collaborative Recommender Algorithms “by Balazs Hidasi (from the book Collaborative Recommendations, Algorithms Practical Challenges and Applications).- **12 Hours**

**Unit 5: Design of Intelligent Web Application**

Design of an Intelligent Web Application, User Requirements, Selecting Algorithms, Data Design, Design for Performance, Architecture of an Intelligent Web Application, Implementation Issues, Summary and Conclusion. Analysis of 3 Research papers (from Artificial Intelligence: Methods in Intelligent Algorithms Proceedings of 8th Computer Science On-line Conference 2019, Vol. 2. WI 2020: Web Intelligence, WI-IAT 2020, the 19TH IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, Melbourne, Australia). Analysis of the paper “Context Aware Recommendations “by

Yong Zheng and Bamshad Mobasher (from the book Collaborative Recommendations , Algorithms Practical Challenges and Applications).- **12 Hours**

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

**Text Book:**
1: "Introduction to Information Retrieval", Christopher D. Manning, Prabhakar Raghavan, Hinrich Schutze, ISBN: 9781107666399, Cambridge University Press, 2009.

**Reference Book(s):**
1: "Algorithms of the Intelligent Web", Haralambos Marmanis, Dmitry Babenko,Manning Publishers, 2011.
2: "Recommender Systems – The Text Book ", Charu C. Agarrwal, ISBN- 978-3-319-29657-9, Springer 2016.
3:"Collaborative Recommendations, Algorithms Practical Challenges and Applications", Shlomo Berkovsky, Ivan Cantador, Domonkos Tikk, ISBN – 978-981-3275-34-8, World Scientific 2019.

## UE19CS333: Image Processing and Computer Vision (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 image and computer vision fundamentals
- Gain an insight to image enhancement and transformation techniques
- Gain insight into various morphological operations and segmentation techniques.
- Learn different feature extraction techniques and understand fundamentals of motion
- Understand Multiview geometry to estimate depth in images.

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

- Apply image fundamentals for various applications.
- Apply image enhancement techniques in both the spatial and frequency (Fourier) domains.
- Design and evaluate methodologies for image segmentation.
- Apply feature extraction techniques to estimate motion.
- Conduct an independent study for depth estimation.

**Desirable Knowledge :** UE19CS251 – Design and Analysis of Algorithms.

**Course Content:**
**Unit 1: Image Basics and Colour Model**
What is Digital Image Processing, examples of fields that use DIP, Fundamental Steps in Digital Image Processing, Image Formation- Geometric Image Formation, Photometric image formation, Sampling and Quantization, Representation of Digital Images, Spatial and Gray-level Resolution, Zooming and Shrinking Digital Images, Basic Relationships Between Pixels, Linear and Nonlinear Operations, Colour Models: Pseudocolour Image Processing, Basics of Full-Colour Image Processing **- 12 Hours**

**Unit 2: Image Enhancement**
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, 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: Slant Transform, Haar Transform and KL Transform.- **12 Hours**

**Unit 3: Morphological Image Processing and Segmentation:**
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, semantic segmentation. **- 12 Hours**

**Unit 4: Feature Extraction and Motion Estimation**
Local feature detectors and descriptors: Hessian corner detection, Harris corner detection, LOG detector, SIFT, HOG, Translational alignment, parametric motion, Optical flow, layered motion, Pose Estimation.- **10 Hours**

**Unit 5 : Depth Estimation**
Stereo and Multiview Geometry, Epipolar Geometry, sparse and dense correspondence, local methods, Multiview stereo, Monocular depth estimation. **- 10 Hours**

**Tools / Languages**: Matlab, Python.

**Text Book:**
1: “Digital Image Processing”, Rafael C Gonzalez and Richard E. WoodsPrentice Hall, 4th Edition, 2018.
2: “Computer Vision : Algorithms and Applications”, Richard Szeliski, 2$^{nd}$ Edition, 2021.

**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

## UE19CS334: 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 central themes, learning problem and the problem solving approaches used 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.
- Learn how lexical and distributional semantics can be used for semantic disambiguation 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 central themes, central problem being solved in NLP and the learning approaches used in solving them.
- Understand various sequence labelling approaches and applications in NLP.
- Understand how syntactic ambiguity removal can contribute in overall disambiguation process.
- Analyze and Apply comfortably appropriate branch of semantics depending on the problem being solved.
- Design and implement neural language model, NLP applications using neural techniques and utilize various transfer learning approaches in NLP.

**Desirable Knowledge :** UE19CS303-Machine Intelligence.

**Course Content:**

**Unit 1: Introduction**

Introduction, Knowledge in Language Processing, Types of ambiguity in natural language processing, Models and Algorithms,**Text normalization:**Content and Function words, type vs. token, word tokenization and 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. Introduction to NLTK, Spacy , Gensim and some important Libraries for NLP.

**12 hours**

**Unit 2: Language Models and Semantics(Lexical and Vector)**

N-grams,n-gram language model, smoothing, discounting and back-off, Kneser-Ney smoothing, interpolation, perplexity as an evaluation measure .Word senses and relations between word senses, WordNet: A Database of Lexical Relations; Word sense disambiguation : supervised word sense disambiguation, WSD : dictionary and thesaurus methods, semi-supervised WSD, unsupervised word sense induction Semantic relatedness based on thesaurus like WordNet : Resnik similarity, Lin similarity, Jiang-Conrath distance, Extended Gloss overlap and Extended Lesk method. 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 and Glove, Improving Word2vec: FastText, Limitation of distributional methods.

**12 Hours**

**Unit 3: Handling sequences of text**

Sequence labelling: Sequence labelling as classification, sequence labelling as structure prediction, Viterbi algorithm and Hidden Markov Model, POS Tagging example, POS Tagging using discriminative models i.e. Maximum Entropy Markov Model ( MEMM). Discriminative Sequence labelling with features-Conditional

Random Field. Other sequence labelling applications – Named Entity Recognition: practical NER architectures.

**12 Hours**

**Unit 4: Parsing - Disambiguating Structure**

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

**10 Hours**

**Unit 5: Natural Language Generation and Neural Network Methods**

**Neural Sequence labelling** - Recurrent Neural network language model for POS tagging. **Convolutional Neural Network for text:** word level and character level language model with CNN and Sentiment analysis. **Dialogue** : Sequence over utterances, chatbots – rule and corpus based. **Neural dialogue agents :** Seq2Seq Chatbots using encoder-decoder architecture, attention models. **Neural Question answering** - Information Retrieval based factoid question answering, knowledge based question answering, Neural Question answering. **Transfer learning in modern NLP -** BERT, ELMo, GPT, ULMfit.

**10 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, 2nd edition paperback,2013. 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.

## UE19CS335 : 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's 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** : UE19CS202-Data Structures and its Applications.

**Course Content:**
**Unit 1 : Blockchain Introduction**
Key Blockchain Concepts, Nodes, Cryptocurrency, tokens, Public Ledger, Peer to peer Network, Types of Blockchain, Permissioned Blockchain model, Permission-less Blockchain model, Blockchain Construction . **-10 Hours**

**Unit 2 : Cryptography**
Machines that encrypted data in the past, Modern encryption, Private and public keys, Hash functions, From blocks to hashes, Hash Pointer, Markle tree, Ledgers, Transactions and trade, The public witness, Computers that witness, Distributed Consensus, Smart contract design, Bitcoin Blockchain Network.**- 12 Hours**

**Unit 3 : The structure of the network: consensus algorithm**
Proof of Work, Proof of Stake, Delegated Proof of Stake, Proof of Authority, Proof of Elapsed Time, Proof of Capacity, Proof of Space, Proof of Burn, RAFT, PAXOS, Byzantine Fault Tolerance System, PBFT. **- 12 Hours**

**Unit 4 : Second generation applications of Blockchain technology**
Smart contracts: origins and how they function, Creating and deploying smart contracts, Tokens, Token standards, Second generation tokens Decentralized applications, How are DApps constructed?, Decentralized Autonomous Organizations (DAOs), Hyperledger Fabric: Blockchain-as-a-service (BaaS), Architecture and core components, Hyperledger fabric model, Working on hyper ledger and transaction processing.**- 12 Hours**

**Unit 5 : Blockchain Security**
Blockchain vulnerabilities, Smart contract vulnerabilities, Blockchain on CIA security triad, Blockchain based DNS security platform, deploying Blockchain based DDOS protection.**-10 Hours**

**Tools / Languages**: Claynet, Python.

**Text Book:**
1: "Introduction to Blockchain Technology", Tiana Laurence, 1st edition, Van Haren Publishing, 2019.

**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 (19 July 2016).

## UE19CS336: Digital Cyber Forensics (4-0-0-4-4)

Cyber Forensics course provides a deep understanding of the techniques to gather, protect and report the digital confirmations.

**Course Objectives:**

- The Cyber Security issues, the Digital Forensics process and the Hard disk structure.
- The process of Data Acquisition and the structure of FAT and NTFS file system on Windows operating system.
- The Structure of Linux File system (EXT3/EXT4) and the file carving process.
- The Android Mobile device forensics and Multimedia Steganography procedures.
- The procedure for Email Forensics Analysis and Final report writing as per the court of law.

**Course Outcomes:**

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

- Understand the phases in Forensic Investigation process and make out the internal structure of HDD and booting process.
- Use SleuthKit Library and Make an image of the Evidence with various open source tools and gain knowledge on FAT and NTFS file systems.
- Analyse the Unix/Linux File systems with exercises and do file carving using open source tools.
- Perform the Mobile device forensics and do Steganalysis for Multimedia forensics.
- Do Email forensics and know how to write a good report to be submitted to the court of law.

**Course Content:**

**Unit 1: Introduction to Forensic Process**

Introduction to computer forensics, Forensics Investigation Process, Forensic Protocol for Evidence Acquisition, Digital Evidences, Types of computer forensics, Challenges in computer forensics, Understanding the Hard disks and File systems-HDD, SSD, Physical structure and Logical Structure of Hard Disk, Tracks, Sector, Cluster, Disk Partitions and Boot process, Open source tools.**10 Hours**

**Unit 2: Data Acquisition and Windows File system Forensic Analysis**

Building a Forensics Work station with The Sleuth Kit, Case Study-Data Acquisition – Imaging using Access Data FTK Imager and Encase, Recovering files from the images using Encase, Examining FAT File system, Examining NTFS File system, Case study - NTFS Timestamp Analysis, Autopsy Tool Hands-on.**12 Hours**

**Unit 3 : Linux File system Analysis andFile carving**

Unix/Linux file systems (Ext2/Ext3), Unix/Linux Forensic Investigation: Unix/Linux forensics, investigation steps and technologies, Case Study: Memory Acquisition of Linux System using LiME , Principles of file carving, Header/Footer carving, Bitfragment Gap carving, Case Study- Image File Foremost File Carving tool.**10 Hours**

**Unit 4: Android Mobile device Forensics and Multimedia Forensics**

Mobile Device Forensic Investigation, Storage Location, Acquisition Methods, Data Analysis of Facebook, Whatsapp, Case study using Android Virtual Device, Steganography Techniques and Tools, Steganalysis Techniques and Tools, Case study-Steganalysis using OpenStego, Anti Forensics Practices-Data Wiping and Shredding, Trail Obfuscation, Encryption, Data Hiding, Case Study-Anti forensic detection using Stegdetect.**12 Hours**

**Unit 5 : Email Forensics and Investigative reports and Legal Acceptance**

Email Forensics, Recovering emails, Email Header Analysis, Case Study-e-Discovery from Enron Corpus, reparation work for report Writing, Structure of the report, Characteristics of a good report, Document design and good writing practices, Legal Acceptance, Case Study – Legal Acceptance in Autopsy tool, Incident Response process.**12 Hours**

**Tools / Languages** : Open source tools on Forensics.

**Text Books:**
1: “Introductory Computer Forensics-A Hands-on practical Approach”, by Xiaodong Lin, Springer, 2018.
2: “Practical Cyber Forensics- An Incident-Based Approach to Forensic Investigations”, by Niranjan Reddy, A Press, 2019.

**Reference Book(s):**
1: “Digital Forensics Workbook_-Hands-on Activities in Digital Forensics”, by Michael K Robinson ,CreateSpace Independent Publishing Platform, 2015.

## UE19CS337: Hardware Accelerated Computing (4-0-0-4-4)

As Moore's Law slows down, there is increasing interest in the use of hardware accelerators, especially for big data AI/ML workloads. Examples include Google's Tensor Processing Units and the recent Inferentia from Amazon. This course provides a big picture overview of the hardware accelerator landscape and clear, in-depth and insightful coverage of design techniques for hardware accelerators (like those mentioned above) and the state-of-the-art in programmable logic based implementation of hardware acceleration. This course requires the student to have a desirable knowledge of Digital Design and Computer Organization.

**Course Objectives:**
- Design principles beyond coding.
- The motivations for the increasing trend of hardware acceleration
- The architecture of modern programmable logic devices
- Efficiencies obtained over software by direct implementation of computation in hardware
- Modern industry standard development tools.

**Course Outcomes:**
At the end of this course, the student will be able to:
- Comprehend the rationale behind and adapt to the shift towards hardware acceleration in mainstream computing.
- Build efficient computation structures to accelerate computation.
- Use industry standard tools to implement accelerator logic.
- Design and implement accelerator logic to speed up an application.

**Desirable Knowledge :** UE19CS201- Digital Design and Computer Organization

**Course Content:**
**Unit 1:Introduction**
Business and technical motivations (Moore/Dennard scaling challenges and ML / big data computation requirements), illustrative acceleration use case (NFA based regular expression matching) .-**12 Hours.**

**Unit 2 : FPGA hardware and software**
Architecture and organization (logic blocks, interconnect, memory and arithmetic blocks) of modern FPGAs (Field Programmable Gate Arrays), HDLs (Hardware Description Languages) and CAD tools. **- 10Hours.**

**Unit 3 : Accelerator Logic**
Efficient logic structures for Signed Digit arithmetic, CORDIC, FFT and convolution computations. **10 Hours.**

**Unit 4 : High Level Synthesis (HLS)**
Need for HLS, HLS Languages for FPGA/ASIC (System C and Open CL), FPGA tools for HLS, ASIC tools for HLS, matrix multiplication design and implementation using HLS . **12 Hours**

**Unit 5 : Platforms and Case Studies** Hardware acceleration platforms (FPGAs, GPUs and TPUs) and case studies (search engines in data centers and machine learning in cloud). **10 Hours**

**Text Book:**
**1. "**Digital Arithmetic", Miloš D. Ercegovac and Tomás Lang, Morgan Kaufmann, 1st Edition,2004.

**Reference Books**
**1.** Parallel Programming for FPGAs by Ryan Kastner, Janarbek Matai, and Stephen Neuendorffer, Creative Commons, 2018.
2. High Level Synthesis: from Algorithm to Digital Circuit, Editors: Coussy Philppe, Morawiec Adam. Springer, 2008 .
3. Relevant academic papers and Intel/AMD user manuals.

## UE19CS341: Design Patterns (4-0-0-4-4)

This course helps in identifying recurring problems in design, solving them using design patterns and implement them suitably for the problem on hand. This course requires the student to have a desirable knowledge of Design and Analysis of Algorithms.

**Course Objectives:**

- To impart design principles beyond coding
- To inculcate good habits in design
- To make the participants appreciate what to do and what not to do
- To compare alternate design solutions
- To appreciate the intricacies of design

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

- Separate the interface from implementation in any complex problem
- Identify the contexts where design patterns can be applied
- Select where idioms can be applied as opposed to design patterns
- Identify where not to apply design patterns
- Reliably re factor a large piece of software

**Desirable Knowledge** : UE19CS251 – Design and Analysis of Algorithms.

**Course Content:**
**Unit 1 : Design Principle**
Interface and Implementation , Open Closed principle, Liskov substitution principle, Dependency Inversion principle, Integration segregation principle.- **10 Hours**

**Unit 2: Idoms**
Handle Body idiom ( PIMPL), Reference counting, Named constructor idiom, Telescoping constructor, Bean pattern for construction,Destruction idiom. **- 12 Hours**

**Unit 3: Design Patterns**
GOF patterns, Constructional Patterns Structural Patterns, Behavioural Patterns.- **12 Hours**

**Unit 4: Patterns**
GOF patterns continued, Beyond GOF, Persistence, Multi threading.-**12 Hours**

**Unit 5: Introduction to Anti-patterns and Refactoring . - 10 Hours**

**Tools/ Languages**: C, C++, Java, C#

**Text Book:**
1: “Design Principles and Design Patterns”, Robert C Martin, 2000

**Reference Book(s):**
1: Design Patterns: Elements of Reusable Object-Oriented Software, Gamma et al, Addison Wesley, 1994.
2: “Java Design Patterns”, James W Cooper,Addison Wesley , 2000.
3: “AntiPatterns **-** The Survival Guide To Software Development Processes”, Alexander Shvets,

## UE19CS342 : 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 industry. This course requires the student to have a desirable knowledge of Problem solving with C and Microprocessor and Computer Architecture.

**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 & challenges in Parallel Programming using popular frameworks.
- Familiarize with opportunities & 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 industry.
- Engineer High performance migration of varied applications appreciating & assimilating varieties of parallelism.

**Desirable Knowledge** : UE19CS151- Problem solving with C, UE19CS252 – 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. **- 12 Hours**

**Unit 2 : Coarse Grained Parallelism**

**Laws of Parallelism,** Data, Task and Pipeline Parallelism, Pthreads, Multithreaded and Multi-Core architectures, GPUs and GPGPUs, Many-Core Heterogeneous Architectures. **- 12 Hours**

**Unit 3 : Parallelism Frameworks**

OpenMP, Race Conditions, Deadlocks & Debugging, Memory Models for Parallel Programming, Memory Consistency Models. **10 Hours**

**Unit 4 : GPU Architecture & Programming**

GPU Architectures, CUDA Programming Frameworks, OpenCL, Parallel Algorithms, Task Decompositions and Mapping. **12 Hours**

**Unit 5 : Parallel Programming Languages**

UPC, Chapel, AI Accelerators, OpenACC, CILK+, C++AMP, Concurrency in Mainstream Languages.- **10 Hours**

**Tools/Languages**: pthread, OpenMP CUDA, openCL, Chapel, UPC.

**Text Book:**

1: “Programming Massively Parallel Processors”, David Kirk and Wen-mei Hwu ,3rd ,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, 2nd, Pearson, 2016.
3: Latest Web Resources and Papers.

## UE19CS343 : 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:**
- To impart hands-on knowledge on Advanced Machine Learning Topics.
- Introduce students to programming with Tensor Flow and Keras tools.
- Provide in-depth coverage of Support Vector Machines.
- Introduce students to Deep Learning techniques – CNN and RNN.
- Introduce students to Reinforcement Learning and Generative Adversarial Networks.

**Course Outcomes:**
At the end of this course, the student will be able to:
- Implement Machine Learning techniques with Tensor Flow and Keras and develop simple game engines using Reinforcement Learning.
- Solve time-series related problems with RNN.
- Classify real-world data using Support Vector Machines.
- Classify images using CNN.
- Generate data in the form of images using GAN.

**Desirable Knowledge :** UE19CS303 – Machine Intelligence.

**Course Content:**
**Unit 1: TensorFlow and Keras and Reinforcement Learning**
Brief overview of Deep Learning Frameworks. TensorFlow: Installation, Creating and Managing Graphs,Lifecycle of a Node Value, Linear Regression, Gradient Descent, Visualizing Graphs using TensorBoard. Keras: Installation, Loading Data, Defining and Compiling Models, Fitting and Evaluating Models, Simple Neural Networks' Implementation, Hyper parameter optimization. Reinforcement Learning: Learning to Optimize Rewards, Credit Assignment Problem, Temporal Difference Learning and Q Learning. Case Study: Learning to play a simple game using deep Q-learning -implementation. **10 Hours**

**Unit 2 : Support Vector Machines**
A Very Brief Recap of the Support Vector Machine (SVM) Problem, Soft-Margin SVM (Noisy Data), Kernel Functions – Linear, Polynomial, Gaussian, Other Types, the SMO Algorithm, Multi-Class SVMs, Text- Classification, Building Applications.**12 Hours**

**Unit 3: Recurrent Neural Networks (RNN) and Unsupervised Feature Learning**
Recurrent Neurons, Memory Cells, Static and Dynamic Unrolling through Time, Variable-Length Input-Output Sequences, Training RNNs – Sequence Classifier, Predicting Time Series, Deep RNNs, LSTM Cell and GRU Cell, Text Classification with RNN, RNN Vs Naive Bayes, Seq2Seq with Attention , Bahdanau attention,Transformer Attention, Unsupervised Feature Learning – Autoencoders and Variational Auto Encoders.**12 Hours**

**Unit 4: CNN, GAN and Transfer Learning**
CNN: Architecture of CNNs, Filters, FeatureMaps, Max-Pool Layers, Other Pooling Types, Case Study: Image Recognition UsingCNN – Hands-On Implementation Using Keras. Capsule Networks – Introduction to Capsules, Dynamic Routing and Capsule Network Architecture GAN - Architecture and Training Methods, Image-Generation, Hands-On Implementation Using Keras.Transfer Learning - Motivation, Variations, Use in CNNs.**12 Hours**

**Unit 5: Paper Review and Implementation**

Selection of two state-of-the-art papers (recent) on deep learning, in depth study of the papers in class and their Implementation – Topics – Meta Learning and Graphical Neural Networks. **10 Hours**

**NOTE:** Unit 5 will be part of End-semester Assessment. Questions will be asked on the chosen papers.

**Tools/ Languages**: Tensorflow 1.15, Keras 2.3.1, Python 3.7.

**Text Book:**
1: "Advanced Deep Learning with Python" - Ivan Vasilev, Packt Publishing, 2019.

**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 for SVM, Transfer Learning.

## UE19CS344 : 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, with a focus on the key design ideas shared across many types of data intensive systems. This course requires the student to have a desirable knowledge of Database Management System.

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

- Storage Techniques, Indexing Mechanisms
- Query Processing and Query optimization techniques
- Parallel and Distributed Databases.
- Big Data systems.
- Design and build specialized database applications.

**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 and build specialized database applications.

**Desirable Knowledge :** UE19CS301 – Database Management Systems.

**Course Content:**
**Unit 1 : Relational Data Model and Storage Formats and Indexing**
Review of Relational Design theory – Functional dependencies, normalization, Overview of secondary storage, RAID and flash storage, Storing tables: row-wise, column database, database buffer, Indexing: concepts, clustered and non-clustered indices, B+-tree indices, multiple key access, hashed files, linear hash files, bitmap indices, Index definition in SQL, ++R-trees. **10 Hours**

**Unit 2 : Query Processing and Optimization**
Memory Hierarchy, accelerating access to Secondary Storage, Disk Failures, Physical-query-plan operators – one and two pass algorithms, buffer management, Query Compiler – Algebraic laws for improving query plans, Parse tree to Logical Query plans, cost based Plan selection, choosing order for joins. **12 Hours**

**Unit 3 : Parallel and Distributed Databases**:
Avenues for parallelism: I/O parallelism, interquery, inter-query and intra operation parallelism, databases for multi-core machines. Parallel Algorithms on Relations, Distributed databases - Distributed data storage, distributed transactions, commit protocols, concurrency control in distributed databases, heterogeneous and cloud-based databases., Distributed query processing, Distributed Locking, Peer-to-Peer Distributed Search. **12 Hours**

**Unit 4 : Data-Stream Management**
Stream processing, Stream processing model, Streaming architectures, Apache Spark as a stream processing engine, Spark’s distributed processing model, Spark’s resilience model, Apache Storm, Apache Kafka, Apache Flink, Amazon Kinesis. **12 Hours**

**Unit 5 : Design and Implementation of Database Systems**
Current trends in database system design and Implementation of decision support, data warehousing and data mining applications including Big Data, support for data analytics and machine learning. **10 Hours**

**Tools / Languages :** MS SQL Server, Oracle, Apache Spark, Storm, Kafka, Flink, Amazon Kinesis.

**Text Book(s):**
1: “Database Systems: The Complete Book”, H Garcia-Molina, JD Ullman and Widom 2nd Ed., Pearson, 2013.

**Reference Book(s):**
1: “Fundamentals of Database Systems”, Elmasri and Navathe, Pearson Education, 7th Edition, 2015.
2: “Streaming Systems” byTyler Akidau, Slava Chernyak, Reuven Lax, O'Reilly, July 2018.
3: “Stream Processing with Apache Spark” by Gerard Maas, Francois Garillot, O’Reilly, June 2019

## UE19CS345: Network Analysis and Mining (4-0-0-4-4)

Networks are ubiquitous in almost everything today – most notably in social graphs, large scale biological system and user-item relations. Networks are mathematically described by graphs. Graphs, being non-Euclidean, pose its own set of challenges in algorithms and computation. This course will start from static network analysis from a social graph perspective, then discuss generative techniques that mimic formation of social graphs and explain dynamics that can happen on such graphs. The course will then explore modern learning algorithms and approaches on graph data that are getting used in learning societal interactions, phenomenon in computational biology and large scale recommendation systems. This course requires the student to have a desirable knowledge of Linear Algebra and Machine Intelligence.

**Course Objectives:**

- Explain scope of this subject, learn different graphical representation of networks, interaction patterns on social graph and frameworks to identify nodes of importance in a graph at the bottom level of a social graph
- Learn how the intermediate level of the social graph can be analyzed–measures of cohesion, groups and substructures and different component detection approaches
- Explore how to model the growth of a social graph and investigate different types of dynamics on asocial graph
- Explore the classical Machine Learning approaches on social graph such as individual and collective behavior analysis, spectral clustering and semi-supervised learning
- Explore modern neural methods of learning on graphs – representation learning and semi-supervised graph algorithms using neural techniques

**Course Outcomes:**

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

- Should be able to analyze static social graph and identify the key actors.
- Should be able to extend the above analysis to identification of sub-structure and components at the intermediate level of the graph.
- Should be able to extend the above analysis by comparing with generative models. The student should additionally be able to model a dynamic phenomenon such as epidemic and diffusion of innovation.
- Should be able to apply classical machine learning techniques on graph to analyze behavior, social recommendation etc.
- Should be able to solve a learning problem on a non-Euclidean graph dataset using semi-supervised neural algorithms.

**Desirable Knowledge** : UE19MA251-Linear Algebra, UE19CS303–Machine Intelligence

**Course Content:**

**Unit 1: Graphical representation of network, Interaction patterns in a social graph, Centrality analysis:**

Applications of network analysis, **Representing a network by graph:** Using matrix to represent social relations. Types of network – directed and undirected graph, dynamic graph, homogenous and heterogenous graph, dynamic graph, hyper graph, knowledge graph, unipartite and bipartite graph-Affiliation Networks. **Interaction Patterns:** Ego-centric Networks, homophily, Tie-strengths and structural holes. **Centrality Analysis in Graph:** Degree, Betweenness, Closeness, Eigen vector, Katz and PageRank Centrality. **- 10 Hours**

**Unit 2: Measures of cohesion, Groups and substructures, Community detection**

**Measures of cohesiveness:** Degree distribution, Diameters, Transitivity and Reciprocity, Clustering coefficient.**Gro ups and substructures:**top-down view – weak and strong component, in-component, out-component, giant component. 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 algorithms**. - 10 Hours**

**Unit 3: Generative models to mimic graph formation, Dynamics in a social graph**
**Generative Models:** Properties of networks to be modelled- Clustering coefficient, Power Laws, Fat Tails, Small World: Milgram's small world experiment. Scale-free networks. Random Graph Models: Erdos Renyi models, Thresholds and Phase Transitions. Growth Models by Preferential attachment: Barabasi and Albert's model. Wats-Strogatz small world model. Refinement of Small World Model - Decentralized or Myopic Search. Comparison of generative models. **Dynamics on Graph** : Herd behavior, Diffusion-Diffusion of innovation, Bass model of diffusion – epidemic modeling, Cascades and Contagions, Percolation and Robustness of Networks, Effects of communities and centralities on diffusion. - **12 Hours**

**Unit 4: Traditional Machine Learning on Graph**
**Classic Graph ML tasks :** Node level prediction, Link level prediction, Graph level prediction and Graph clustering. **Individual behavior analysis** – analysis, modeling and prediction. **Collective behavior analysis** – analysis, modeling and prediction. **Social recommendation** – recommendation using social context. **Spectral clustering**.**Semi-supervised Link Prediction** based on Label Propagation. **- 12 Hours**

**Unit 5: Graph Neural Network**
**Representation learning algorithms:** Learning Challenges posed by non-Euclidean data in Graph, Random Walk based Methods - DeepWalk, Node2Vec. **Representation learning using deep learning:** Graph Neural Network, Graph Convolution Networks, GraphSAGE – an inductive extension, Graph Attention Network. Applications of concepts discussed. - **12 Hours**

**Tools/Languages**: Gephi, NetworkX, Pytorch/Tensorflow, Python.

**Text Book:**
1: "Networks, Crowds, and Markets: Reasoning About a Highly Connected World", D Easley and J Kleinberg, Cambridge University Press, 2010.

**Reference Book(s):**
1:"Social Media Mining",Reza Zafarani, Cambridge University Press, 2015 (Asian Economic edition available)
3."Introduction to Graph Neural networks", Zhiyuan Liuand Jie Zhou, Synthesis Lectures on Artificial Intelligence and Machine learning, Morgan and Claypool Publishers, 2020

## UE19CS346: Information Security(4-0-0-4-4)

This course will present security aspects from a software life cycle process - requirement, Architecture - Design, coding, and testing. Students will have opportunity well dwell in to technical "how to " with hands on sessions and some case studies.

**Course Objectives:**

- Understand various cyber threats and attacks; learn about Secure Software development process.
- Understand how privilege escalation attacks and Buffer overflows happen Learn to analyze malware and differentiate between categories such as Virus and Worms.
- Understand the concept of Threat Modelling and its application.
- Understand and learn about the most common Web application security vulnerabilities.
- Understand and apply various security testing techniques and tools.

**Course Outcomes:**

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

- Be able to identify possible misuse cases in the context of software development.
- Apply the different concepts and techniques learnt to prevent privilege escalation and Buffer overflow attacks. Apply the acquired knowledge to create a Worm from a Virus.
- Apply Threat Modelling techniques to expose inherent/dormant vulnerabilities in a given software design /architecture and propose alternate solutions.
- Be able to design and develop Secure Web applications.
- Be able to perform Penetration testing on the given software system.

**Course Content:**

**Unit 1: Introduction**

Software Threats and Vulnerabilities, OWASP Top 10, SANS Top 25, CVE, etc. The CIA Triad - Core Security Principles, Vulnerabilities, Threats and Attacks, 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 Software Development Life Cycle, Risk analysis, SDL, SDL practices.**12 Hours**

**Unit 2: Privilege Escalation Attacks**

Set UID program and environment variable, Shell shock attack, buffer overflow. Non-executable stack, defeat countermeasures, Environment Setup, Tasks involved in the attack, Function prologue and epilogue, Format string Vulnerabilities, Vulnerabilities code launching attack. Malware and abra worm, Stuxnet worm, Morris worm.**12 Hours**

**Unit 3: Threat Modelling**

Threat Modelling, Trust Boundaries Threat Modelling, Brainstorming, Modelling Methods, stride privacy threats, Taxonomy Of Privacy, privacy tools, processing threats, defensive tactics and technologies. EOP card game.**10 Hours**

**Unit 4 : Web Application Security Issues**

Challenges, browser security, web security. SQL injection. Basic Structure of Web Traffic, Relational Database Elements, SQL Tutorial, Interacting with Database in Web Application, Launching SQL Injection Attacks. Cross Site Request Forgery, CSRF Attacks on HTTP GET / POST Services, Countermeasures, Cross-Site Scripting Attack, XSS Attacks, Countermeasures. HTTP Security: Overview of HTTP Security, MITM Attacks and Solutions, HTTP Security Headers Privacy Issues and HTTP Authentication. **12 Hours**

**Unit 5: Security Testing Countermeasures - Tools, Frameworks, and Services:**

Static analysis tools, penetration testing, Benefits, Drawbacks, Web hacking Tools, Nmap for network probing, Web proxies, Metasploit, Ethical Hacking, Fuzzing.**10 Hours**

**Note:** Giving hands on experience for relevant topics in the form of Lab or Assignment. Relevant cyber security case for undergraduate students is discussed.

**Tools / Languages**: Seed labs, Scapy, Burp-suit, N-Map,  C- Language.

**Text Book:**
1: “Computer and Internet Security”, Hands on Approach”, Wenliang Du, 2nd Edition, 2019.

**Reference Book(s):**
1: Computer Security – Principles and Practice”, William Stallings, 3rd Edition, 2014.

## UE19CS347: Wireless Network Communication (4-0-0-4-4)

Wireless Networks Communication is a dynamic field that has spurred tremendous excitement and technological advances. This course provides a comprehensive understanding of the fundamental principles, characteristics, performance limits of wireless systems, their security issues and the insights associated with their design. This course requires the student to have a desirable knowledge of Computer Networks.

**Course Objectives:**

- Introduce the emerging trends of wireless network technologies
- Compare and contrast the wireless network technologies depending on the usage models
- Explain the characteristics of wireless channels and analyze its impact during communication
- Discuss various design parameters of communication.
- Identify different attacks on wireless network and explore several mitigation approaches.

**Course Outcomes:**

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

- Identify and Apply the appropriate wireless technology for real time applications.
- Simulate the channel characteristics such as path loss, shadowing, analyze the wireless networks and understand the Multipath channel models.
- Analyze emerging enhancements such as Adaptive Modulation and Multiple Input Multiple Output System.
- Capture the transmitted packets of wireless networks and analyze them for the wireless communication protocols
- Determine the threats on wireless network and Apply wireless security mechanisms.

**Desirable Knowledge** : UE19CS253 – Computer Networks.

**Unit 1: Overview of Wireless communication**
Introduction,Wireless LAN Technology: IEEE 802.11, WPAN Technologies: Bluetooth, Zigbee, NFC, 6LOWPAN, LPWAN- LORA, Weightless, Wireless Local Loop (WLL)-LMDS, MMDS, WiMAX, Long Range Communication- Satellite Communication, Wireless communication analysis using Wireshark. **12 Hours**

**Unit 2: Overview of Wireless Communication-II** Cellular Network: Cellular System Fundamentals, Channel Reuse, SIR and User Capacity, Interference Reduction Techniques, Mobile Applications and Mobile IP, Tradeoff between Battery, Bandwidth and Distance, Wireless Channel Models: Path Loss and Shadowing Models, Millimeter Wave Propagation, Fading Models: Statistical Fading Models, Narrowband Fading, Wideband Fading Models. **12 Hours**

**Unit 3: Impact of Fading and ISI on Wireless Performance**
Capacity of Wireless Channels, Digital Modulation and its Performance, Adaptive Modulation: Adaptive Transmission System, Adaptive Techniques; Variable-Rate Variable-Power MQAM, Multiple Input/ Multiple Output (MIMO): Narrowband MIMO Model, Parallel Decomposition of the MIMO Channel, MIMO channel capacity, MIMO Diversity Gain: Beam forming, Diversity/Multiplexing Tradeoffs, Space-Time Modulation, Frequency-Selective MIMO Channels, Smart Antennas. **12 Hours**

**Unit 4: Multicarrier Systems**
Data Transmission using Multiple Carriers, Multicarrier Modulation with Overlapping Subchannels, OFDM, OFDMA, Single- carrier FDMA, Challenges in Multicarrier Systems, Multiuser Channels: The Uplink and Downlink, Multiple Access, Random Access, Downlink (Broadcast) Channel Capacity, Uplink (Multiple Access) Channel Capacity, Uplink/Downlink Duality. **10 Hours**

**Unit 5: Security**
Attacks on Wireless Network, Attacks on Wireless Clients, WEP-Wired Equivalent Privacy Protocol Security, WiFi Security: WiFi Protected Access, WiFi Protected Access 2, WPA2 Wireless Enterprise Network, RADIUS, Handling Rogue Access Points, Theory of Defense for security wireless Networks.
**10 Hours**

**Tools/ Language :** Wireshark, Claynet/packetracer.

**Text Book:**
1: “Wireless Communication”, Andrea Goldsmith, First Edition, Cambridge University Press, 2012

**Reference Book(s):**
1: "Wireless Communication Networks and Systems", by Cory Beard and William Stallings,1st edition, pearson, 2015.
2: "Wireless Network Security: A Beginner’s Guide" by Tyler Wrightson, McGraw-Hill Education; edition, 2012.