Appendix-72
Resolution No. 27 {27-1 (27-1-11)}

INDEX
BACHELOR OF VOCATION – SOFTWARE DEVELOPMENT
SEMESTER-III TO SEMESTER-VI

Annexure A
Detailed Syllabus – Discipline Specific Core

DISCIPLINE SPECIFIC CORE COURSE – DSC-07: DATA STRUCTURES

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Data Structures	4	3	0	1	Class XII pass with Mathematics	DSC-01

Learning Objectives:

1. *To introduce the fundamentals of data structures*
2. *To get familiar with programming*

Learning Outcomes:

1. *Develop the ability to use basic data structures like array, stacks, queues, lists, trees and hash tables to solve problems.*
2. *Use well-organized data structures in solving various problems.*
3. *Differentiate the usage of various structures in problem solutions.*
4. *Implement algorithms to solve problems using appropriate data structures.*

Unit I **(5 hours)**

Arrays: Single and multi-dimensional arrays, analysis of insert, delete and search operations in arrays (both linear search and binary search), implementing sparse matrices, applications of arrays to sorting: selection sort, insertion sort, bubble sort, comparison of sorting techniques via empirical studies.

Unit II **(5 hours)**

Linked Lists: Singly- linked, doubly-linked and circular lists, analysis of insert, delete and search operations in all the three types, implementing sparse matrices.

Unit III **(10 hours)**

Queues: Array and linked representation of queue, de-queue, comparison of the operations on queues in the two representations. Applications of queues.

Unit IV **(15 hours)**

Stacks: Array and linked representation of stacks, comparison of the operations on stacks in the two representations, implementing multiple stacks in an array; applications of stacks: prefix, infix and postfix expressions, utility and conversion of these expressions from one to another;

applications of stacks to recursion: developing recursive solutions to simple problems, advantages and limitations of recursion.

Unit V **(10 hours)**

Trees and Heaps: Introduction to tree as a data structure; binary trees, binary search trees, analysis of insert, delete, search operations, recursive and iterative traversals in binary search trees. Height-balanced trees (AVL), B trees, analysis of insert, delete, search operations on AVL and B trees. Introduction to heap as a data structure. Analysis of insert, extract-min/max and delete-min/ max operations, applications to priority queues.

Hash Tables: Introduction to hashing, hash tables and hashing functions -insertion, resolving collision by open addressing, deletion, searching and their analysis, properties of a goodhash function.

References

1. Michael T. Goodrich, Roberto Tamassia and Michael H. Goldwasser (2013), Data Structures and Algorithms in Python, Wiley.

2. Rance D. Necaise, Data Structures and Algorithms Using Python, John Wiley & Sons, Inc.

3. Introduction to Algorithms, by Cormen, Leiserson, Rivest, and Stein, MIT Press, Third Edition, 2009.

List of Practical **(30 hours)**

A practical implementation of various data structure such as Array, Queues, Stacks, Linked List and Trees.

DISCIPLINE SPECIFIC CORE COURSE – DSC-08: Web Design and Development

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Web design and development	4	3	0	1	Class XII	NIL

***Learning objectives*:**

1. To introduce the fundamentals of Internet, and the principles of web design.
2. To construct basic websites using HTML and Cascading Style Sheets.
3. To build dynamic web pages with validation using Java Script objects and by applying different event handling mechanisms.
4. To develop modern interactive web applications using PHP, XML and MySQL

Learning Outcomes:

1. *Structure and implement HTML/CSS.*
2. *Implement basic JavaScript.*
3. *Learn server side scripting language PHP and integration with database using MYSQL.*

UNIT-I **(5 Hours)**

Introduction to HTML & CSS:

HTML Basics, HTML Responsive, HTML Entities, HTML Forms, HTML5 Canvas, HTML5 SVG, HTML5 Data Storage, HTML5 Audio and Video , CSS Introduction, CSS Syntax, CSS Text, CSS Backgrounds, CSS Fonts, CSS Links, CSS Lists, CSS Tables, CSS Box Model, CSS Margins, Dimensions, Display, CSS Navigation Bar, CSS Attribute Selectors, CSS Rounded Corners, CSS Border Images, CSS Backgrounds, CSS Colors, CSS Animations.

UNIT-II **(10 Hours)**

Introduction to JavaScript:

JavaScript Introduction, JavaScript Output, JavaScript Variables, JavaScript Operators, JavaScript Arithmetic, JavaScript Data Types, JavaScript Assignment, JavaScript Functions, JavaScript Objects, JavaScript Scope, JavaScript Events, JavaScript Strings and String Methods, JavaScript Numbers and Number Methods, JavaScript Math, JavaScript Dates: Formats and Methods, JavaScript Booleans , JavaScript Comparisons, JavaScript Conditions, JavaScript Switch, JavaScript Loops, JavaScript Break, JavaScript Type, JavaScript Forms (API and Validation), JavaScript Objects, JavaScript Functions, JavaScript DOM, JavaScript Browser BOM, JavaScript Frameworks

UNIT-III **(15 Hours)**

Introduction to Bootstrap: Bootstrap Introduction, Bootstrap Components, Bootstrap Plugins, Bootstrap Grids, Bootstrap JS , PHP Introduction-Installing PHP, PHP Syntax, PHP Variables, PHP Data Types, PHP Strings, PHP Constants, PHP Operators, PHP Programming Loops, PHP Functions, PHP Arrays, PHP Super-global, PHP Forms and XML- PHP Form Handling, PHP Form Validation (Server side).

UNIT-IV **(15 Hours)**

PHP with MySQL: PHP MySQL Database, PHP Connecting to Database, PHP Creating Records, PHP Selecting Records, PHP Deleting Records, PHP Updating Records, PHP Limit Data, PHP Insert Multiple.

References

1. Learning PHP, MySQL & JavaScript: With JQuery, CSS & HTML5 by Robin Nixon , O'Reilly Media, Inc.

2. PHP and MySQL for Dynamic Web Sites: Visual QuickPro Guide by Larry Ullman , Fifth Edition.

List of Practicals (30 hours)

1. Design a home page which displays information about your college department using headings, HTML entities and paragraphs.
2. Implement different types of list tags, hyperlinks, marquee tag and HTML formatting tags in the college department homepage.
3. Create a web page having two frames, Frame 1 containing links and another with contents of the link. When a link is clicked appropriate contents should be displayed on Frame 2. Also, insert an iframe in the same page.
4. Design your course timetable and display it in tabular format.
5. Design an admission form for any course in your college with text, password fields, drop-down list, check-boxes, and radio buttons, submit and reset button etc. with proper CSS formatting.
6. Create a website for online book stores with Home, Login, Catalogue, Registration page with links to all these pages in a menu on top of every page. Embed heading, paragraph, images, video, .iframe, form controls, table, and list in this website. Use both Internal and external CSS in this.
7. Write a JavaScript program to display the current day and time.
8. Write a JavaScript program to
   a) Remove a character at the specified position of a given string and return the new string.
   b) Change the case of a string. (I.e. upper case to lower case and vice-versa).
9. Write a JavaScript program to compute the sum of elements of a given array of integers.
10. Develop and demonstrate a HTML file that includes JavaScript script for taking full name in a text field and display first, middle, last name *in 3 different labels. Middle and last name may be optional, thus messages like "NA" should be displayed in corresponding labels. If input contains 2 words, then they should be considered as first and last names.
11. Design HTML form for keeping student record, apply JavaScript validation for restriction of mandatory fields, numeric field, email-address field, specific value in a field etc.
12. Write a JavaScript code that displays text "Bigger Text" with increasing font size in the interval of 10ms in red color, when the font size reaches 50 pt. it displays "Smaller Text" in green color. Then the font size should decrease to 5pt and then stop.
13. Write a PHP script that removes the whitespaces from a string.
14. Create a login page having user name and password. On clicking submit, a welcome message should be displayed if the user is already registered (i.e.name is present in the database) otherwise error message should be displayed.
15. Create a simple 'birthday countdown' script, the script will count the number of days between current day and birth day.

DISCIPLINE SPECIFIC CORE COURSE – DSC-09: Operating Systems

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Operating Systems	4	3	0	1	Class XII	NIL

***Learning objectives*:**

1. *Learn fundamental operating system abstractions such as processes, threads, files, semaphores, IPC abstractions, shared memory regions, etc.,*
2. *Learn how the operating system abstractions can be used in the development of application programs, or to build higher level abstractions,*
3. *Learn how the operating system abstractions can be implemented,*
4. *Learn the principles of concurrency and synchronization, and apply them to write correct concurrent programs/software,*
5. *Learn basic resource management techniques (scheduling, time management, space management) and principles and how they can be implemented. These also include issues of performance and fairness objectives, avoiding deadlocks, as well as security and protection.*

Learning Outcomes:

1. *Understand the need of an Operating System & Define Multiprogramming and multithreading concepts.*
2. *Implement Process Synchronization service (Critical Section, Semaphores), CPU scheduling service with various algorithms.*
3. *Learn Main memory Management (Paging, Segmentation) algorithms, Handling of Deadlocks*
4. *Identify and appreciate the File systems Services, Disk Scheduling service*

UNIT - I **(5 hours)**

Introduction: Operating Systems (OS) definition and its purpose, Multi-programmed and Time Sharing Systems, OS Structure, OS Operations: Dual and Multi-mode, OS as resource manager.

UNIT – II **(5 hours)**

Operating System Structures: OS Services, System Calls: Process Control, File Management, Device Management, and Information Maintenance, Inter-process Communication, and Protection, System programs, OS structure- Simple, Layered, Microkernel, and Modular.

UNIT - III **(10 hours)**

Process Management : Process Concept, States, Process Control Block, Process Scheduling, Schedulers, Context Switch, Operation on processes, Threads, Multicore Programming, Multithreading Models, Threads, Process Scheduling Algorithms: First Come First Served, Shortest-Job-First, Priority & Round-Robin, Process Synchronization: The critical-section problem and Peterson's Solution, Deadlock characterization, Deadlock handling.

UNIT – IV **(10 hours)**

Memory Management: Physical and Logical address space, Swapping, Contiguous memory allocation strategies - fixed and variable partitions, Segmentation, Paging.

Virtual Memory Management: Demand Paging and Page Replacement algorithms: FIFO Page Replacement, Optimal Page replacement, and LRU page replacement.

UNIT – V **(15 hours)**

File System: File Concepts, File Attributes, File Access Methods, Directory Structure: Single-Level, Two-Level, Tree-Structured, and Acyclic-Graph Directories.

Mass Storage Structure: Magnetic Disks, Solid-State Disks, And Magnetic Tapes, Disk Scheduling algorithms: FCFS, SSTF, SCAN, C-SCAN, LOOK, and C-Look Scheduling.

References

1. *Silberschatz, A., Galvin, P. B., Gagne G. Operating System Concepts, 9th edition, John Wiley Publications, 2016*
2. *Dhamdhere, D. M. Operating Systems: A Concept-based Approach. 2nd edition, Tata McGraw-Hill Education, 2017*
3. *Kernighan, B. W., Rob Pike, R. The UNIX Programming Environment. Englewood Cliffs, NJ: Prentice-Hall, 1984*
4. *Stallings, W. Operating Systems: Internals and Design Principles. 9th edition, Pearson Education, 2018*
5. *Tanenbaum, A. S. Modern Operating Systems. 3rd edition, Pearson Education, 2007*

List of Practicals :(30 hours)

1. Write a program (using fork() and/or exec() commands) where parent and child execute:
   a) Same program, same code.
   b) Same program, different code.
   c) Before terminating, the parent waits for the child to finish its task.
2. Write a program to report behavior of Linux kernel including kernel version, CPU type and model. (CPU information)
3. Write a program to report behavior of Linux kernel including information on configured memory, amount of free and used memory. (Memory information)
4. Write a program to print file details including owner access permissions, file access time, where file name is given as argument.
5. Write a program to copy files using system calls.
6. Write a program to implement FCFS scheduling algorithm.
7. Write a program to implement Optimal scheduling algorithm.
8. Write a program to implement the SJF scheduling algorithm.
9. Write a program to implement a non-preemptive priority based scheduling algorithm.
10. Write a program to implement SRJF scheduling algorithm.
11. Write a program to calculate sum of n numbers using thread library.
12. Write a program to implement first-fit, best-fit and worst-fit allocation strategies.

DSE-01(a): Programming Using R

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Programming using R	4	2	0	2	Class XII with Mathematics	NIL

***Learning objectives*:**

1. *Master the use of the R and RStudio interactive environment.*
2. *Expand R by installing R packages.*
3. *Explore and understand how to use the R documentation.*
4. *Read Structured Data into R from various sources.*
5. *Understand the different data types in R.*
6. *Understand the different data structures in R.*

Learning Outcomes:

1. *Develop an R script and execute it*
2. *Install, load and deploy the required packages, and build new packages for sharing and reusability*
3. *Extract data from different sources using API and use it for data analysis*
4. *Visualize and summarize the data*
5. *Design application with database connectivity for data analysis*

UNIT-I **(5 hours)**

Introduction: R interpreter, Introduction to major R data structures like vectors, matrices, arrays, list and data frames, Control Structures, vectorized if and multiple selection, functions.

UNIT-II **(10 hours)**

Installing, loading and using packages: Read/write data from/in files, extracting data from web-sites, Clean data, Transform data by sorting, adding/removing new/existing columns, centering, scaling and normalizing the data values, converting types of values, using string in-built functions, Statistical analysis of data for summarizing and understanding data, Visualizing data using scatter plot, line plot, bar chart, histogram and box plot

UNIT-III **(10 hours)**

Designing GUI: Building interactive application and connecting it with database.

UNIT-IV **(5 hours)**

Building Packages.

References:

1. Cotton, R., Learning R: a step by step function guide to data analysis. 1st edition. O'reilly Media Inc.
2. Gardener, M.(2017). Beginning R: The statistical programming language, WILEY.
3. Lawrence, M., & Verzani, J. (2016). Programming Graphical User Interfaces in R. CRC press. (ebook)

List of Practical :(60 hours)

Q1. Write an R script to do the following:

a) Simulate a sample of 100 random data points from a normal distribution with mean 100 and standard deviation 5 and store the result in a vector.
b) Visualize the vector created above using different plots.
c) Test the hypothesis that the mean equals 100.
d) Use Wilcox test to test the hypothesis that mean equals 90.

Q2. Using the Algae data set from package DMwR to complete the following tasks.

a) Create a graph that you find adequate to show the distribution of the values of algae a6.
b) Show the distribution of the values of size 3.
c) Check visually if oPO4 follows a normal distribution.
d) Produce a graph that allows you to understand how the values of NO3 are distributed across the sizes of river.
e) Using a graph check if the distribution of algae a1 varies with the speed of the river.
f) Visualize the relationship between the frequencies of algae a1 and a6. Give the appropriate graph title, x-axis and y-axis title.

Q3. Read the file Coweeta.CSV and write an R script to do the following:

a) Count the number of observations per species.
b) Take a subset of the data including only those species with at least 10 observations.
c) Make a scatter plot of biomass versus height, with the symbol color varying by species, and use filled squares for the symbols. Also add a title to the plot, in italics.
d) Log-transform biomass, and redraw the plot.

Q4. The built-in data set mammals contain data on body weight versus brain weight. Write R commands to:

a) Find the Pearson and Spearman correlation coefficients. Are they similar?
b) Plot the data using the plot command.
c) Plot the logarithm (log) of each variable and see if that makes a difference.

Q5. In the library MASS is a dataset UScereal which contains information about popular breakfast cereals. Attach the data set and use different kinds of plots to investigate the following relationships:

a) relationship between manufacturer and shelf
b) relationship between fat and vitamins
c) relationship between fat and shelf
d) relationship between carbohydrates and sugars
e) relationship between fiber and manufacturer
f) relationship between sodium and sugars

Q6. Write R script to:

Do two simulations of a binomial number with n = 100 and p = .5. Do you get the same results each time? What is different? What is similar?

Do a simulation of the normal two times. Once with n = 10, $\mu = 10$ and $\sigma = 10$, the other with n = 10, $\mu = 100$ and $\sigma = 100$. How are they different? How are they similar? Are both approximately normal?

Q.7 Create a database medicines that contains the details about medicines such as {manufacturer, composition, price}. Create an interactive application using which the user can find an alternative to a given medicine with the same composition.
Q.8 Create a database songs that contains the fields {song_name, mood, online_link_play_song}. Create an application where the mood of the user is given as input and the list of songs corresponding to that mood appears as the output. The user can listen to any song form the list via the online link given
Q.9 Create a package in R to perform certain basic statistics functions.

DSE-01 (b): Discrete Structures

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Discrete Structures	4	3	0	1	Class XII with Mathematics	NIL

***Learning objectives*:**

1. To teach students how to think logically and mathematically.
2. To stress on mathematical reasoning and describe different ways in which mathematical problems could be solved.
3. To cover four thematic areas: mathematical reasoning, combinatorial analysis, discrete structures, and mathematical modelling.
4. To touch upon topics like logic, proofs, set theory, counting, probability theory (the discrete part of the subject), graph theory, trees, Boolean algebra, and modelling computation.

Learning Outcomes:

1. *Relate mathematical concepts and terminology to examples in the domain of Computer Science.*
2. *Model real world problems using various mathematical constructs.*
3. *Use different proofing techniques; construct simple mathematical proofs using logical arguments.*
4. *Divide a problem or a proof into smaller cases.*
5. *Formulate mathematical claims and construct counterexamples.*

UNIT-I **(7 hours)**

Sets, Functions, Sequences and Summations, Relations. Sets: Set Operations, Computer Representation of Sets, Countable and Uncountable Set, Principle of Inclusion and Exclusion, Multi-sets; Functions: One-to-one and Onto Functions, Inverse Functions and Compositions of Functions, Graphs of Functions Sequences and Summations: Sequences, Special Integer Sequences, Summations; Relations: Properties of Binary Relations, Equivalence relations and Partitions, Partial Ordering Relations and Lattices.

UNIT-II **(8 hours)**

Logic and Proofs. Propositional Logic, Propositional Equivalences, Use of first-order logic to express natural language predicates, Quantifiers, Nested Quantifiers, Rules of Inference, Introduction to Proofs, Proof Methods and Strategies, Mathematical Induction.

UNIT-III **(7 hours)**

Number Theory. Division and Integers, Primes and Greatest Common Divisors, Representation of Integers, Algorithms for Integer Operations, Modular Exponentiation, Applications of Number

Theory.

UNIT-IV **(8 hours)**

Combinatorics/Counting. The Pigeonhole Principle, Permutations and Combinations, Binomial Coefficients, Generalized Permutations and Combinations, Generating Permutations and Combinations.

UNIT-V **(10 hours)**

Graphs and Trees. Graphs: Basic Terminology, Multigraphs and Weighted Graphs, Paths and Circuits, Eulerian Paths and Circuits, Hamiltonian paths and Circuits, Shortest Paths, Spanning Trees, Graph Isomorphism, Planar Graphs; Trees: Trees, Rooted Trees, Path Lengths in Rooted Trees.

UNIT-VI **(5 hours)**

Recurrence. Recurrence Relations, Generating Functions, Linear Recurrence Relations with Constant Coefficients and their solution.

References

1. *C.L. Liu & Mahopatra, Elements of Discrete mathematics. 3rd edition. Tata McGraw Hill. 2008.*
2. *Kenneth R., Discrete Mathematics and Its Applications. 6th edition. Mc Graw Hill. 2006.*

List of practicals (30 Hours)

1. Write a Program to create a SET A and determine the cardinality of SET for an input array of elements (repetition allowed) and perform the following operations on the SET:
a) ismember (a, A): check whether an element belongs to set or not and return value as true/false.
b) powerset(A): list all the elements of power set of A.

2. Create a class SET and take two sets as input from user to perform following SET Operations:
a) Subset: Check whether one set is a subset of other or not.
b) Union and Intersection of two Sets.
c) Complement: Assume Universal Set as per the input elements from the user.
d) Set Difference and Symmetric Difference between two SETS
e) Cartesian Product of Sets.

3. Create a class RELATION, use Matrix notation to represent a relation. Include functions to check if the relation is Reflexive, Symmetric, Anti-symmetric and Transitive. Write a Program to use this class.

4. Use the functions defined in Ques 3 to check whether the given relation is:
a) Equivalent, or
b) Partial Order relation, or
c) None

5. Write a Program to implement Bubble Sort. Find the number of comparisons during each pass and display the intermediate result. Use the observed values to plot a graph to analyse the complexity of algorithm.

6. Write a Program to implement Insertion Sort. Find the number of comparisons during each pass and display the intermediate result. Use the observed values to plot a graph to analyse the complexity of algorithm.

7. Write a Program that generates all the permutations of a given set of digits, with or without repetition. (For example, if the given set is {1,2}, the permutations are 12 and 21). (One method is given in Liu)

8. Write a Program to accept the truth values of variables x and y, and print the truth table of the following logical operations:
a) Conjunction f) Exclusive NOR
b) Disjunction g) Negation
c) Exclusive OR h) NAND
d) Conditional i) NOR
e) Bi-conditional

9. Write a Program to store a function (polynomial/exponential), and then evaluate the polynomial. (For example store f(x) = 4n3 + 2n + 9 in an array and for a given value of n, say n = 5, evaluate (i.e. compute the value of f(5)).

10. Write a Program to represent Graphs using the Adjacency Matrices and check if it is a complete graph.

11. Write a Program to accept a directed graph G and compute the in-degree and out-degree of each vertex.

DSE-01 (c): Digital Image Processing

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Digital Image Processing	4	3	0	1	Class XII with Mathematics	NIL

***Learning objectives*:**

1. *To understand the sensing, acquisition and storage of digital images.*
2. *To study the image fundamentals and mathematical transforms necessary for image processing.*
3. *To understand the digital processing systems and corresponding terminology.*
4. *To understand the base image transformation domains and methods.*

Learning Outcomes:

1. *Understand the fundamentals of Image Processing and its role and importance in avariety of applications.*
2. *Write programs to read/write and manipulate images for the purpose of enhancement.*
3. *Understand the need for image transforms and their properties.*
4. *Understand different causes for image degradation and use various techniques to restore images.*

UNIT-I **(8 hours)**

Introduction: Digital Image Fundamentals, Brightness, Adaptation and Discrimination, Light and Electromagnetic Spectrum, Image Sampling and Quantization, Some Basic Relationships between Pixels Types of images.

UNIT-II **(7 hours)**

Spatial Domain Filtering: Some Basic Intensity Transformation Functions, Histogram Equalization, Spatial Correlation and Convolution, Smoothening Spatial Filters-Low pass filters, Order Statistics filters; Sharpening Spatial Filters- Laplacian filter.

UNIT-III **(8 hours)**

Filtering in Frequency Domain: The Discrete Fourier Transformation (DFT), Frequency Domain Filtering:-Ideal and Butterworth Low pass and high pass filters

UNIT-IV **(7 hours)**

Image Degradation and Compression: Noise models, Noise Restoration Filters, Fundamentals of Image Compression, Huffman Coding, Run Length Coding

UNIT-V **(10 hours)**

Morphological Image Processing: Erosion, Dilation, Opening, Closing, Hit-or-Miss Transformation, Basic Morphological Algorithms.

UNIT VI **(5 hours)**

Image Segmentation: Point, Line and Edge Detection, Thresholding.

References:

1. *Gonzalez, R. C., & Woods, R. E. Digital Image Processing. 4th edition. Pearson Education, 2017*
2. *Castleman, K. R. Digital Image Processing. 1st edition. Pearson Education,2007*
3. *Gonzalez, R. C., Woods, R. E., & Eddins, S. Digital Image Processing using MATLAB. Pearson Education Inc., 2004*
4. *Jain, A. K. Fundamentals of Digital Image Processing. 1st edition Prentice Hall of India, 1988.*

List of practicals (30 Hours)

1. Write program to read and display digital image using MATLAB or SCILAB
a. Become familiar with SCILAB/MATLAB Basic commands
b. Read and display image in SCILAB/MATLAB
c. Resize given image
d. Convert given color image into gray-scale image
e. Convert given color/gray-scale image into black & white image
f. Draw image profile
g. Separate color image in three R G & B planes
h. Create color image using R, G and B three separate planes
i. Flow control and LOOP in SCILA
j. Write given 2-D data in image file

2. To write and execute image processing programs using point processing method
a. Obtain Negative image
b. Obtain Flip image
c. Thresholding
d. Contrast stretching

3. To write and execute programs for image arithmetic operations
a. Addition of two images
b. Subtract one image from other image
c. Calculate mean value of image
d. Different Brightness by changing mean value

4. To write and execute programs for image logical operations
a. AND operation between two images
b. OR operation between two images
c. Calculate intersection of two images
d. Water Marking using EX-OR operation
e. NOT operation (Negative image)

5. To write a program for histogram calculation and equalization using
a. Standard MATLAB function
b. Program without using standard MATLAB functions
c. C Program

6. To write and execute program for geometric transformation of image
a. Translation
b. Scaling
c. Rotation
d. Shrinking
e. Zooming

7. To understand various image noise models and to write programs for
a. image restoration
b. Remove Salt and Pepper Noise
c. Minimize Gaussian noise
d. Median filter and Weiner filter

8. Write and execute programs to remove noise using spatial filters
a. Understand 1-D and 2-D convolution process
b. Use 3x3 Mask for low pass filter and high pass filter

9. Write and execute programs for image frequency domain filtering
a. Apply FFT on given image
b. Perform low pass and high pass filtering in frequency domain
c. Apply IFFT to reconstruct image

10. Write a program in C and MATLAB/SCILAB for edge detection using different edge detection mask

11. Write and execute program for image morphological operations erosion and dilation.

DISCIPLINE SPECIFIC CORE COURSE – DSC-10: Software Modelling

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Software Modelling	4	3	0	1	Class XII	NIL

***Learning objectives*:**

1. *Design and develop software systems (including analysis, design, construction, maintenance, quality assurance and project management) using the appropriate theory, principles, tools and processes.*
2. *Use appropriate computer science and mathematics principles in the development of software systems.*
3. *Solve problems in a team environment through effective using various tools, techniques and processes.*
4. *Introduce the current issues presently involved in effectively performing duties as a software practitioner in an ethical and professional manner for the benefit of society.*
5. *Practice the lifelong learning needed in order to keep current as well as new challenging issues in real life scenario.*
6. *Develop software in at least one application domains like Healthcare, safety, Society, Legal, Environment, Communication etc.*

Learning Outcomes:

1. *Illustrate the strengths and weaknesses of certain models and logics including state machines, algebraic and process models, and temporal logic;*
2. *Describe appropriate abstract formal models for certain classes of systems, describe abstraction relations between different levels of description, and reason about the correctness of refinements;*
3. *Prove elementary properties about systems described by the models introduced in the course.*

Unit-I **(10 hours)**

Introduction: The Evolving Role of Software, Software Characteristics, Changing Nature of Software, Software Engineering as a Layered Technology, Software Process Framework, Framework and Umbrella activities, process models, Capability Maturity Model Integration (CMMI). Software Requirement Analysis, Initiating Requirement Engineering Process, Requirement Analysis and Modeling Techniques, Flow Oriented Modeling, Need for SRS, Characteristics and Components of SRS.

Unit-II **(7 hours)**

Software Project Management: Estimation in Project Planning Process, Project Scheduling. Software Risks, Risk Identification, Risk Projection and Risk Refinement, RMMM Plan.

Unit-III **(8 hours)**

Quality Management Quality Concepts, Software Quality Assurance, Software Reviews, Metrics for Process and Projects.

Unit-IV **(10 hours)**

Design Engineering Design Concepts, Architectural Design Elements, Software Architecture, Data Design at the Architectural Level and Component Level, Mapping of Data Flow into Software Architecture, Modeling Component Level Design.

Unit-V **(10 hours)**

Testing Strategies & Tactics: Software Testing Fundamentals, Strategic Approach to Software Testing, Test Strategies for Conventional Software, Validation Testing, System Testing, Black-Box Testing, White-Box Testing and their type, Basis Path Testing.

***Referenced Books**:*

1. R.S. Pressman, Software Engineering: A Practitioner's Approach (7th Edition), McGraw-Hill, 2009.
2. P. Jalote, An Integrated Approach to Software Engineering (2nd Edition), Narosa Publishing House, 2003.
3. K.K. Aggarwal and Y. Singh, Software engineering (revised 2nd Edition), New Age International Publishers, 2008.
4. R. Mall, Fundamentals of Software Engineering (2nd Edition), Prentice-Hall of India, 2004.

List of Practicals :(30 hours)

A project report needs to be submitted which includes the following:

1. Problem Statement and Process Model
2. Requirement Analysis:
    a. Creating a Data Flow
    b. Data Dictionary, Use Cases
3. Project Management:
    a. Computing FP
    b. Effort
    c. Schedule, Risk Table, Timeline chart
4. Design Engineering:
    a. Architectural Design
    b. Data Design, Component Level Design

DISCIPLINE SPECIFIC CORE COURSE – DSC-11: FULL STACK WEB DEVELOPMENT -1

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
FULL STACK WEB DEVELOPMENT -1	4	3	0	1	Class XII	DSC-08

***Learning objectives*:**

1. To introduce the fundamentals of Internet, and the principles of web design.
2. To construct basic websites using JQuery and AJAX.

Learning Outcomes:

1. *Assimilate and master latest framework like frameworks like js, Node.js, and Mongo DB.*
2. *Build Responsive Web application using Angular Typescript*
3. *Learn Angular Binding and events with templates*
4. *Use Mongo DB queries, tools and apply CRUD operations.*

UNIT I **(10 hours)**

Introduction to JQuery: JQuery Introduction, JQuery Syntax, JQuery Selectors, JQuery Events, JQuery Effects- JQuery Hide/Show, JQuery Fade, JQuery Slide(), JQuery Animate, JQuery Stop(), JQuery Callback, JQuery Chaining, JQuery AJAX- JQuery AJAX Introduction, JQuery Load, JQuery Get/Post, JQuery HTML, JQuery Get, JQuery Set, JQuery Add, JQuery Remove, JQuery CSS Classes, JQuery CSS(), JQuery forms.

UNIT II **(5 hours)**

Introduction to Angular JS: Angular Architecture, Building blocks of Angular, Angular CLI and commands, Angular Modules, Understanding files in Angular, Angular forms.

UNIT III **(10 hours)**

Working of Angular Applications: Angular App Bootstrapping ,Angular Components, Creating A Component Through Angular CLI , Ways to specify selectors , Template and styles , Installing bootstrap to design application , Data Binding , Types of Data Binding , Component Interaction using @Input and @Output decorator , Angular Animations , Component Life-cycle Hooks , Angular Directives.

UNIT IV **(10 hours)**

Introduction of Mongo DB: Overview , Design Goals for Mongo DB Server and Database, Mongo DB Tools , How to modularize code by separating routes , Usage of various Mongo DB Tools available with Mongo DB Package , Mongo DB Development Architecture.

UNIT V **(10 hours)**

Crud Operations : Mongo DB CRUD Introduction, Mongo DB Datatypes , Analogy between RDBMS & Mongo DB Data Model, Mongo DB Data Model (Embedding & Linking), Challenges for Data Modelling in Mongo DB.

References

1. Node.js, Mongo DB and Angular Web Development: The definitive guide to using the MEAN stack to build web applications (Developer's Library) - by Brad Dayley , Addison-Wesley; 2nd edition

2. JQuery Cookbook by Cody Lindley, O'Reilly Media, Inc.

List of Practicals: (30 hours)

A web development project implementing following technologies:

- JQuery
- JavaScript
- Angular JS
- Mongo DB

DISCIPLINE SPECIFIC CORE COURSE – DSC-12:
Data communication and Networks

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Data Communication and Networks	4	3	0	1	Class XII	NIL

***Learning objectives*:**

1. *The objective of the course is to equip the students with a general overview of the concepts and fundamentals of computer networks.*
2. *Familiarize the students with the standard models for the layered approach to communication between machines in a network and the protocols of the various layers*

Learning Outcomes:

1. *Understand the basics of data communication, networking, internet and their importance*
2. *Analyze the services and features of various protocol layers in data networks.*
3. *Differentiate wired and wireless computer networks*
4. *Analyze TCP/IP and their protocols.*
5. *Recognize the different internet devices and their functions.*
6. *Identify the basic security threats of a network.*

UNIT-I **(8 hours)**

Basics of Networking: Network Concept, Benefits of Network, Network classification (PAN, LAN, MAN, WAN), Peer to Peer, Client Server architecture,
Transmission media: Guided & Unguided, Network Topologies.
Networking terms: DNS, URL, client server architecture, TCP/IP, FTP, HTTP, HTTPS, SMTP, Telnet
OSI and TCP/IP Models: Layers and their basic functions and Protocols, Comparison of OSI and TCP/IP. Networking Devices: Hubs, Switches, Routers, Bridges, Repeaters, Gateways and Modems, ADSL.

UNIT-II **(7 hours)**

Ethernet Networking: Half and Full-Duplex Ethernet, Ethernet at the Data Link Layer, Ethernet at the Physical Layer.
Switching Technologies: layer-2 switching, address learning in layer-2 switches, network loop problems in layer-2 switched networks, Spanning-Tree Protocol, LAN switch types and working with layer-2 switches, Wireless LAN

UNIT- III **(15 hours)**

Internet layer Protocol: Internet Protocol, ICMP, ARP, RARP.
IP Addressing: Different classes of IP addresses, Sub-netting for an internet work, Classless Addressing. Comparative study of IPv4 & IPv6.
Introduction to Router Configuration. Introduction to Virtual LAN.

UNIT- IV **(15 hours)**

Transport Layer: Functions of transport layer, Difference between working of TCP and UDP.
Application Layer: Domain Name System (DNS), Remote logging, Telnet, FTP, HTTP, HTTPS.

References:

1. *Tananbaum A.S, "Computer Networks" 3rd Ed. PHI, 1999*
2. *Dr. Sanjay Sharma, "A Course in Computer Network" S. K. Kataria & Sons*
3. *Todd Lammle, "CCNA Cisco Certified Network Associate Study Guide", SYBEX.*
4. *A Forouzan, "Data Communications & Networking",4th Ed, Tata McGraw Hill, 2007*

List of Practicals: (30 hours)

Introduce students to any network simulator tool and do the following:

1. To Study basic network command and Network configuration commands.
2. To study and perform PC to PC communication.
3. Create a Network Using Bluetooth-(Piconet/Scatternet)
3. To create Star topology using Hub and Switch.
4. To create Bus, Ring, Tree, Hybrid, Mesh topologies.
5. Perform an initial Switch configuration.
6. Perform an initial Router configuration.
7. To implement Client Server Network.
8. To implement connection between devices using router.
9. To perform remote desktop sharing within LAN connection.

DSE-02 (a): Big Data

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Big Data	4	2	0	2	Class XII	DSC-04

***Learning objectives*:**

This course gives an overview of Big Data, i.e. storage, retrieval and processing of big data. In addition, it also focuses on the "technologies", i.e., the tools/algorithms that are available for storage, processing of Big Data. It also helps a student to perform a variety of "analytics" on different data sets and to arrive at positive conclusions.

Learning Outcomes:

1. *Perform data gathering of large data from a range of data sources.*
2. *Critically analyze existing Big Data datasets and implementations, taking practicality, and usefulness metrics into consideration.*
3. *Understand and demonstrate the role of statistics in the analysis of large of datasets*
4. *Select and apply suitable statistical measures and analyses techniques for data of various structure and content and present summary statistics*
5. *Understand and demonstrate advanced knowledge of statistical data analytics as applied to large data sets*
6. *Employ advanced statistical analytical skills to test assumptions, and to generate and present new information and insights from large datasets*

Unit-I **(5 hours)**

Introduction to big data: Introduction to Big Data Platform – Challenges of Conventional Systems - Intelligent data analysis – Nature of Data - Analytic Processes and Tools - Analysis vs. Reporting.

Unit-II **(5 hours)**

Mining data streams: Introduction to Streams Concepts – Stream Data Model and Architecture - Stream Computing - Sampling Data in a Stream – Filtering Streams – Counting Distinct Elements in a Stream – Estimating Moments – Counting Oneness in a Window – Decaying Window - Real time Analytics Platform (RTAP) Applications - Case Studies – Real Time Sentiment Analysis- Stock Market Predictions.

Unit-III **(5 hours)**

Hadoop: History of Hadoop- the Hadoop Distributed File System – Components of Hadoop

Analyzing the Data with Hadoop - Scaling Out- Hadoop Streaming- Design of HDFS- Java interfaces to HDFS Basics- Developing a Map Reduce Application-How Map Reduce Works-Anatomy of a Map Reduce Job Run-Failures-Job Scheduling-Shuffle and Sort – Task execution - Map Reduce Types and Formats- Map Reduce Features Hadoop environment.

Unit-IV **(5 hours)**

Frameworks: Applications on Big Data Using Pig and Hive – Data processing operators in Pig – Hive services – HiveQL – Querying Data in Hive - fundamentals of HBase and Zoo Keeper - IBM Info Sphere Big Insights and Streams.

Unit-V **(10 hours)**

Predictive Analytics: Simple linear regression- Multiple linear regression- Interpretation of regression coefficients. Visualizations - Visual data analysis techniques- interaction techniques - Systems and applications.

References:

1. Michael Berthold, David J. Hand, "Intelligent Data Analysis", Springer, 2007.
2. Tom White "Hadoop: The Definitive Guide" Third Edition, O'reilly Media, 2012.
3. Chris Eaton, Dirk DeRoos, Tom Deutsch, George Lapis, Paul Zikopoulos, "Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data", McGraw Hill Publishing, 2012.
4. Anand Rajaraman and Jeffrey David Ullman, "Mining of Massive Datasets", CUP, 2012.
5. Bill Franks, "Taming the Big Data Tidal Wave: Finding Opportunities in Huge Data Streams with Advanced Analytics", John Wiley& sons, 2012.

***List of Practicals*:** **(60 hours)**

1. (i) Perform setting up and Installing Hadoop in its two operating modes:
a) Pseudo distributed,
b) Fully distributed.
(ii) Use web-based tools to monitor your Hadoop setup.
2. (i) Implement the following file management tasks in Hadoop:
a) Adding files and directories
b) Retrieving files
c) Deleting files
3. Run a basic Word Count Map Reduce program to understand Map Reduce Paradigm.
a) Find the number of occurrences of each word appearing in the input file(s).
b) Performing a Map Reduce Job for word search count (look for specific keywords in a file).
4. Install and Run Pig then write Pig Latin scripts to sort, group, join, project, and filter your data.
5. Write a Pig Latin script for finding TF-IDF value for book dataset (A corpus of eBooks available at: Project Gutenberg).
6. Install and Run Hive then use Hive to create, alter, and drop databases, tables, views, functions.

DSE-02 (b): Advance DBMS

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Advance DBMS	4	2	0	2	Class XII	DSC-04

***Learning objectives*:**

1. *Explain and evaluate the fundamental theories for advanced database architectures and query operators.*
2. *Design and implement parallel database systems with evaluating different methods of storing, managing of parallel database.*
3. *Assess and apply database functions of distributed database.*

Learning Outcomes:

1. *Identify advance database concepts and database models.*
2. *Apply and analyze various terms related to transaction management in centralized and distributed database.*
3. *Learn concept of transactional processing and its commands.*
4. *Improve the database design by normalization.*
5. *Administer and analyze database with query optimization technique*

UNIT-I **(5 hours)**

Introduction: Formal review of relational database and FDs Implication, Closure, its correctness.

UNIT-II **(5 hours)**

Normalization: 3NF and BCNF, Decomposition and synthesis approaches, Review of SQL Queries, Basics of query processing, Query optimization, external sorting, file scans.

UNIT-III **(5 hours)**

Transactional Control: Commit, Save point, Rollback, DCL Commands: Grant and Revoke, Types of locks: Row level locks, Table level locks, Shared lock, Exclusive lock, Deadlock.

UNIT-IV **(5 hours)**

Creating and altering Views: Fundamentals of Database Triggers, Creating Triggers, Types of Triggers: Before, after for each row, for each statement, Basics of PL/SQL.

UNIT-V **(10 hours)**

T/O based techniques: Multiversion approaches, Comparison of CC methods, dynamic databases, Failure classification, recovery algorithm, XML and relational databases.

***References*:**

1. R. Ramakrishnan, J. Gehrke, Database Management Systems, McGraw Hill, 2004
2. A. Silberschatz, H. Korth, S. Sudarshan, Database system concepts, 5/e, McGraw Hill, 2008.
3. R. Elmasri, S.B. Navathe Database Systems Models, Languages, Design and application Programming, 6th Edition, Pearson Education, 2013.

***List of Practicals :*(60 hours)**

1. Perform queries for DCL Commands and Locks.
2. Implement authorization, authentication, and privileges on database.
3. Perform queries to Create synonyms, sequence and index.
4. Perform queries to Create, alter and update views.
5. Implement PL/SQL programmes using control structures.
6. Implement PL/SQL programmes using Cursors.
7. Implement PL/SQL programmes using exception handling.
8. Implement user defined procedures and functions using PL/SQL blocks.
9. Perform various operations on packages.
10. Implement various triggers.
11. Practice on functional dependencies
12. Practice on Normalization – using any database perform various normal forms.
13. Practice on transaction processing.

DSE-02 (c): Android Programming

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Android Programming	**4**	**2**	**0**	**2**	**Class XII**	**DSC-05**

***Learning objectives*:**

1. *Creating robust mobile applications and learn how to integrate them with other services.*

Learning Outcomes:

1. *Describe characteristics of Android operating system.*
2. *Describe components of an android applications.*
3. *Design user interfaces using various widgets, dialog boxes, menus.*
4. *Define interaction among various activities/applications using intents, broadcasting, and service.*
5. *Develop Android applications that require database handling.*

UNIT-I **(10 hours)**

Introduction: Review to JAVA & OOPS Concepts, History of Android, Introduction to Android Operating Systems, Android Development Tools, and Android Architecture, Android components including activities, view and view group, services, content providers, broadcast receivers, intents, parcels, instance state.

UNIT-II **(12 hours)**

User Interface Architecture: Application context, intents: explicit intents, returning results from activities, implicit intents, intent filter and intent resolution, and applications of implicit intents, activity life cycle, activity stack, application's priority and its process' states, fragments and its life cycle.

UNIT-III **(12 hours)**

User Interface Design: Layouts, optimizing layout hierarchies, form widgets, text fields, button control, toggle buttons, spinners, images, menu, dialog.

UNIT-IV **(11 hours)**

Broadcast receivers and Database: Broadcast sender, receiver, broadcasting events with intents, notifications and services.

SQLite, Content Values and Cursors, creating SQLite databases, querying a database.

References

1. *Griffiths, D., & Griffiths, D., (2015). Head First Android Development, O'reilly Media.*
2. *Meier, R., (2012). Professional Android™ 4 Application Development. John Wiley & Sons, Inc.*

List of Practicals: **(60 hours)**

1. Create "Hello World" application. That will display "Hello World" in the middle of the screen in the emulator. Also display "Hello World" in the middle of the screen in the Android Phone.
2. Create an application with three buttons (increment, decrement and reset) and a textView aligned vertically. On clicking, increment/decrement button, the value of the textview should increment/decrement by 1while selecting reset button, the value of textview should become zero.
3. Create an application with login module. (Check username and password).
4. Create spinner with strings taken from resource folder (res >> value folder) and on changing the spinner value, Image will change.
5. Create a menu with 5 options and selected option should appear in text box.
6. Create a list of all courses in your college and on selecting a particular course teacher-incharge of that course should appear at the bottom of the screen.
7. Create an application with three option buttons, on selecting a button colour of the screen will change.
8. Create an application to display various activity life cycle and fragment life cycle methods.
9. Create an application with 2 fragments, one to set the background and other to set the fore-color of the text.
10. Create an application with an activity having EditText and a button (with name "Send"). On clicking Send button, make use of implicit intent that uses a Send Action and let user select app from app chooser and navigate to that application.
11. Create a Login application. On successful login, use explicit intent to second activity displaying welcome message (Welcome Username) to the user and a logout button. When user presses logout button, a dialog box with a message ("Are you sure you want to exit?") and two buttons ("Yes" and "No") should appear to confirm logout. On "Yes" button click, go to login activity and on "No", stay on the same activity.
12. Create an application for Broadcast sender and receivers.
13. Create an application to create notification having icon, text and title.
14. Create an application to create services.
15. Create an application to Create, Insert, update, Delete and retrieve operation on database

DISCIPLINE SPECIFIC CORE COURSE – DSC-13: Machine Learning

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Machine Learning	4	3	0	1	Class XII	DSC-01

Learning Objectives:

1. *To understand the basic theory underlying machine learning.*
2. *To be able to formulate machine learning problems corresponding to different applications.*
3. *To understand a range of machine learning algorithms along with their strengths and weaknesses.*
4. *To be able to apply machine learning algorithms to solve problems of moderate complexity.*
5. *To apply the algorithms to a real-world problem, optimize the models learned and report on the expected accuracy that can be achieved by applying the models.*

Learning Outcomes:

1. *Differentiate between supervised and unsupervised learning tasks.*
2. *Appreciate the need of preprocessing, feature scaling and feature selection.*
3. *Understand the fundamentals of classification, regression and clustering*
4. *Implement various machine learning algorithms learnt in the course.*

Unit I **(9 Hours)**

Introduction: Basic definitions and concepts, key elements, supervised and unsupervised learning, introduction to reinforcement learning, applications of ML.

Unit II **(9 Hours)**

Preprocessing: Feature scaling, feature selection methods. Dimensionality reduction (Principal Component Analysis).

Unit III **(9 Hours)**

Regression: Linear regression with one variable, linear regression with multiple variables,gradient descent, over-fitting, regularization. Regression evaluation metrics.

Unit IV **(9 Hours)**

Classification: Decision trees, Naive Bayes classifier, logistic regression, k-nearest neighbor classifier, perceptron, multilayer perceptron, neural networks, back-propagation algorithm, Support Vector Machine (SVM). Classification evaluation metrics.

Unit V **(9 Hours)**

Clustering: Approaches for clustering, distance metrics, K-means clustering, hierarchical clustering.

References

1. Mitchell, T.M. Machine Learning, McGraw Hill Education, 2017.
2. James, G., Witten. D., Hastie. T., Tibshirani., R. An Introduction to Statistical Learning with Applications in R, Springer, 2014.
3. Alpaydin, E. Introduction to Machine Learning, MIT press, 2009.

Practical List: (30 Hours)

Use Python for practical labs for Machine Learning. Utilize publically available datasets from online repositories like https://data.gov.in/ and
https://archive.ics.uci.edu/ml/datasets.php
For evaluation of the regression/classification models, perform experiments as follows:

- Scale/Normalize the data
- Reduce dimension of the data with different feature selection techniques
- Split datasets into training and test sets and evaluate the decision models
- Perform k-cross-validation on datasets for evaluation

Report the efficacy of the machine learning models as follows:

- MSE and R2 score for regression models
- Accuracy, TP, TN, FP, TN, error, Recall, Specificity, F1-score, AUC for classification models.

DISCIPLINE SPECIFIC CORE COURSE – DSC 14: FULL STACK WEB DEVELOPMENT -2

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
FULL STACK WEB DEVELOPMENT -2	4	3	0	1	Class XII	DSC-11

***Learning objectives*:**

1. *Assimilate and master latest framework like frameworks like js, Node.js, and Mongo DB.*

***Learning outcomes*:**

1. *Able to use basic to advanced Node js.*
2. *Integrate Node js with mongo database*
3. *Install and use different tools like Github, Maven and Jenkins.*
4. *Develop a fully functioning website and deploy on a web server.*

UNIT I **(5 hours)**

Introduction to Node JS: What is Node.js, Why Node.js, Node in-built packages (buffer, fs, http, os, path, util, url), Node.js Modules, Import your own Package, Node Package Manager (NPM), Local and Global Packages, File System: Get Input from Users, Pass Multiple Arguments with Yargs, File System Module.

UNIT II **(10 hours)**

Advanced Node JS : Express Framework, Run a Web Server using Express Framework, Routes, Deploy application using PM2 and Nginx, Asynchronous Programming- Call Stack, Callbacks, Callback Queue and Event Loop , Callback Abstraction , Callback Chaining

UNIT III **(10 hours)**

Integration of Node.js with Mongo DB: Inserting Documents, Querying, Updating and Deleting Documents, Connect Mongo DB and Node.js Application, REST API

UNIT IV **(10 hours)**

Overview of Git, Jenkins and Maven: Git- Understand the differences between Git, Github and Gitlab, Install and configure Git for use, Use Git to manage files using CLI commands, Create, Clone and manage repositories.

Jenkins- Jenkins and its architecture, Jenkins tools management, user management in Jenkins

Maven - Maven project structure, maven plugins, Project object model (POM), maven build lifecycle, adding external dependencies to maven pom.xml, maven build and test project

UNIT V **(10 hours)**

Introduction to Docker: Comparing VM and Docker, Docker- an Architectural overview, The Docker Hub A brief Introduction, Preparing docker - machine- Installation and configuration, Start containerizing, Play with docker images, Customizing container on your own, Running Container with Docker - commands, Port forwarding with docker container.

References:

1. Node.js, Mongo DB and Angular Web Development: The definitive guide to using the MEAN stack to build web applications (Developer's Library) - by Brad Dayley , Addison-Wesley; 2nd edition

2. DevOps. Building CI/CD Pipelines with Jenkins, Docker Container, AWS ECS, JDK 11, Git and Maven by John Edward Cooper Berg , Kindle Edition

List of Practicals **(30 hours)**

A web development project implementing technologies such as Node JS, Mongo DB, Angular JS, JQuery, JavaScript, Git, Jenkins and Maven.

DISCIPLINE SPECIFIC CORE COURSE – DSC 15: Minor Project-1

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Minor Project-1	4	0	0	4	Class XII	NIL

Learning Objectives:

The students will undergo one semester of project work based on the concepts studied in a subject of their choice. The objective is to train the students for the industry by exposing them to prototype development of real life software.

Learning Outcomes:

On successful completion of this course, a student will be able to:

1. *Develop a project plan based on informal description of the project.*
2. *Implement the project as a team.*
3. *Write a report on the project work carried out by the team and defend the work done by the team collectively.*
4. *Present the work done by the team to the evaluation committee.*

Each student shall carry out a minor project in the fifth semester. The students will work on any project based on the concepts studied in core/elective/ skill based elective courses. Specifically, the project could be a research study, or a software development project.

In case the student is opting for research project, students are required to select a relevant topic, carryout a detailed literature review followed by a critical analysis or implementation. The conclusions drawn from the analysis/ implementation must also be brought out in the form of a research paper.

PROJECT GROUP ORGANIZATION/PLAN

- Students will initially prepare a synopsis (500 words) and submit it to their respective department/supervisor. Only after obtaining the approval of supervisor the student can initiate the Project work.
- For a given project, the group size could be a maximum of four (04) students.
- Each group will be assigned a teacher as a supervisor who will be responsible for their lab classes.
- A maximum of four (04) projects would be assigned to one teacher.

PROJECT EVALUATION

The project will be evaluated as follows:

(a) Mid-semester evaluation 25% weightage

(b) End-semester evaluation

(i) External Examination 50% weightage

Thesis/Project report - 25% of total marks.

Software Coding

i) Documentation - 10% of total marks.

ii) Software - 15% of total marks.

(ii) Viva-voce **25% weightage**

- Practical/discussion sessions based on the area of the project. Work carried out in each lab session will be assessed out of five marks (zero for being absent). Finally, the marks obtained will be scaled out of a maximum marks of mid-semester evaluation (i.e. 25% of total marks).
- The end-**semester evaluation marks** to be awarded jointly by the examiner and supervisor / mentor.
- The **Mid-semester evaluation** to be awarded by the supervisor/mentor. Work carried out in each lab session will be assessed.
- The students will submit both the soft copy and the hard copy of the report.
- The reports may be retained by the examiners.

PROJECT REPORT

Two copies of the Project Report certified by the supervisor shall be submitted to the Department. The format of report can be downloaded from the website/guide/ coordinator.

DSE-03 (a): Distributed Systems

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Distributed Systems	4	4	0	0	Class XII	DSC-09

***Learning objectives*:**

1. *To provide hardware and software issues in modern distributed systems.*
2. *To get knowledge in distributed architecture, naming, synchronization, consistency and replication, fault tolerance, security, and distributed file systems.*
3. *To analyze the current popular distributed systems such as peer-to-peer (P2P) systems will also be analyze*

Learning Outcomes:

1. *To understand the foundations of distributed systems.*
2. *To learn issues related to clock Synchronization and the need for global state in distributed systems.*
3. *To learn distributed mutual exclusion and deadlock detection algorithms.*

UNIT-I **(15 hours)**

Characterization of Distributed Systems: Introduction, Examples of distributed Systems, Resource sharing and the Web Challenges. Architectural models, Fundamental Models. Theoretical Foundation for Distributed System: Limitation of Distributed system, absence of global clock, shared memory, Logical clocks, Lamport's & vectors logical clocks. Concepts in Message Passing Systems: causal order, total order, total causal order, Techniques for Message Ordering, Causal ordering of messages, global state, and termination detection.

UNIT-II **(15 hours)**

Distributed Mutual Exclusion: Classification of distributed mutual exclusion, requirement of mutual exclusion theorem, Token based and non-token based algorithms, performance metric for distributed mutual exclusion algorithms. Distributed Deadlock Detection: system model, resource Vs communication deadlocks, deadlock prevention, avoidance, detection & resolution, centralized dead lock detection, distributed dead lock detection, path pushing algorithms, edge chasing algorithms.

UNIT -III **(15 hours)**

Agreement Protocols: Introduction, System models, classification of Agreement Problem, Byzantine agreement problem, Consensus problem, Interactive consistency Problem, Solution to Byzantine Agreement problem, Application of Agreement problem, Atomic Commit in Distributed Database system. Distributed Resource Management: Issues in distributed File Systems, Mechanism for building distributed file systems, Design issues in Distributed Shared Memory, Algorithm for Implementation of Distributed Shared Memory.

UNIT-IV **(15 hours)**

Failure Recovery in Distributed Systems: Concepts in Backward and Forward recovery, Recovery in Concurrent systems, obtaining consistent Checkpoints, Recovery in Distributed Database Systems. Fault Tolerance: Issues in Fault Tolerance, Commit Protocols, Voting protocols, Dynamic voting protocols

References

1. Singhal&Shivaratri, "Advanced Concept in Operating Systems", McGraw Hill
2. Ramakrishna,Gehrke," Database Management Systems", McGraw Hill
3. Vijay K.Garg Elements of Distributed Computing, Wiley
4. Coulouris, Dollimore, Kindberg, "Distributed System: Concepts and Design", Pearson Education
5. Tenanuanbaum, Steen," Distributed Systems", PHI.

DSE-03 (b): Artificial Intelligence

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Artificial Intelligence	4	3	0	1	Class XII	DSC-03 DSC-06

Learning Objectives:

1. Study the concepts of Artificial Intelligence.
2. Learn the methods of solving problems using Artificial Intelligence.
3. Learn the knowledge representation techniques, reasoning techniques and planning
4. Introduce the concepts of Expert Systems and machine learning.

Learning Outcomes:

1. Identify problems that are amenable to solutions by specific AI methods.
2. Appreciate the utility of different types of AI agents.
3. Apply different informed search techniques for solving real world problems.
4. Use knowledge representation techniques for AI systems..
5. Understand human level, data driven and end to end approaches to AI.

UNIT-I **(10 Hours)**

Introduction to Artificial Intelligence: background and applications, Turing test, Weak AI, Strong AI, Narrow AI, Artificial General Intelligence, Super AI, rational agent approaches to AI, introduction to intelligent agents, their structure, behavior and task environment , the Present and the Future of AI.

UNIT-II **(12 Hours)**

Problem Solving and Searching Techniques: Problem characteristics, production systems, control strategies, breadth-first search, depth-first search, hill climbing and its variations, heuristics search techniques: best-first search, A* algorithm, constraint satisfaction problem, means-end analysis, introduction to game playing, min-max and alpha-beta pruning algorithms.

UNIT-III **(12 Hours)**

Knowledge Representation: Propositional logic, First-Order Predicate logic, resolution principle, unification, semantic nets, conceptual dependencies, frames, and scripts, production rules, Introduction to Programming in Logic (PROLOG).

UNIT-IV **(11 Hours)**

Understanding Natural Languages: Components and steps of communication, the contrast between formal and natural languages in the context of grammar, Chomsky hierarchy of grammars, parsing, and semantics, Parsing Techniques, Context-Free and Transformational

Grammars, Recursive and Augmented transition nets.

References:

1. Stuart J. Russell and Peter Norvig, Artificial Intelligence - A Modern Approach, Pearson, 4th edition, 2020.
2. Elaine Rich and Kelvin Knight, Artificial Intelligence, 3 rd edition, Tata McGraw Hill, 2010.
3. Ivan Bratko, Prolog Programming for Artificial Intelligence, Addison-Wesley, Pearson Education, 4th edition, 2012.

List of Practicals: (30 hours)

1. Write a prolog program to calculate the sum of two numbers.
2. Write a Prolog program to implement max(X, Y, M) so that M is the maximum of two numbers X and Y.
3. Write a program in PROLOG to implement factorial (N, F) where F represents the factorial of a number N. 60
4. Write a program in PROLOG to implement generate_fib(N,T) where T represents the Nth term of the fibonacci series.
5. Write a Prolog program to implement GCD of two numbers.
6. Write a Prolog program to implement power (Num,Pow, Ans) : where Num is raised to the power Pow to get Ans.
7. Prolog program to implement multi (N1, N2, R) : where N1 and N2 denotes the numbers to be multiplied and R represents the result.
8. Write a Prolog program to implement memb(X, L): to check whether X is a member of L or not.
9. Write a Prolog program to implement conc (L1, L2, L3) where L2 is the list to be appended with L1 to get the resulted list L3.
10. Write a Prolog program to implement reverse (L, R) where List L is original and List R is reversed list.
11. Write a program in PROLOG to implement palindrome (L) which checks whether a list L is a palindrome or not.
12. Write a Prolog program to implement sumlist(L, S) so that S is the sum of a given list L.
13. Write a Prolog program to implement two predicates evenlength(List) and oddlength(List) so that they are true if their argument is a list of even or odd length respectively.
14. Write a Prolog program to implement nth_element (N, L, X) where N is the desired position, L is a list and X represents the Nth element of L.
15. Write a Prolog program to implement maxlist(L, M) so that M is the maximum number in the list.
16. Write a prolog program to implement insert_nth (I, N, L, R) that inserts an item I into Nth position of list L to generate a list R.
17. Write a Prolog program to implement delete_nth (N, L, R) that removes the element on Nth position from a list L to generate a list R.
18. Write a program in PROLOG to implement merge (L1, L2, L3) where L1 is first ordered list and L2 is second ordered list and L3 represents the merged list.

DSE-03 (c): Design and Analysis of Algorithms

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Design and Analysis of Algorithms	4	3	0	1	Class XII	DSE-07

***Learning objectives*:**

1. *Introduces the recurrence relations for analyzing the algorithms.*
2. *Introduces the graphs and their traversals.*
3. *Describes major algorithmic techniques (divide-and-conquer, greedy, dynamic programming, Brute Force, Transform and Conquer approaches) and mention problems for which each technique is appropriate.*
4. *Describes how to evaluate and compare different algorithms using worst-case, average-case and best-case analysis.*

Learning Outcomes:

1. *Compute the asymptotic time complexity of algorithms*
2. *Prove correctness of algorithms*
3. *Use appropriate algorithm design technique(s) for solving a given problem*
4. *Appreciate the difference between tractable and intractable problems*

UNIT-1 **(10 hours)**

Sorting: Selection. Insertion Sort, Selection Sort, Bubble Sort, Heap sort, Linear Time Sorting, Selection Problem, running time analysis and correctness.

UNIT-II **(10 hours)**

Graphs: Review of graph traversals, graph connectivity, testing bi-partiteness, Directed Acyclic Graphs and Topological Ordering.

UNIT-III **(13 hours)**

Divide and Conquer. Introduction to divide and conquer technique, Merge Sort, Quicksort, Maximum-subarray problem.

Intractability: Decision vs optimization problems, NP as a class of problems, NP-hardness, NP-completeness with examples.

UNIT-IV **(12 hours)**

Greedy and dynamic Algorithms: Introduction to the Greedy algorithm design approach, application to minimum spanning trees, fractional knapsack problem.

Introduction to the Dynamic Programming approach, application to subset sum, integer knapsack problem.

References

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein C., Introduction to Algorithms. 3rd edition. Prentice Hall of India. 2010.
2. Kleinberg, J., Tardos, E. Algorithm Design. 1st edition. Pearson. 2013.

List of Practicals **(30 hours)**

A practical implementation of various algorithmic techniques such as sorting, graphs,greedy and dynamic programming.

DISCIPLINE SPECIFIC CORE COURSE – 16: Cloud Computing

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Cloud Computing	**4**	**3**	**0**	**1**	**Class XII pass with Mathematics**	**NIL**

Learning Objectives:

1. *To provide the students basic understanding about cloud and virtualization along with it how one can migrate over it.*
2. *To provide students concepts of security and privacy in a cloud.*

Learning Outcome:

1. *The fundamental ideas behind Cloud Computing, the evolution of the paradigm, its applicability; benefits, as well as current and future challenges;*
2. *The basic ideas and principles in data center design; cloud management techniques and cloud software deployment considerations;*
3. *Different CPU, memory and I/O virtualization techniques that serve in offering software, computation and storage services on the cloud; Software Defined Networks (SDN) and Software Defined Storage (SDS);*
4. *Could storage technologies and relevant distributed file systems, NoSQL databases and object storage;*
5. *The variety of programming models and develop working experience in several of them.*

Unit I **(7 Hours)**

Evolution of Cloud Computing: Trends of computing, Introduction to distributed computing, cloud computing, Cloud Based Application Development Approach Vs. Traditional Application Development Approach, What's cloud computing, Properties & Characteristics, Service models, Deployment models, SLA(Service Level Agreements), SLA at various levels, SOA(Service oriented Architecture), SOA characteristics

Unit II **(8 Hours)**

Cloud Computing Architectural Framework: Infrastructure as a Service (IAAS), Platform as a Service (PAAS), Software as a Service (SAAS), cloud computing vendors, Cloud Computing threats, Cloud Reference Model, The Cloud Cube Model, issues in Cloud Computing ,Managing and administrating the cloud services and cloud resources, Virtualization -Hypervisor Architecture, Hardware Virtualization, Software Virtualization, Memory Virtualization, Storage Virtualization, Data Virtualization, Network Virtualization,
Virtualization Security Recommendations

Unit III **(8 Hours)**

Security in Cloud: Infrastructure security: Network Level, Host Level and Application Level

Security and Storage: Aspects of Data Security, Data control, Network Security, Host Security, Data Security Mitigation, Encryption, storage- confidentiality, integrity, and availability.
Security Management in the Cloud: Security Management Standards, Availability Management- PAAS, SAAS, IAAS, Access Control, Security Vulnerability, Patch and Configuration Management.

Unit IV **(7 Hours)**

Privacy in Cloud: Data Life-Cycle, Key Privacy Concerns in the Cloud, Responsibility for protecting Privacy, Risk Management and Compliance in relation to Cloud Computing, Legal and Regulatory Implications. Disaster Recovery: Disaster recovery planning, Disaster in Cloud, Disaster Management

Unit V **(15 Hours)**

Case study: Hadoop- architecture, Hadoop Distributed file system, map- reduce model, getting started with the Hadoop, Amazon EC2 / S3 and EC2 Commands. Introduction of MS Windows Azure, Google Apps / Google Docs.

Reference Books:

1. Tim Mather, Subra Kumaraswamy, Shahed Latif, "Cloud Security and Privacy," O Reilly
2. George Reese, "Cloud Application Architectures," O Reilly
3. David S. Linthicum, "Cloud Computing and SOA Convergence in your Enterprise, A Step by Step Guide, "Pearson
4. Dr. Gautam Shroff,"Enterprise Cloud Computing Technology, Architecture, Applications", Cambridge University Press.

List of practicals (30 Hours)

1. What are the fundamental differences between centralized and distributed computing?
2. How do elasticity and scalability differ in the context of cloud computing?
3. How to set up an Amazon EC2 instance?
4. Design a basic service-oriented architecture for a simple e-commerce website?
5. Explain the role of firewalls in cloud network security.
6. Launch a Linux Virtual Machine
7. Host a Static Website
8. Create an Amazon Elastic Kubernetes Service (EKS) and S3 Bucket
9. Writing IAM Policies: How to Grant Access to an Amazon S3 Bucket

DISCIPLINE SPECIFIC CORE COURSE – 17: Information Security

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Information Security	**4**	**3**	**0**	**1**	**Class XII pass with Mathematics**	DSC-12

Learning Objectives:

1. *To make a student learn basic principles of information security.*
2. *To familiarize students with cryptography, authentication and access control methods along with software security.*
3. *To touch upon the implications of security in cloud and Internet of Things (IoT).*
4. *To discuss potential security threats and vulnerabilities of systems along with their impacts and countermeasures.*

Learning Outcome:

1. *Identify the major types of threats to information security*
2. *Describe the role of cryptography in security*
3. *Discover the strengths and weaknesses of private and public key cryptosystems*
4. *Identify and apply various access control and authentication mechanisms*
5. *Discuss data and software security and, related issues*
6. *Explain network security threats and attacks*

Unit I **(5 Hours)**

Overview Computer Security Concepts, Threats, Attacks, Security Functional Requirements,Fundamental Security Design Principles, Attack Surfaces and Attack Trees.

Unit II **(10 Hours)**

Cryptographic tools Confidentiality with Symmetric Encryption, Message Authentication and Hash Functions, Public-Key Encryption, Digital Signatures and Key Management, Random and Pseudorandom Numbers, DES (Data Encryption Standard), RSA , Diffie-Hellman key exchange, Post quantum cryptography.

Unit III **(5 Hours)**

Data Security User authentication and Access Control, Database and Data Center Security

Unit IV **(12 Hours)**

Software Security Types of Malicious Software, Threats, Viruses, Worms, SPAM E-Mail, Trojans, Payload, System Corruption, Payload, Attack Agent, Zombie, Bots, Payload, Information Theft, Key-loggers, Phishing, Spyware, Payload Stealthing Backdoors, Rootkits, Countermeasures. Overflow Attacks - Stack Overflows, Buffer Overflows. Handling Program Input, Writing Safe Program Code, Interacting with the Operating System and Other Programs.

Unit V **(13 Hours)**

Network Security Denial-of-Service Attacks, Flooding Attacks, Distributed Denial-of-Service Attacks, Overview of Intrusion Detection, Honeypots, Firewalls, Secure Email and S/MIME, Secure Sockets Layer (SSL) and Transport Layer Security (TLS), HTTPS, IPv4 and IPv6 Security, Public-Key Infrastructure.

References:

1. Stallings, W. and Brown L. (2018) Computer Security: Principles and Practice, Fourth edition, Pearson Education.

2. Pfleeger, C.P., Pfleeger, S.L., & Margulies, J. (2015). Security in Computing. 5th edition. Prentice Hall

3. Lin, S. & Costello, D. J. (2004). Error Control Coding: Fundamentals and applications. 2nd edition. Pearson Education

List of Practicals **(30 hours)**

1. Demonstrate the use of Network tools: ping, ipconfig, ifconfig, tracert, arp, netstat,whois
2. Use of Password cracking tools : John the Ripper, Ophcrack. Verify the strength of passwords using these tools.
3. Use nmap/zenmap to analyse a remote machine.
4. Use Burp proxy to capture and modify the message.
5. Demonstrate sending of a protected word document.
6. Demonstrate sending of a digitally signed document.
7. Demonstrate sending of a protected worksheet.
8. Demonstrate use of gpg utility for signing and encrypting purposes.

DISCIPLINE SPECIFIC CORE COURSE – 18: MINOR PROJECT-2

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
MINOR PROJECT-2	**4**	**0**	**0**	**4**	**Class XII pass with Mathematics**	**DSC-15**

Learning Objectives:

The students will undergo one semester of project work based on the concepts studied in a subject of their choice. The objective is to train the students for the industry by exposing them to prototype development of real life software.

Learning Outcomes:

On successful completion of this course, a student will be able to:

1. *Develop a project plan based on informal description of the project.*
2. *Implement the project as a team.*
3. *Write a report on the project work carried out by the team and defend the work done by the team collectively.*
4. *Present the work done by the team to the evaluation committee.*

Each student shall carry out a minor project in the sixth semester that can be a continuation of advancement in Minor Project-1 or can be done from scratch. The students will work on any project based on the concepts studied in core/elective/ skill based elective courses. Specifically, the project could be a research study, or a software development project.

In case the student is opting for research project, students are required to select a relevant topic, carryout a detailed literature review followed by a critical analysis or implementation. The conclusions drawn from the analysis/ implementation must also be brought out in the form of a research paper.

PROJECT GROUP ORGANIZATION/PLAN

- Students will initially prepare a synopsis (500 words) and submit it to their respective department/supervisor. Only after obtaining the approval of supervisor the student can initiate the Project work.
- For a given project, the group size could be a maximum of four (04) students.
- Each group will be assigned a teacher as a supervisor who will be responsible for their lab classes.
- A maximum of four (04) projects would be assigned to one teacher.

PROJECT EVALUATION

The project will be evaluated as follows:

(a) Mid-semester evaluation 25% weightage

(b) End-semester evaluation

(i) External Examination 50% weightage

Thesis/Project report - 25% of total marks.

Software Coding

i) Documentation - 10% of total marks.

ii) Software - 15% of total marks.

(ii) Viva-voce **25% weightage**

- Practical/discussion sessions based on the area of the project. Work carried out in each lab session will be assessed out of five marks (zero for being absent). Finally, the marks obtained will be scaled out of a maximum marks of mid-semester evaluation (i.e. 25% of total marks).
- The end-**semester evaluation marks** to be awarded jointly by the examiner and supervisor / mentor.
- The **Mid-semester evaluation** to be awarded by the supervisor/mentor. Work carried out in each lab session will be assessed.
- The students will submit both the soft copy and the hard copy of the report.
- The reports may be retained by the examiners.

PROJECT REPORT

Two copies of the Project Report certified by the supervisor shall be submitted to the Department. The format of report can be downloaded from the website/guide/ coordinator.

DSE – 04 (a): Deep Learning

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Deep Learning	**4**	**2**	**0**	**2**	**Class XII pass**	DSC-03 DSC-13

Learning Objectives:

To introduce students to deep learning algorithms and their applications in order to solve real problems.

Learning Outcomes:

1. *Describe the feed-forward and deep networks.*
2. *Design single and multi-layer feed-forward deep networks and tune various hyper parameters.*
3. *Implement deep neural networks to solve a problem*
4. *Analyze performance of deep networks.*
5. *Use pre-trained models to solve a problem*

UNIT-I **(6 hours)**

Introduction to neural networks: Artificial neurons, perceptron, computational models of neurons, Structure of neural networks, Multilayer feed-forward neural networks (MLFFNN), Back-propagation learning, Empirical risk minimization, bias-variance tradeoff, Regularization, output units: linear, softmax , hidden units:tanh, RELU

UNIT-II **(6 hours)**

Deep neural networks: Difficulty of training DNNs, Greedy layerwise training, Optimization for training DNN's, Newer optimization methods for neural networks (AdaGrad, RMSProp, Adam), Regularization methods (dropout, drop connect, batch normalization).

UNIT-III **(6 hours)**

Convolution neural networks (CNNs): Introduction to CNN - convolution, pooling, Deep CNNs - LeNet, AlexNet. Training CNNs, weights initialization, batch normalization, hyper parameter optimization, Understanding and visualizing CNNs, Using a pre trained convnet

UNIT-IV **(6 hours)**

Recurrent neural networks (RNNs): Sequence modeling using RNNs, Back propagation through time, Longshot Term Memory (LSTM), Bidirectional RNN, Bidirectional LSTM

UNIT-V **(6 hours)**

Unsupervised deep learning: Auto-encoders, Generative Adversarial Networks. Applications of Deep learning - Computer vision, Speech recognition and NLP.

References:

1. *Ian Goodfellow, Yodhua Bengio and Aaron Courville, Deep Learning, MITPress Book*
2. *Francois Chollet, Deep Learning with python second edition, Meaning Publications Co.*
3. *Bunduma, N. (2017). Fundamentals of Deep Learning. O'reilly Books.*
4. *Heaton, J. (2015). Deep Learning and Neural Networks, Heaton Research Inc.*

***List of Practicals*: (60 Hours)**

1. Implement a feed-forward neural networks for classifying movie reviews as positive or negative(using IMDB dataset)
2. Implement a deep-neural feed-forward network for estimating the price of house, given real-estate data(Boston Housing Price)
3. Implement a deep-neural network for classifying news wires by topic (Reuter's dataset).
4. Implement CNN for classifying MNIST dataset
5. Create a model for time-series forecasting using RNN/LSTM
6. Implement an auto-encoder

DSE – 04 (b): Internet of Things (IoT)

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Internet of Things(IoT)	**4**	**2**	**0**	**2**	**Class XII pass**	**DSC-01**

Learning Objectives

1. *To make students understand what IoT is and how it works today*
2. *To make students aware of different applications of IoT.*
3. *To introduce students to technologies and smart systems under IoT*

Learning Outcomes:

1. *Able to understand the application areas of IOT ·*
2. *Able to realize the revolution of Internet in Mobile Devices, Cloud & Sensor Networks*
3. *Able to understand building blocks of Internet of Things and characteristics.*

UNIT-I **(5 Hours)**

Introduction to IoT: Definition and Characteristics, Physical Design Things- Protocols, Logical Design- Functional Blocks, Communication Models- Communication APIs- Introduction to measure the physical quantities.

UNIT-II **(5 Hours)**

IoT Enabling Technologies - Wireless Sensor Networks, Cloud Computing Big Data Analytics, Communication Protocols- Embedded System- IoT Levels and Deployment Templates.

UNIT-III **(10 Hours)**

Introduction to Smart Systems using IoT: IoT Design Methodology- IoT Boards (Raspberry Pi, Arduino) and IDE - Case Study: Weather Monitoring- Logical Design using Python, Data types & Data Structures- Control Flow, Functions- Modules- Packages, File Handling - Date/Time Operations, Classes- Python Packages of Interest for IoT.

UNIT-IV **(5 Hours)**

Sensing and Sensors: Wireless Sensor Networks, Challenges and Constraints, Introduction – Fundamentals of MAC Protocols – MAC protocols for WSN – Sensor MAC Case Study.

UNIT-V **(5 Hours)**

Applications: Home Automation, Smart Cities, Environment, Energy, Retail, Logistics, Agriculture, Industry, Health and Lifestyle, IoT and M2M

References:

1. Michael Miller, The Internet of Things, Pearson Education, 2015.
2. Arshdeep Bahga and Vijay Madisetti, Internet of Things: Hands-on Approach, Hyderabad University Press, 2015.
3. Greengard, Samuel. The internet of things. MIT press, 2015.

List of Practicals: (60 Hours)

1. Familiarization with Arduino/Raspberry Pi and perform necessary software installation.
2. To interface LED/Buzzer with Arduino/Raspberry Pi and write a program to turn ON LED for 1 sec after every 2 seconds.
3. To interface Push button/Digital sensor (IR/LDR) with Arduino/Raspberry Pi and write a program to turn ON LED when push button is pressed or at sensor detection.
4. To interface DHT11 sensor with Arduino/Raspberry Pi and write a program to print temperature and humidity readings.
5. To interface motor using relay with Arduino/Raspberry Pi and write a program to turn ON motor when push button is pressed.
6. To interface OLED with Arduino/Raspberry Pi and write a program to print temperature and humidity readings on it.
7. To interface Bluetooth with Arduino/Raspberry Pi and write a program to send sensor data to smartphone using Bluetooth.
8. To interface Bluetooth with Arduino/Raspberry Pi and write a program to turn LED ON/OFF when '1'/'0' is received from smartphone using Bluetooth.
9. Write a program on Arduino/Raspberry Pi to upload temperature and humidity data to thingspeak cloud.
10. Write a program on Arduino/Raspberry Pi to retrieve temperature and humidity data from thing speak cloud.
11. To install MySQL database on Raspberry Pi and perform basic SQL queries.
12. Write a program on Arduino/Raspberry Pi to publish temperature data to MQTT broker.
13. Write a program on Arduino/Raspberry Pi to subscribe to MQTT broker for temperature data and print it.
14. Write a program to create TCP server on Arduino/Raspberry Pi and respond with humidity data to TCP client when requested.
15. Write a program to create UDP server on Arduino/Raspberry Pi and respond with humidity data to UDP client when requested.

DSE – 04 (c): SOFTWARE TESTING

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Software Testing	**4**	**3**	**0**	**1**	**Class XII pass**	**NIL**

Learning Objectives

1. To study fundamental concepts in software testing
2. To discuss various software testing issues and solutions in software unit test, integration and system testing.
3. To expose the advanced software testing topics, such as object-oriented software testing methods.

Learning Outcomes:

1. List a range of different software testing techniques and strategies and be able to apply.
2. Distinguish characteristics of structural testing methods.
3. Demonstrate the integration testing which aims to uncover interaction and compatibility problems as early as possible.
4. Discuss about the functional and system testing methods.
5. Demonstrate various issues for object oriented testing.

UNIT-I **(12 Hours)**

Review of Software Engineering: Overview of Software Evolution, SDLC, Testing Process, Terminologies in Testing: Error, Fault, Failure, Verification, Validation, Difference between Verification and Validation, Test Cases, Testing Suite, Test, Oracles, Impracticality of Testing All Data; Impracticality of Testing All Paths. Verification: Verification Methods, SRS Verification, Source Code Reviews, User Documentation Verification, Software, Project Audit, Tailoring Software Quality Assurance Program by Reviews, Walkthrough, Inspection and Configuration Audits.

UNIT-II **(13 Hours)**

Functional Testing: Boundary Value Analysis, Equivalence Class Testing, Decision Table Based Testing, Cause Effect Graphing Technique. Structural Testing: Control Flow Testing, Path Testing, Independent Paths, Generation of Graph from Program, Identification of Independent Paths, Cyclomatic Complexity, Data Flow Testing, Mutation Testing.

UNIT-III **(8 Hours)**

Regression Testing: What is Regression Testing? Regression Test cases selection, reducing the number of test cases, Code coverage prioritization technique. Reducing the number of test cases: Prioritization guidelines, Priority category, Scheme, Risk Analysis.

UNIT-IV **(7 Hours)**

Software Testing Activities: Levels of Testing, Debugging, Testing techniques and their applicability, Exploratory Testing Automated Test Data Generation: Test Data, Approaches to test data generation, test data generation using genetic algorithm, Test Data Generation Tools, Software Testing Tools, and Software test Plan

UNIT-V **(5 Hours)**

Object Oriented Testing: Definition, Issues, Class Testing, Object Oriented Integration and System Testing. Testing Web Applications: Web Testing, User Interface Testing, Usability Testing, Security Testing, Performance Testing, Database testing, Post Deployment Testing.

References:

1. *Yogesh Singh, "Software Testing", Cambridge University Press, New York, 2012*
2. *K.K. Aggarwal & Yogesh Singh, "Software Engineering", New Age International Publishers, New Delhi, 2003.*
3. *Roger S. Pressman, "Software Engineering – A Practitioner's Approach", Fifth Edition, McGraw-Hill International Edition, New Delhi, 2001.*
4. *Marc Roper, "Software Testing", McGraw-Hill Book Co., London, 1994.*
5. *M.C. Trivedi, Software Testing & Audit, Khanna Publishing House*
6. *Boris Beizer, "Software System Testing and Quality Assurance", Van Nostrand Reinhold, New York, 1984.*

Practicals (30 Hours)

Practicals related to basic path testing and other testing techniques.