MCA (2 years) - Programme Structure

SEMESTER-1						SEMESTER-2					
Course_Name		**Contact_Hours/week**			**Credits**	**Course_Name**		**Contact_Hours/ week**			**Credits**
		L	**T**	**P**				**L**	**T**	**P**	
CSC-101	*Data Structures & Algorithms*	*3*	*0*	*0*	*3*	*CSC-201*	*Web Development*	*3*	*0*	*0*	*3*
CSC-102	*Object Oriented Concepts*	*2*	*0*	*0*	*2*	*CSC-202*	*Database Management Systems*	*3*	*0*	*0*	*3*
CSC-103	*Operating Systems*	*4*	*0*	*0*	*4*	*CSC-203*	*Mathematics for Computer Science*	*4*	*0*	*0*	*4*
CSC-104	*Internet Technologies*	*3*	*0*	*0*	*3*	*CSC-204*	*Software Design & Engineering LAB*	*0*	*2*	*4*	*4*
CSC-105	*Data Structures & Algorithms LAB*	*1*	*0*	*4*	*3*	*CSC-205*	*Web Development LAB*	*1*	*0*	*4*	*3*
CSC-106	*Object Oriented Programming LAB*	*1*	*0*	*4*	*3*	*CSC-206*	*Database Management Systems LAB*	*1*	*0*	*4*	*3*
CSC-107	*LINUX LAB*	*2*	*0*	*4*	*4*		*Elective - 1*	*4*	*0*	*0*	*4*
CSC-108	*Communication Skills*	*2*	*0*	*0*	*2*			**Total_Credits**			**24**
		Total_Credits			**24**						

SEMESTER-3						SEMESTER - 4					
Course_Name		**Contact_Hours/week**			**Credits**	**Course_Name**					**Credits**
		L	**T**	**P**							
CSC-301	*Machine Learning*	*3*	*0*	*0*	*3*	*Industry Internship/ Project*					*16*
CSC-302	*Modern Development Platforms*	*3*	*0*	*0*	*3*			**Total_Credits**			**16**
CSC-303	*Machine Learning LAB*	*1*	*0*	*4*	*3*						
CSC-304	*Modern Development Platforms LAB*	*1*	*0*	*4*	*3*						
	Elective - 2	*4*	*0*	*0*	*4*	**Total Credits**	**72+16 = 88**				
	Elective - 3	*4*	*0*	*0*	*4*						
	Elective - 4	*4*	*0*	*0*	*4*						
		Total_Credits			**24**						

Programme: MCA
Course Code: CSC-101 **Title of Course:** Data Structures & Algorithms
Number of Credits: 3 (3L-0T-0P) **Contact Hours:** 36 hours (36L-0T-0P)
Effective from AY: 2021-22

Prerequisites for the course	Program Prerequisites	
Objectives	The aim of the course is to emphasize the importance of data structures in implementing efficient algorithms. It provides an exposure to various algorithm design techniques and an introduction to algorithm analysis.	
Content	**Revision of Programming & Data Structures** Problem solving, Data Types: Primitive and User DefinedSelection Constructs, Repetition Constructs, Recursion Pointers Algorithm Representation: - Pseudocode and flowcharts Three level Approach Abstract Data Types (ADTs) Basic Linear Data Structures (LinkedList, Stack, Queue)	6 hours
	Algorithm Analysis Analysis of Algorithms Algorithm Complexity: Space and Time Cases of Complexity: Best, Worst and Average Growth of Functions: Asymptotic Notation	3 hours
	Advanced Linear Data Structures Variants of Linked List and its applications (e.g. Polynomial addition, Sparse matrices) Applications of stacks (e.g. Infix-to-Postfix conversion, Evaluating Postfix Expressions, Bracket Matching) Variants of Queue and Applications	5 hours
	Nonlinear Data Structures: Trees: Binary Search Trees, AVL Trees, B-trees & variants. Tree Traversal Algorithms Heaps and its applications (e.g. implementation of Priority Queue) Graph: Adjacency Matrix and Adjacency List Representations Graph Traversal Algorithms: Breadth First Search and Depth First Search	12 hours

	Divide & Conquer Strategy Algorithms based on Divide and Conquer Strategy: Sorting Algorithms (QuickSort, MergeSort) Binary Search	3 hours

	Greedy Algorithms 1. Huffman Coding Algorithm 2. Minimum Cost Spanning Tree (Prim's, Kruskal's) 3. Single Source Shortest Path (Dijkstra's)	4 hours
	Dynamic Programming Coin Change Problem Longest Common Subsequence All-pair shortest Path (floyd-warshall)	3 hours
Pedagogy	• Lectures/Tutorials/Assignments/Quizzes • Each data structure should be explained along with implementation of its ADT, its applications and complexity	
References/ Readings	1. Horowitz, Ellis, Sartaj Sahni, and Susan Anderson-Freed. "Fundamentals of data structures in C" WH Freeman & Co., Latest Edition. 2. Thomas H. Cormen, Charles E. Leiserson, et al "Introduction to Algorithms", Latest Edition 3. Allen, Weiss Mark. Data structures and algorithm analysis in C. Pearson Education India, Latest Edition. 4. Dasgupta, Papadimitriou, and Vazirani,Algorithms, by McGraw-Hill. 5. Jeri R. Hanly and Eliot B. Koffman "Problem Solving and Program Design in C" Pearson Education, VII Edition, 2012 6. R.G.Dromey "How to Solve it by Computer ", PHI , Latest Edition	
Learning Outcomes	Upon successful completion of the course, a student will be able to • Implement common data structures such as lists, stacks, queues, graphs, and binary trees for solving programming problems. • Identify and use appropriate data structures in the context of solution to a given problem. • Be able to analyze the complexity of a given algorithm	

Programme: MCA
Course code: CSC-102 **Title of course:** Object Oriented Concepts
Number of credits: 2 (2L-0T-0P) **Total contact hours:** 24 hours (24L-0T-0P)
Effective from AY: 2021-22

Prerequisites for the course	Program Prerequisites	
Objectives	Aim of this course is to introduce the learner to the object oriented paradigm.	
Content	**Classes and objects** Programming paradigm; procedural to object oriented Class; attributes & methods; classes as modules & types; uniform type system, wrapper type classes Object; object references; objects instantiation & interaction; constructor & destructor; pass-by-reference & pass-by-value Object copying & cloning; composite objects Static & non-static members Enumeration & Annotations	7 hours
	Object oriented principles Encapsulation Inheritance; types of inheritance; diamond problem Abstraction; virtual methods Polymorphism; overloading and overriding	6 hours
	Object oriented features Interfaces Access modifiers Errors & Exceptions; user-defined exceptions Collections Anonymous & Inner classes Type parametric polymorphism (e.g. Generics in Java & Templates in C++)	6 hours
	Advanced features Persistence & Serialization; JSON User packages & custom libraries; reflection Predicates & streams Lambda functions	5 hours
Pedagogy	Hands-on assignments / tutorials / peer-teaching / flip classroom. Concepts can be explained using UML class diagrams.	
References/ Readings	**Main Reading** 1. Timothy Budd, “An Introduction to Object Oriented Programming”, Pearson Education, 3rd Edition 2. Brett D. McLaughlin, Gary Pollice & David West, “Head First Object-Oriented Analysis Design”,	

	O'Reilly 3. Ken Arnold, James Gosling, David Holmes, "The Java Programming Language", Addison-Wesley Professional 4. Stanley Lippman, "C++ Primer", Addison Wesley 5. Cay S. Horstmann, "Core Java Volume I—Fundamentals", Pearson 6. Herbert Schildt, "Java: The Complete Reference", Oracle Press 7. Joshua Bloch, "Effective Java", Addison Wesley 8. Kathy Sierra & Bert Bates, "Head First Java", O'Reilly 9. Bjarne Stroustroup, "The C++ Programming Language", Addison Wesley 10. https://www.tutorialspoint.com/java/index.htm	
Learning Outcomes	1. Learner will appreciate mapping real-world scenarios in the object-oriented world 2. Learner will understand object-oriented principles 3. Learner will be able to design object oriented softwares 4. Learner will be able to analyse	

Programme: MCA
Course code: CSC-103 **Title of course:** Operating Systems
Number of credits: 4(4L-0T-0P) **Total contact hours:** 48 (48L-0T-0P)
Effective from AY: 2021-22

Prerequisites for the course	Programme Prerequisites	
Objectives	This course focuses on the principles and understanding of the functionality of an operating system and evaluates their trade-off in various environments.	
Content	**Introduction and Systems Structures** Computing Environments, Operating-systems Services, System Calls, System Programs, Virtual Machines, monolithic and micro kernel architectures	4 hours
	Process Management Process-Concept and states, Process Creation and Control, Scheduling Criteria, Scheduling Algorithms, MultiLevel Queues., Multiple-processor scheduling, real time CPU scheduling	6 hours
	Threads Motivation and Challenges, Multithreading Models, Threading Issues, Thread libraries, Thread scheduling	4 hours
	Process Synchronization Cooperating processes and Race Conditions, The critical-section problem, Peterson's solution, mutex locks, Synchronization Hardware, Semaphores and theirImplementation, Classic problems of synchronization	6 hours
	Inter process Communication, Overview of IPC, Examples of IPC Systems, Communication in Client Server Systems.	3 hours
	Deadlocks System Model, Deadlock characterization, Methods for Handling Deadlocks, Deadlock Prevention, Deadlock Avoidance, Deadlock Detection, Recovery From Deadlock	4 hours
	Memory Management Hardware Support, Address Binding, Swapping, Contiguous Memory Allocation, Fragmentation, memory Protection, Paging, Structure of the page table, Segmentation, Example: Intel architecture	6 hours
	Virtual-Memory Management Background, Demand Paging, Copy-on-write, Page Replacement algorithms, Allocation of Frames, Thrashing, Allocating Kernel Memory	6 hours

	File System File Concept, Access Methods, Directory Structure, File-system mounting, File sharing, Protection; Virtual file systems, Implementing File Systems, Directory implementation, Allocation Methods, Free-space Management, Efficiency and performance, Recovery, Log-structured file systems	6 hours
	Secondary-storage Structure Overview of Mass-storage Structure, Disk Structure, Disk Attachment, Disk Scheduling ,Disk Management, Swap-Space Management	3 hours
Pedagogy	lectures/ tutorials/assignments/class presentations and debates/peer reviews/self-study.	
References/ Readings	**Main Reading** 1. Silberschatz ,Galvin and Gagne , Operating systems Principles – 8th edition or Later(Wiley Asia Student Edition) 2. Deitel H.M., "An Introduction to Operating Systems", Addison Wesley Publishers Company, Latest Edition 3. Milenkovic M., "Operating Systems : Concepts and Design", McGraw Hill International Edition Computer Science series ; Latest Edition 4. Tanenbaum A. S., Modern Operating Systems", Prentice Hall of India Pvt. Ltd.,Latest Edition 5. Operating Systems – a modern perspective - Gary Nutt , Addison Wesley, Latest Edition	
Learning Outcomes	1. To understand the services provided by and the design of an operating system. 2. To understand the structure and organization of the file system. 3. To understand what a process is and how processes are synchronized and scheduled. 4. To understand different approaches to memory management. 5. Students should be able to understand the implementation and use of system calls for managing processes, memory and the file system. 6. Students should understand the data structures and algorithms used to implement an OS. 7. Evaluate operating system implementations	

Programme: M.C.A

Course Code: CSC-104 **Title of the Course:** Internet Technologies

Number of Credits: 3 (3L-0T-0P) **Contact Hours:** 36 hours (36L-0T-0P)

Effective from AY: 2021-22

Prerequisites for the course:	Programme requisites	
Objectives:	The objective to introduce the TCP/IP architecture and allied protocols of the Internet by following a top-down approach.	
Content:	**Computer Networks and the Internet:** Networking and Inter-networks, Internetworking devices, Internet: Network edge and Network core. TCP/IP protocol stack: Protocol stack, Connection oriented, connectionless services, Packet switching, circuit switching, Delay, Loss, and Throughput in Packet-Switched Networks.	6 hours
	Application layer: Principles of Application Layer Protocols, the Web and HTTP, MIME, mail access protocols, DNS, Peer to Peer Applications.	8 hours
	Transport layer: Transport-layer services, Multiplexing and demultiplexing, UDP protocol, Principles of reliable data transfer, Connection oriented transport - TCP protocol, Principles of congestion control, TCP congestion control.	6 hours
	Network layer: Packet switching: virtual circuit & datagram networks, The Internet Protocol (IP): Forwarding and Addressing in the Internet, route aggregation, subnetting, CIDR, IP datagram, fragmentation, NAT, DHCP, ICMP. Routing protocols: shortest path, link state routing algorithm, distance vector routing. Internet routing: Autonomous Systems (AS), RIP, OSPF, BGP. Address Resolution Protocol (ARP) and RARP.	10 hours
	Internet Security protocols Basic cryptography concepts, Secure Socket Layer (SSL), Internet Security Protocol (IPSec), Virtual Private Network (VPN).	6 hours
Pedagogy:	lectures/ tutorials/assignments/self-study	

References/ Readings	1. Forouzan, Behrouz A., and Firouz Mosharraf. "Computer networks: a top-down approach". McGraw-Hill, 2012. 2. Andrew S. Tanenbaum., "Computer Networks", (5th Edition) Prentice Hall of India. 3. James F. Kurose, Keith W. Ross, "Computer Networking: A Top-Down Approach" Pearson, Sixth Edition 2017.
Learning Outcomes	After completion of this course, students will be able to • Have a good understanding of layered communication architecture (TCP/IP) and knowledge of some of the important networking protocols • Understand the concepts of reliable data transfer and how TCP implements these concepts. • Basic knowledge of the routing algorithms.

Programme: MCA
Course Code: CSC-105 **Title of Course:** Data Structures & Algorithms Lab
Number of Credits: 3 (1L-0T-2P) **Contact Hours:** 60 hours (12L-0T-48P)
Effective from AY: 2021-22

Prerequisites for the course	Programme Prerequisites	
Objectives	To develop skills to design and implement linear and nonlinear data structures and to identify the most appropriate data structure for solving a real world problem.	
Content	**Lab Assignments may be based on the following** Advanced Linear Data Structures Infix-to-Postfix conversion, Evaluating Postfix Expressions, Bracket Matching Non-linear data structures Binary Trees Tree Traversal Algorithms Binary Search Trees Heap Priority Queue using Heap Heap Sort Graph implementation using Adjacency list and matrix Graph Traversal Algorithms Divide & Conquer Strategy MergeSort QuickSort Binary Search Algorithm Greedy Algorithms Huffman Coding Algorithm Prims' and Kruskal's Algorithm Dijkstra's Algorithm Dynamic Programming Coin Change Problem Longest Common Subsequence Floyd-Warshall Algorithm A Mini Project	2L + 8P 4L+16P 2L+8P 2L+8P 2L+8P
Pedagogy	Programming assignments/ discussions/ self-review/ peer-review/ testing of code/ debugging of code/ projects	
References/ Readings	1. Horowitz, Ellis, Sartaj Sahni, and Susan Anderson-Freed. "Fundamentals of data structures in C" WH Freeman & Co., Latest edition.	

	2. Thomas H. Cormen, Charles E. Leiserson, et al "Introduction to Algorithms", Latest Edition 3. Allen, Weiss Mark. "Data structures and algorithm analysis in C." Pearson Education India, Latest Edition. 4. Dasgupta, Papadimitriou, and Vazirani, "Algorithms" McGraw-Hill. 2017	
Learning Outcomes	Upon successful completion of the course, a student will be able to • Implement common data structures such as lists, stacks, queues, graphs, and binary trees for solving programming problems. • Identify and use appropriate data structures in the context of solution to a given problem.	

Programme: MCA
Course code: CSC-106 **Title of course:** Object Oriented Programming Lab
Number of credits: 3 (1L-0T-4P) **Total contact hours:** 60 hours (12L-0T-48P)
Effective from AY: 2021-22

Prerequisites for the course	Program Prerequisites	
Objectives	To impart programming skills using object oriented paradigms.	
Content	**Lab assignments using Java/C++/C#** Classes and objects Class, object, attributes & methods; classes as modules & types; uniform type system, wrapper type classes Object; object references; objects instantiation & interaction; constructor & destructor; pass-by-reference & pass-by-value Object copying & cloning; composite objects Static & non-static members Enumeration & Annotations Object oriented principles Encapsulation Inheritance; types of inheritance; diamond problem Abstraction; virtual methods Polymorphism; overloading and overriding Object oriented features Interfaces Access modifiers Errors & Exceptions; user-defined exceptions Collections Anonymous & Inner classes Type parametric polymorphism Advanced features Persistence & Serialization; JSON User packages & custom libraries; reflection Predicates & streams Lambda functions Mini-Project	3L+12P 3L+12P 3L+12P 3L+12P
Pedagogy	Hands-on assignments / pair programming / group project/ git project management.	
References/ Readings	**Main Reading** 1. Timothy Budd, “An Introduction to Object Oriented Programming”, Pearson Education, Latest Edition. 2. Brett D. McLaughlin, Gary Pollice & David West,	

	"Head First Object-Oriented Analysis Design", O'Reilly, Latest Edition. 3. Ken Arnold, James Gosling, David Holmes, "The Java Programming Language", Addison-Wesley Professional, Latest Edition 4. Stanley Lippman, "C++ Primer", Addison Wesley, 2012 5. Cay S. Horstmann, "Core Java Volume I—Fundamentals", Pearson, 2018 6. Herbert Schildt, "Java: The Complete Reference", Oracle Press, latest edition 7. Joshua Bloch, "Effective Java", Addison Wesley 8. Kathy Sierra & Bert Bates, "Head First Java", O'Reilly, 2012 9. Bjarne Stroustroup, "The C++ Programming Language", Addison Wesley, Latest Edition 10. https://www.tutorialspoint.com/java/index.htm	
Learning Outcomes	1. Learner will be able to write good object oriented code 2. Learner will understand object-oriented principles 3. Learner will be able to design object oriented softwares	

Programme: M.C.A

Course Code: CSC-106 **Title of the Course:** LINUX Lab

Number of Credits: 4 (2L-0T-2P) **Contact hours:** 72 hours (24L-0T-48P)

Effective from AY: 2021-22

Prerequisites for the course:	Program Prerequisites	
Objectives:	The objective is to introduce students to the Linux operating system environment and provide a knowledge of basic Linux commands and shell scripting and system call API.	
Content:	**LINUX Environment** Linux Installation and disk partitioning. Shell, Linux commands, Internal and External Commands, using the documentation/manual, users in Linux: user id, effective user id, use of commands su, sudo, id Basic commands: echo, who, whoami, date, cal, ls, passwd, history, shutdown. Input and output redirection operators (<,<<, >, >>)	3L + 3P
	The Linux File System, File and Directory management Structure of LINUX file system. Parent-child relationship. Concept of Home directory, current working directory and referring to home directory. Special Files: . and .. Absolute and relative pathnames. Use of PATH variable, Use of command: mkdir, rmdir, pwd, ls and cd. Use of file management commands: nano, touch, cat, cp, mv and rm. FIND command: Searching for a file using find, Finding List of files and directories. Concept of hard disk partitions, file system, Superblock and Inodes, General structure of Linux inode. use of stat command. Analysing the output of ls -l command. File type and permission. Use of chmod command. File ownership: Changing ownership using chown and chgrp commands. Modification and access times. Default file and directory permissions. Use of umask command. Concept of symbolic links. Hard and soft links. Use of ln command to create hard and soft links. Use of commands du, df, tar, zip, gzip, type, which	4L + 8P

	Filters: File commands- sort, wc, uniq, comm, cmp, diff, pg, tail, head, less, and more , Cut and Paste command Shells' sequence of interpretation of a command; Connecting commands with pipes Regular expressions: grep & sed command AWK script: Selection criteria and action- The BEGIN and END sections, Splitting a line into fields and using printf. Getline function and reading input from files. Writing output to file and pipes. Awk system variables. Using regular expressions. Relational and Boolean operations. Command line parameters and environment variables. Programming constructs: if, for, while.	6L + 8P
	Process Management Concept of UNIX process. Role of init in process creation. Process ID and exit status of a process. Displaying process attributes using ps command, Killing processes, foreground and background processes. Use of commands job, fg, bg	1L + 2P
	Package management: Installing & removing packages	1L+1P
	Shell Script Shell scripts and execution methods. The dot command, Interactive and Non Interactive execution. Use of export command, Aliases and command history. Shell variables, Special variables, Built-in shell parameters. Command line arguments. Escaping and quoting. Difference between single and double quotes. Command substitution, brace and tilde expansion, I/O using read and echo. Escape sequences, 'test' command, arithmetic expressions, operators, Control flow: For, If, While, Case. Shell functions, error handling, debugging.	4L + 8P
	System programming Introduction to system programming, System calls and library functions. **Files and Directory system calls** List of sample programs *1.* Write a program to implement the functionality of Linux command *touch* *2.* Write a program to implement the functionality of Linux command *cat*	5L + 18P

	3. Write a program to implement the functionality of Linux command *ls* 4. Write a program to redirect the output of all the printf statements to a user file using dup system call. 5. Write a program to read the standard input from a user file using dup system call. 6. Write a program to implement the functionality of Linux command *chmod* 7. Write two programs : one called parent.c, the other called child.c. The parent program reads two integers from the keyboard and arithmetic operator (+, -, *, /). The read information is transmitted to a child process. After the child process finishes the operation, it transmits the result to the parent process. The parent process prints the result on the screen. 8. Write a c program namely "parent.c", which reads the processes along with their burst time (bt) and saves it in a file. Using fork, create a child process namely fcfs.c, which takes the filename containing process information as a parameter from the parent. The child process task is to calculate the average waiting time using the FCFS scheduling algorithm.	
Pedagogy:	Practical/ tutorials/assignments/self-study	
References/ Readings	1. Unix Concepts and Applications – Sumitaba Das, Tata MacGraw Hill. 2. Unix and Shell Programming – Graham Glass and King Ables Pearson Education 3. C and Unix Programming – Kerningham and Pike, Prentice Hall 4. UNIX man pages	
Learning Outcomes	Upon completion of this course, the student will be able to: 1. Run various LINUX commands 2. Write shell script on LINUX OS. 3. Use various advanced LINUX tools such as grep, SED and AWK	

Programme: M.C.A

Course Code: CSC-108 **Title of the Course:** Communication Skills

Number of Credits: 2 (2L-0T-0P) **Contact Hours:** 24 hours (24L-0T-0P)

Effective from AY: 2021-22

Prerequisites for the course:	Programme requisites	
Objectives:	To introduce essentials of effective communication in different contexts	
Content:	**Oral Communication** Difference between formal and informal communication, importance of non verbal communication, skills required for effective communication, Public Speaking Skills.	4 hours
	Written Communication Writing cover letters, Resumés/CVs/Biodata, Letters of Invitation, Report Writing	4 hours
	Content creation Creating content for the website, creating profiles, creating content for brochures of events.	4 hours
	Multimedia and E-Correspondence Conducting Research before presentation, Making PowerPoint Presentation effective (visual), Communication during PowerPoint Presentation, Email etiquette (components, formats, attachments, content and language), Maintaining social media presence.	6 hours
	Preparing for Interview Types (personal, telephonic, online), Techniques of answering interviews, Participating in group discussions.	4 hours
	Allied Communication Effective Reading techniques, analyzing feedback and giving inputs.	2 hours
Pedagogy:	Lectures/ tutorials/laboratory work/ field work/outreach activities/ project work/ vocational training/viva/ seminars/ term papers/assignments/presentations/ self-study/ Case Studies etc. or a combination of some of these. Sessions shall be interactive in nature to enable peer group learning.	
References/ Readings	1. Kelly M. Quintanilla and Shawn T. Wahl, "Business and Professional Communication " Sage Publications, 2018,	

	2. Anjanee Sethi ,Bhavna Adhikari, "Effective Business Communication " Tata MacGraw Hill Education, India. 2009; 3. Nido Qubein, "How to be a Great Communicator in Person, On Paper, and on Podium" Viva Books, India. 2008; 4. Stanton, Nicky. "Mastering Communication", (5th Edition), Macmillan, 2009. 5. Dalmar, Fisher. "Communication in Organisation", West Pub, 1993. 6. Kilian, Crawford. "Writing for the Web. Self-Counsel Press", Fifth edition, 2015. 7. Kallos, Judith. "Email Etiquette Made Easy", Lulu.com. 2007.
Learning Outcomes	The participant will be able to facilitate interpersonal Communication, participate in group discussions, and to write effectively.

Programme: MCA
Course code: CSC-201 **Title of course:** Web Development
Number of credits: 3 (3L-0T-0P) **Total contact hours:** 36 hours (36L-0T-0P)
Effective from AY: 2021-22

Prerequisites for the course	Knowledge of HTML and basic of CSS(Program Prerequisites); Internet Technologies(CSC-104); Object Oriented Programming(CSC-102;CSC-106)	
Objectives	This course will introduce the learner to the different website development technologies	
Content	**Introduction** ● Evolution of internet & World Wide Web ● Client-Server Architecture ● Revisit HTML & CSS	1 hours
	Enhancing HTML & CSS ● HTML 5 ● CSS3	2 hours
	Front-end Design ● Good Design Rubrics ● Separation of concerns for HTML & CSS; structure vs visual representation ● HTML DOM ● CSS Box Model, pseudo -classes & -elements, CSS animation ● Adaptive & responsive design, viewport & media queries, mobile-first design ● Introduction to a design library and/or & framework (e.g. Bootstrap)	4 hours
	Client-side Scripting ● Dynamic web pages ● JavaScript, programming features, javascript events & functions ● Manipulating DOM ● Beyond ECMA 4 ● Introduction to a Javascript library and framework (e.g. JQuery, ReactJS)	10 hours
	HTTP & Middle-ware ● HTTP, Request & Response, methods & error code, headers, URL encoding & decoding ● XML, data & XPath ● JSON	3 hours
	Server-side Programming ● Server instance	9 hours

	• Request handling & response creation • HTML forms & file uploads • Session management & application data • Database connectivity • Introduction to a Server-side library and/or template engine and/or framework (e.g. PHP - Laravel; JSP - Spring)	
	Advanced Web Development • Model-View-Controller (MVC) & Model-View-ViewModel and others • Web service architecture and micro-services • REST calls, Asynchronous JavaScript and XML (AJAX) • Independent client-server web development • Difference between Server-side vs client-side rendering • Introduction to Web stacks, JAM stack & full stack development	7 hours
Pedagogy	Hands-on assignments / tutorials / peer-teaching / flip classroom/ presentations	
References/ Readings	1. Robert W. Sebesta, ‒Programming the World Wide Web‖, Pearson Education 2. https://www.w3schools.com/ 3. Steven Holzner, ‒HTML 5 Black Book‖ 4. https://www.tutorialspoint.com/ 5. Frank W. Zammetti, ―Modern Full-Stack Development‖, Apress 6. Nader Dabit, ‒Full Stack Serverless‖, O'Reilly	
Learning Outcomes	1. Learner will be able to make decision on what web technology to use and for what purpose 2. Learner will have fair idea on the popular technologies used in website development 3. Learner will appreciate the architecture of web applications and the design decisions	

Programme: MCA
Course Code: CSC-202 **Title of Course:** Database Management Systems
Number of Credits: 3 (3L-0T-0P) **Contact hours:** 36 hours (36L-0T-0P)
Effective from AY: 2020-21

Prerequisites for the course	A High Level Programming Language(Program Prerequisites);Data Structures and Algorithms(CS101); Operating Systems(CS103)	
Objectives	This course will enable the learner to understand the different issues involved in the design and implementation of a database system and provide both theoretical knowledge and practical skills required in the creation and use of a Relational DataBase Management System.	
Content	**Basic concepts** Database & Database Users, Characteristics of the Database Approach, Database Systems,Concepts & Architecture Data Models(RDBMS, Legacy systems, Object Oriented, NOSQL), Schemes & Instances DBMS Architecture of Data Independence, Database languages & Interfaces	4L
	Data Modelling using the Entity – Relationship approach	4L
	Relational Model, Languages & Systems Relational Data Model & Relational Algebra Relational Model Concepts Relational Model Constraints, Relational Algebra/Relational Calculus	5L
	SQL-A Relational Database Language Data SQL - DDL,DML. Views & Queries in SQL. Specifying Constraints & Indexes in SQL. Nested Subqueries, correlated Subqueries	6L
	Advanced SQL Embedded SQL, Dynamic SQL, Triggers and Stored Procedures.	3L
	Relational Data Base Design Function Dependencies & Normalization for Relational Database Functional Dependencies Normal forms based on primary keys (1NF, 2NF, 3NF, BCNF) Covers of Functional Dependencies, Canonical covers. Loss less join and Dependency preserving decomposition algorithms.	5L
	Transactions and Recovery Techniques Concept of a transaction, Recovery concepts, Recovery Techniques.	4L
	Concurrency Control	5L

	Serializability, Locking Techniques, Time stamp ordering Granularity of Data items	
Pedagogy	Hands-on assignments / tutorials / peer-teaching / troubleshooting	
References/ Readings	**Main Reading** 1. Korth, Silberchartz, ― Database System Concepts‖ McGrawhill Publication. 2. Elmasri and Navathe, ― Fundamentals of Database Systems‖, Addison Wesley, New Delhi. 3. Database Management Systems –R. Ramakrishnan, J.Gehrke – T.McGraw Hill 4. Desai B., ― An Introduction to Database Concepts‖, Galgotia Publications, New Delhi. 5. 2. Rob,Coronel, ―Database Systems (Design, Implementation and Management)‖ 6. Date C. J. , ― An Introduction to Database Systems‖, Publication House, New Delhi.	
Learning Outcomes	1. Understand and evaluate the role of a DBMS in information Technology applications in Organizations. 2. Recognise and use logical design methods and tools required in the design of DB applications. 3. Understand the relational database design principles. 4. Implement a database Solution to an IT Platform. 5. Understand the basics of SQL and construct queries using SQL. 6. Develop sophisticated queries to extract information from databases. 7. Use embedded SQL queries in a Host Level Language. Understand how the DBMS manages and recovers from concurrent and multiple transactions.	

Programme: MCA
Course code: CSC-203 **Title of course:** Mathematics for Computer Science
Number of credits: 4 (4L-0T-0P) **Total contact hours:** 48 hours (48L-0T-0P)
Effective from AY: 2021-22

Prerequisites for the course	Program Prerequisites	
Objectives	Students will be able to: Apply the concepts of mathematics in the modeling and design of computational problems and deeper understanding of subjects like machine learning/deep learning and other computer science subjects.	
Content	Introduction – importance of mathematics and their applications for computer science/machine learning/data science/deep learning *Functions, variables, equations, graphs* revision **Probability and Statistics:** Probability Rules & Axioms, Bayes' Theorem, Random Variables, Variance and Expectation, Conditional and Joint Distributions, Standard Distributions (Bernoulli, Binomial, Multinomial, Uniform and Gaussian), Moment Generating Functions, Maximum Likelihood Estimation (MLE), Prior and Posterior, Maximum a Posteriori Estimation (MAP) and Sampling Methods-confidence intervals, Hypothesis testing, p-values, A/B testing-ANOVA, t-test-Linear regression, regularization **Calculus** Overview of Differential and Integral Calculus, Partial Derivatives Product and chain rule-Taylor's series, infinite series summation/integration concepts-Fundamental and mean value-theorems of integral calculus, evaluation of definite and improper integrals-Beta and Gamma functions, Functions of multiple variables, limit, continuity, partial derivatives-Basics of ordinary and partial differential equations -Applications of Calculus	2 hrs 7 hrs 7L

	Linear Algebra: Systems of Linear Equations-Matrices-Solving Systems of Linear Equations-Vector Spaces-Linear Independence-Basis and Rank-Linear Mappings Affine Spaces **Analytic Geometry** Norms-Inner Products-Lengths and Distances Angles and Orthogonality-Orthonormal Basis Orthogonal Complement-Inner Product of Functions-Orthogonal Projections-Rotations **Matrix Decompositions** Determinant and Trace-Eigenvalues and Eigenvectors-Cholesky Decomposition Eigendecomposition and Diagonalization Singular Value Decomposition-Matrix Approximation. **Vector Calculus** Differentiation of Univariate Functions-Partial Differentiation and Gradients-Gradients of Vector-Valued Functions-Gradients of Matrices Useful Identities for ComputingGradients-Backpropagation and Automatic Differentiation-Higher-Order Derivatives-Linearization and Multivariate Taylor Series **Optimization** Primal Solutions and Concept and Need for Duality; Optimization Using Gradient Descent-Constrained Optimization -Lagrange Multipliers-Convex Optimization,	7L 7L 6L 7L 5L
Pedagogy	Problem solving approach and carrying out small project work using matlab tools	
References/ Readings	1. Statistics -Robert S. Witte and John S. Witte 2. Barron's AP Statistics, 8th Edition -Martin Sternstein, PhD. 3. Statistics for Business and Economics - James T. McClave, P. George Benson and Terry T Sincich 4. Naked Statistics: Stripping the Dread from	

	the Data – Charles Wheelan 5. Introduction to Linear Algebra - Gilbert Strang 6. Linear Algebra and Its Applications - David C. Lay 7. Functions and Graphs - I M Gelfand 8. Cartoon guide to calculus – Larry Gonick 9. Optimization Methods in Business Analytics—edX, MIT	
Learning Outcomes	1. To build a strong foundation in maths required for learning computer science/data science subjects. 2.To understand fundamental concepts and tools in calculus and linear algebra with emphasis on their applications to computer science in particular to data science/machine learning	

Programme: MCA
Course Code: CSC-204 **Title of Course:** Software Design & Engineering LAB
Number of Credits: 4 (2L-0T-2P) **Contact hours:** 72 hours (24L-0T-48P)
Effective from AY: 2021-22

Prerequisites for the course	Hands-on experience in object oriented programming (Program Prerequisites); Object Oriented Concepts(CSC-106)	
Objectives	This course will enable the learner to work in the software development ecosystem with tools and in team with a stress on how to adopt them in various activities and techniques	
Content	**Introduction to Software Development process** ● Software development processes and methodologies:Waterfall, agile methodologies.	3L
	Version control, build and continuous integration ● Baseline, identification, accounting, control, audit, source and version control (Git and Github; subversion), change control procedure, pull requests, fork & branches. ● Tools used in SCM	4L+4P
	Project Management using Scrum/ Lean approach ● Project planning and monitoring. Team management. Retrospectives. ● User story, estimation using user story, sprint planning, burndown chart.	6L+4P
	TDD and Refactoring ● TDD, Refactoring exercises(Eclipse Refactoring, Junit), BDD (ex. JBehave/Cucumber). ● Debugging: principles, approaches, use of debuggers.	2L+6P
	Testing ● Integration testing, GUI testing etc ● Automatic testing(desktop/web-selenium) ● Alpha and beta testing ● Continuous Integration tool (Travis) ● Defect tracking, bugzilla ● Static code analysis tools (Gprof, Sonar) ● Build management & Dependency tools – (Maven/Ant build)	3L+16P
	Design patterns: ● Reusability at design level. Principles of good design. Creational, structural and behavioral patterns. ● Refactoring to patterns-(Decorators,Observer and Factory Pattern).	6L+14P

	Documentation: ● Java Doc	1P
	Project ● Mini project implementation using Scrum tools	3P
Pedagogy	Hands-on assignments / tutorials / peer-teaching / troubleshooting	
References/ Readings	1. Martin Fowler, –Refactoring‖, Pearson Education. 2nd Edition, 2019 2. Erich Gamma, Richard Helm, Ralph Johnson,, John Vlissides,‖ Design Patterns: Elements of Reusable Object-oriented Software‖, Pearson Education. 3. Joshua Kareivesky, –Refactoring to Patterns‖, Pearson Education 4. Steve McConnell, –Code Complete‖, 2nd Edition. Redmond, Wa.: Microsoft Press, 2004 5. Chris Sims andHillary Louise Johnson –The Elements of Scrum‖, 2011 6. Rachel Davies, Liz Sedley –Agile Coaching‖, Pragmatic Bookshelf, 2009 7. Venkat Subramaniam, Andrew Hunt ―Practices of an Agile Developer‖ Pragmatic Bookshelf, 2006	
Learning Outcomes	Students will be able to use tools for development, testing and project management.	

Programme: MCA
Course code: CSC-205 **Title of course:** Web Development Lab
Number of credits: 3 (1L-0T-2P) **Total contact hours:** 60 hours (12L-0T-48P)
Effective from AY: 2021-22

Prerequisites for the course	Knowledge of HTML and basic of CSS(Program Prerequisites); Internet Technologies(CSC-104); Object Oriented Programming(CSC-102;CSC-106)	
Objectives	This course will focus on the practical use and aspects of the different website development technologies	
Content	**Web Design Assignments** *Suggested Sample (non-exhaustive) Assignments*:- ● Create a website on a topic given by the instructor. Evaluating the website with rubrics for good web design. ● Build a website using HTML & CSS by looking at a screenshot/picture of a website component given by the instructor. ● Websites built with tables, forms, images, iframes, etc. ● A website for each of design strategies (fixed, adaptive, responsive, fluid, mobile-first, etc.). ● Assignments using css pseudo-classes & -elements; grid & flex design; understanding the CSS box model & working with the browser developer tools; CSS transformations, transitions & animations ● Assignment to create a website built with Bootstrap based on a topic given by the instructor.	2L +12P
	Client-side Scripting Assignments *Suggested Sample (non-exhaustive) Assignments*:- ● An assignment for understanding the programming aspects of JavaScript and working with the browser developer tools. The use of the newer features of JavaScript (after ECMA 4) is encouraged. ● An assignment working with regular expressions. A search and filter utility can be built. ● Assignments for form data processing and validation and use of HTML5 form elements. A web page with form and validated data could be put in a table. The code could be written using table DOM methods and/or HTML DOM methods and/or XML DOM methods. ● Assignments using various events (mouse, keyboard, etc. events for the form elements, drag-and-drop, window, browser, etc.). ● A web component built using HTML, CSS & JavaScript based on a existing Bootstrap component	2L + 20P

	(e.g. Accordion) ● Assignment with the use of a JavaScript library (JQuery, AngularJS, ReactJS, etc.)	
	Developing a Game with HTML, CSS & JavaScript. The game should have at least 500 lines of (HTML+Javascript) code and make use of various mouse/keyboard events.	2L
	Server-side Programming Assignments *Suggested Sample (non-exhaustive) Assignments*:- ● Assignments to work with HTTP headers for passing data and meta-data, cookies, localStorage ● Assignments to handle data from web forms; handling the request and response payload ● Assignment to manage web sessions ● Assignment to develop a CRUD functionality by connecting to a database; AJAX calls	2L + 12P
	Full stack Web Developments *Develop a CRUD application with MEAN/MERN stack*	2L + 4P
	Mini-project *Ideally done in a group. It should include design and implementation of a web application. Project implementation should mandatorily be built using a templating engine or programming framework (client-side and/or server-side). Project should also use a design framework (e.g. Bootstrap). Conduct and progress of the project could follow industry practices (e.g. git, scrum etc.).*	2L
Pedagogy	Hands-on assignments / tutorials / peer-teaching / projects	
References/ Readings	1. Robert W. Sebesta, ‒Programming the World Wide Web‖, Pearson Education 2. https://www.w3schools.com/ 3. Steven Holzner, ‒HTML 5 Black Book‖ 4. https://www.tutorialspoint.com/ 5. Frank W. Zammetti, ‒Modern Full-Stack Development‖, Apress 6. Nader Dabit, ‒Full Stack Serverless‖, O'Reilly	
Learning Outcomes	1. Learner will be gain experience and be able to create complete websites 2. Learner will be able to make decision on what web technology to use and for what purpose 3. Learner will appreciate the architecture of web applications and the design decisions	

Programme: MCA
Course Code: CSC-202 **Title of Course:** Database Management Systems LAB
Number of Credits: 3 (1L-0T-2P) **Contact hours:** 60 hours (12L-0T-48P)
Effective from AY: 2020-21

Prerequisites for the course	A High Level Programming Language(Program Prerequisites); Hands-on experience in object-oriented programming(CSC-106).	
Objectives	This course aims at enabling learners to develop a skill set to design and implement a realistic application, representative of a typical real-life software system.	
Content	**Installation of DBMS Softwares**	2P
	Data Definition Language(DDL) Statements ● Creating a Database. ● Creating a table, with or without constraints. ● Understanding Data types. ● Altering the structure of the table like adding attributes at a later stage, modifying size of attributes or adding constraints to attributes. ● Removing the table created, i.e Drop table in SQL. ● Creating Sequence (Auto increment field)	1L+4P
	Query in Data Dictionary ● To view the structure of the table created by the user. ● To view user information. ● To view integrity constraints. ● Altering Session Parameters	1L+4P
	Data Manipulation Language(DML) Statements ● Inserting Data into the table. ● Updating Data into the table. ● Deleting Data from the table.	1L+6P
	Simple SQL statements ● Displaying all the attributes and tuples from the table. ● Displaying selected attributes/tuples from the table. ● Using Logical and comparison operators. ● String manipulation ● Date Comparisons	2L+6P
	Complex SQL Statements ● Using aggregate functions (using Group by and having clauses). ● Sorting Data. ● Creating SQL Aliases and Views. ● Joins and Nested queries.	2L+12P

	• Correlated subquery • Derived tables • Given a complex table structure, display records from tables.	
	Transaction Control Language(TCL) statements • Transactions could be made permanent in memory • To rollback the transaction.	1L+2P
	Embedded SQL statements • Loops/ if else statements • Creating Triggers/Procedures/packages • ArrayList and Cursor. • PL/SQL Strings • PL/SQL Object Oriented • Exceptions	3L+6P
	No SQL	1L+2P
	Mini Project *Will be done in a group of max 3 members(ideally). It should include design and implementation of a real world scenario. Project implementation should mandatorily be built using a two tiered architecture.* 1. Obtaining Client Specifications and converting the same to a Conceptual Design. 2. The project will be implemented on a (at least) two tiered architecture on any RDBMS platform. 3. A Project Report will need to be submitted – ideally Documentation for A User and an Administrator seperately. The project report that they submit consists of (i) Feasibility study (ii) ER Diagrams (iii) Tables normalized in an appropriate normal form with integrity and domain constraints noted. (iv) User Interface Design -Form and Report design , including triggers that may need to be written (v) User Manual Peer reviews of ERDs are held in the class.	4P (in class)
Pedagogy	Hands-on assignments / tutorials / peer-teaching / troubleshooting	
References/ Readings	1. Korth, Silberchartz, ― Database System Concepts‖ McGrawhill Publication. 2. Elmasri and Navathe, ― Fundamentals of Database Systems‖, Addison Wesley, New Delhi. 3. Documentation of the DBMS Platform	

Learning Outcomes	1. Design and implement a database schema for a given problem-domain 2: Create and maintain tables using SQL 3: Populate and query a database 4. Use Transaction Control Language 5. Creating and Using User Defined Data Types 6. Writing Triggers & Stored Procedures 7. Prepare reports 8. Application development using PL/SQL & front end tools	

Programme: M .C.A
Course : CSC 301 **Title of the Course:** Machine Learning
Number of Credits: 3(3L+0P) Hours: 36L
Effective from AY: 2020-21

Prerequisites for the course:	Mathematics for Computer Science(CSC-203)	
Objectives:	This course provides students with an in-depth introduction to three main areas of Machine Learning: supervised and unsupervised and reinforcement learning.this course will cover some of the main models and algorithms for regression, classification, clustering and Markov decision processes	
Content:	1. Introduction :- well posed learning problem – designing a learning system-perspectives and issues in machine learning. 2. Concept learning – concept learning task –notation – inductive learning hypothesis-concept learning as search-version space and candidate elimination algorithm-decision tree –random forest. 3. Revision of linear regression - logistic regression-Support vector machine kernel- Model selection and feature selection-Ensemble methods: Bagging, boosting. Evaluating and debugging learning algorithms. 4. Continuous Latent Variables-Revision of Principal Component Analysis -Maximum variance formulation - Minimum-error formulation - Applications of PCA - PCA for high-dimensional data. 5. Neural Networks -Feed-forward Network Functions – perceptron -Weight-space symmetries -Network Training - Parameter optimization -Local quadratic approximation - Use of gradient information - Gradient descent optimization - Error Backpropagation - Evaluation of error-function derivatives - A simple example - Efficiency of backpropagation - The Jacobian matrix - The Hessian Matrix - Diagonal approximation - Outer product	2 hrs 3 hrs 5 hrs 5 hrs 7 hrs

	approximation - Inverse Hessian- Finite differences - Exact evaluation of the Hessian - Fast multiplication by the Hessian. 6. Deep learning – Deep Feedforward Networks - - Gradient-Based Learning - Hidden Units - Architecture Design -CNN and RNN (simple RNN and LSTM) , language models(Transformers,BERT, GPT3) 8. Unsupervised learning ;Clustering.K-means.EM.Mixture of Gaussians. 9. Sequential Data - Markov Models - Hidden Markov Models -Maximum likelihood for the HMM -The forward-backward algorithm - The sum-product algorithm for the HMM -Scaling factors - The Viterbi algorithm. 10. Reinforcement learning – introduction- learning task-Q learning-non deterministic rewards and actions-temporal difference learning.	5 hrs 3 hrs 3 hours 3 hrs
Pedagogy:	lectures/ tutorials/assignments/self-study	
References/Readings	Main Reading :- 1.Introduction to Statistical Learning, Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, Springer, 2013. 2. EthemAlpaydin, Introduction to Machine Learning, MIT Press. 3. Richard O. Duda, Peter E. Hart, David G. Stork Pattern Classification,. 4. Peter Flach , Machine Learning , Cambridge 5.Christopher M. Bishop,Pattern recognition and machine Learning, springer. 6.Deep Learning, Ian Good fellow, MIT press	

	7.Tom Michele, Machine Learning, McGraw-Hill.	
Learning Outcomes	1. Develop an appreciation for what is involved in learning from data. 2. Understand a wide variety of learning algorithms. 3. Understand how to apply a variety of learning algorithms to data. 4. Understand how to perform evaluation of learning algorithms and model selection. 5. Equips them with a general understanding of deep learning.	

Programme: MCA

Course code: CSC-302 **Title of course:** Modern Development Platforms

Number of credits: 3 (3L-0T-0P) **Total contact hours:** 36 hours (36L-0T-0P)

Effective from AY: 2021-22

Prerequisites for the course	Programming(Program Prerequisites), Knowledge of OS (CSC-103), Networks(CSC-104) and Web Development(CSC-201,CSC-204)	
Objectives	This course will focus on the modern development technologies, tools and platforms prevalent in the software development industry	
Content	**Overview** ● Ever-changing development terrain, Importance of development at scale. Emergence of Cloud Services, Devops	1 hour
	Development at scale ● Introduction to API Query ● Introduction to ELK stack	4 hours
	Cloud Computing ● Overview ● Cloud Models - IaaS, PaaS, SaaS, Public/Private/Hybrid Cloud ● Components - Virtualisation & VMs, File Storage, Server Instances, Content Delivery Network, etc. ● Setting up cloud ● Cloud Services ● Case study of any one cloud (e.g. *Amazon AWS/ Google Cloud/ MS Azure)*	16 hours
	DevOps ● Overview of DevOps: ○ Introduction to DevOps ○ DevOps Lifecycle ○ DevOps Delivery Pipeline ● Continuous Integration/ Continuous Delivery (CI/CD) ○ Introduction to CI/CD	15 hours

	○ Continuous Delivery v/s Continuous Deployment ○ Case study of any one CI/CD tool(CircleCI/Jenkins, etc). Case study should include architecture, pipeline and plugin management ● Configuration Management ○ Introduction to Configuration Management ○ Case study of any one Configuration Management(e.g. Ansible, Chef, etc). Case study should include Infrastructure as Code, Inventory Management, playbooks/cookbooks ● Containerization ○ Introduction to Containerization ○ Container Lifecycle ○ Case study of any one containerization tool (e.g. Docker, etc) which should include namespaces, commands,CLI, image creation, image registry ● Continuous Monitoring ○ Introduction to continuous monitoring ○ Types: Infrastructure Monitoring, Application Monitoring and Network Monitoring ○ Case study on one continuous monitoring tool(e.g. Nagios, Prometheus, etc)	
Pedagogy	Hands-on assignments / tutorials / peer-teaching / flip classroom.	
References/ Readings	1. Frank W. Zammetti, "Modern Full-Stack Development", Apress 2. Nader Dabit, "Full Stack Serverless", O'Reilly 3. Joakim Verona, "Practical DevOps" 4. https://www.elastic.co/guide/index.html 5. https://docs.aws.amazon.com/ 6. https://cloud.google.com/docs 7. https://docs.microsoft.com/en-us/azure/?product=featured 8. https://docs.docker.com	
Learning	1. Learner will learn about the latest tools and platforms used in the software industry	

Outcomes	2. Learner will have fair idea on the popular cloud services used 3. Learner will appreciate the different devops tools and why devops is important	

Programme: M.C.A
Course Code: CS 302 **Title of the Course:** Machine Learning lab
Number of Credits: 3 **(1L+ 2P)** **Total contact Hours:** 1L +2P(12L+48P)
Effective from AY: 21-22

Prerequisites for the course:	Programming language (Program Prerequisites) Mathematics for Computer Science(CSC -203)	
Objectives:	The objective is to learn to build the different machine learning models by doing a set of assignments and mini projects.	
Content:	Introduction to python libraries for machine learning - scikit learn, tensor flow, keras, pytorch,pandas, matplotlib, seaborn, numpy and other relevant libraries.	1L + 5P
	Four branches of machine learning-supervised, unsupervised,self-supervised, reinforcement, Evaluating machine learning models ,Data preprocessing,featue engineering and feature learning,overfitting and underfitting-NumericalProgrammingfundamentals-finding nearest neighbours via euclidean distance-splitting data sets into training and testing	1L + 8P
	Regression,cross validation and regularization-polynomial regression -model selection on a fixed validation set -Polynomial Regression - Model Selection with Cross-Validation-Polynomial Regression with L2 Regularization - Model Selection with Cross-Validation-Comparison of methods on the test set Evaluating Binary Classifiers and Implementing Logistic Regression-Binary Classifier for movies reviews-classifying newswires-predicting house prices -Computing the Loss for Logistic Regression without Numerical Issues	2L + 10P
	Neural Networks and Stochastic Gradient Descent-MLPs with L-BFGS: What model size is effective?-MLPs with SGD: What batch size and step size?-Producing your own figure comparing batch size and learning rate.	2L + 10P
	Trees and Random Forests for Bag of Words-Code Implementation of Decision Tree Regression-Decision Trees for Review Classification -Random Forests for Review Classification -Comparing Trees to Linear Models	2L + 5P

	for Review Classification. Implementation of CNN, RNN, LSTM,Implementation of Boltzmann machine and Transformers (BERT, GPT3) Generative deep learning (GAN) Project discussions -Classifying Images with Feature Transformations-Classifying Sentiment from Text Reviews-Recommendation Systems via Matrix Factorization-Text summarization - language Translation - Sentimental analysis- speech to text translatioXiv, Explore the keras ecosystem.	2L + 10 P 2L + 5P
Pedagogy:	Programming in lab and practical exercises	
References/Readings	1. hands on machine learning with scikit learn by Aurielien 2. deep learning with python by Francois 3. Text Analytics with Python: A Practitioner's Guide to Natural Language Processing by dipanjan sarkar. 4. keras: the python deep learning API 5. https://www.cs.tufts.edu/comp/135/2020f/assignments.html 4.Python library reference	
Learning Outcomes	1.To be able to collect data and preprocess them 2 To choose the suitable machine learning model 3. To study its performance 4. To be able to carry out mini project	

Programme: MCA

Course code: CSC-304 **Title of course:** Modern Development Platforms Lab

Number of credits: 3 (0L-0T-3P) **Total contact hours:** 72 hours (0L-0T-72P)

Effective from AY: 2021-22

Prerequisites for the course	Hands-on experience in programming(Program Prerequisites), web development(CSC-201;CSC-205); A source and version control tool (CSC-204); Knowledge of OS(CSC-103) ; Networks(CSC-104)	
Objectives	This course will focus on the practical use and aspects of modern development technologies, tools and platforms prevalent in the software development industry	
Content	**Development at scale** ● Introduction to ELK stack ○ Assignments should be based on use of ELK stack. ● Introduction to API Query ○ Assignments should be based on querying API using tools like Graph QL	8 hours
	Cloud Services ● Assignments should be based on *(using one of: Amazon AWS/ Google Cloud/ MS Azure)* ○ Storage service (e.g. AWS S3) ○ Database service (e.g. AWS RDS) ○ Virtual Server service (e.g. AWS EC2) ○ Server-less service (e.g. AWS Lambda) ○ CDN service (e.g. AWS CloudFront) ○ Authentication (e.g AWS Cognito) ○ Load Balancing services (e.g. AWS Elastic Load Balancing)	30 hours
	DevOps ● CI/CD ○ Assignments should be based on constructing a CI/CD Pipeline using Git, Maven, Jenkins/CircleCI ● Configuration Management ○ Assignments should be based on configuration Management using tools like Ansible, Chef etc. ● Containerization (e.g. Docker)	28 hours

	○ Assignments should be based on creating containers from pre-existing images using tools like Docker, creating own container images and pushing container images to Docker Hub. ● Continuous monitoring for Infrastructure, Application & Network ○ Assignments should be based on continuous monitoring for Infrastructure, Application & Network using tools like (e.g. Nagios, Prometheus)	
	Mini-Project *Ideally done in a group. Concepts and tools (or similar) learnt in the course will need to be implemented/incorporated.*	6 hours
Pedagogy	Hands-on assignments / tutorials / peer-teaching Assignments may combine topics from Cloud Services and DevOps	
References/ Readings	1. Joakim Verona, "Practical DevOps" 2. Gene Kim , Patrick Debois , et al., "The DevOPS Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations" 3. https://www.elastic.co/guide/index.html 4. https://docs.aws.amazon.com/ 5. https://cloud.google.com/docs 6. https://docs.microsoft.com/en-us/azure/?product=featured 7. https://docs.docker.com	
Learning Outcomes	1. Learner will get hands-on experience working with the ELK stack and API Query 2. Learner will be able to configure and use different cloud services 3. Learner will get hands-on experience working with the Devops tools and platforms	

Programme: MCA
Course Code: CSO -1 **Title of Course:** Introduction to Cryptography
Number of Credits: 4 (4L-0T-0P) **Total contact hours:** 48 hours (48L-0T-0P)
Effective from AY: 2021-22

Prerequisites for the course	Knowledge of computer networks and protocols(CSC-104)	
Objectives	To understand basics of Cryptography and Network Security.To learn about how to maintain the Confidentiality, Integrity and Authenticity of a data. To understand different protocols for network security to protect against the threats in the networks.	
Content	**Foundations of Cryptography and Security** ● Ciphers and Secret Messages, ● Security Attacks and Services. ● Classical encryption techniques	4 hours
	Mathematical Tools for Cryptography ● Substitutions and Permutations ● Modular Arithmetic ● Euclid‟s Algorithm ● Finite Fields ● Polynomial Arithmetic	4 hours
	Design Principal of Block Ciphers ● Theory of Block ciphers ● Feistel Cipher network Structures ● DES, triple DES and AES ● Modes of Operation (ECB, CBC, OFB, CFB) ● Strength of DES, AES	8 hours
	Pseudo Random Numbers and Stream Ciphers ● Pseudo random sequences ● Liner Congruential generators ● Cryptographic generators ● Design of stream Ciphers ● RC4	2 hours
	Public Key Cryptography ● Prime Numbers and testing for primality ● Factoring large numbers ● Discrete Logarithms	2 hours
	Asymmetric Algorithms ● RSA ● Diffie-Hellman ● ElGamal ● Introduction of Ecliptics curve cryptosystems ● Key Management	8 hours

	● Key exchange algorithms ● Public Key Cryptography Standards	
	Hashes and Message Digests ● Hashing functions and their properties ● Message Authentication code ● MD5 ● SHA-3 ● HMAC	6 hours
	Digital Signatures, Certificate and Standards ● Digital signature standards (DSS and DSA) ● Public Key Infrastructures ● Digital certificates ● Basics of PKCS standards	4 hours
	Authentication ● Kerberos ● X509 Authentication Service	4 hours
	Web Security protocols ● IP Security ● Transport Layer Security (TLS) ● Wireless Security	6 hours
Pedagogy	Hands-on assignments / tutorials / peer-teaching / troubleshooting	
References/ Readings	1. Stallings William, " Cryptography and Network Security: Principles and Practises", 5th edition, Prentice Hall 2. Kahate Atul, "Cryptography and Network Security" Tata McGraw-Hill 3. Menezes A. J., P.C. Van Oorschot and S.A. Vanstone, "Handbook of Applied Cryptography"	
Learning Outcomes	After successful completion of the course, the learners would be able to ● Provide security of the data over the network. ● Implement various networking protocols. ● Protect any network from the threats in the world.	

Programme: MCA
Course Code: CSO-2 **Title of Course:** Android App Development
Number of Credits: 4 (4L-0T-0P) **Total contact hours:** 48 hours (48L-0T-0P)
Effective from AY: 2021-22

Prerequisites for the course	Hands-on experience in object oriented programming(CSC-106) and web development basics(CSC-201,CSC-205) and Knowledge of OS(CSC-103)	
Objectives	On completion of this course, the learner should be able to successfully build, debug and deploy android apps.	
Content	**Android OS, Ecosystem & Basics** ● Mobile Platforms & OSs; Approaches to mobile development; Android OS; Android System Architecture; Android App Lifecycle; Play Store ● Intro; Create Your First Android App; Layouts, Views and Resources; Text and Scrolling Views; Resources to Help You Learn ● Activities and Intents; The Activity Lifecycle and Managing State; Starting Activities with Implicit Intents ● Debugging your apps; Testing your app; Support libraries, and Backwards Compatibility	15 hours
	User Interface ● Screen Sizes; User Interaction - User Input Controls, Menus; Screen Navigation; RecyclerView ● Delightful User Experience; Drawables, Themes and Styles; Material Design; Providing Resources for adaptive layouts ● Testing the User Interface	15 hours
	Background Tasks ● Background Tasks; AsyncTask and AsyncTaskLoader; Connecting to the Internet; Broadcast Receivers; Services ● Triggering, Scheduling, and Optimizing Background Tasks; Notifications; Alarm Manager; Transferring Data Efficiently	9 hours
	Data Saving, Retrieving, Loading ● Overview to storing data ● Shared Preferences; App Settings ● SQLite; Firebase ● Sharing Data: Content Resolvers and Content Providers ● Using Loaders to Load and Display Data ● Connecting with API service endpoints	9 hours
Pedagogy	Hands-on assignments / tutorials / peer-teaching /	

	troubleshooting	
References/ Readings	1. Bill Philips & Brian Hardy, “Android Programming: The Big Nerd Ranch Guide” 2. Dawn Griffiths & David Griffiths, “Head First Android Development” 3. Ian F. Darwin, “Android Cookbook” 4. https://developer.android.com 5. https://kotlinlang.org	
Learning Outcomes	1. Learner will understand the android ecosystem, android versions & compatibility across them. 2. Learner will be able to design user interfaces specifically to be run native android devices. 3. Learner will be able to evaluate which type of views & widgets are preferable for various use cases. 4. Learner will be able to build and design navigation flows in an app. 5. Learner will be able to connect the app to Android services or apps already available on the device. 6. Learner will be able to build apps that can store data locally or remotely.	

Programme: MCA
Course Code: CSO-3 **Title of Course:** Programming Paradigms
Number of Credits: 4 (4L-0T-0P) **Total contact hours:** 48 hours (48L-0T-0P)
Effective from AY: 2021-22

Prerequisites for the course	Hands-on experience in programming(Program Prerequisites; CSC-101)	
Objectives	To Learn and understand various programming paradigms.	
Content	**Introduction** ● Programming paradigm concept, motivation, types and classification of paradigms. ● Factors with respect to programming languages: Binding times and flexibility; Scoping; First class values; Abstraction; Typing; Storage Allocation & Dynamic Memory	4 hours
	Imperative Programming ● Variables and data types; Operators and expressions; Input/Output operations, Decision constructs; Looping constructs ● Procedural (*in Python/C*) -- blocks & scope; procedures (functions) ● Object Oriented (*in Java/C++*) -- classes & objects, object-oriented principles (encapsulation, abstraction, inheritance, polymorphism)	4 hours
	Functional Programming (*in Haskell/Clojure*) ● Revision of mathematical Functions‟ concepts ● Side effects; Pure functions ● Type induction ● Defining functions ● Currying; Function composition ● Recursion ● Lazy evaluation; infinite lists ● List comprehensions ● Higher order functions; Folds	16 hours
	Logic Programming (*in Prolog/ECLiPSe Constraint language*) ● Revision of mathematical Logic concepts ● Programming “without algorithms” ● Logic programming with facts, rules and goals ● Recursion; Lists ● Constraint logic programming; constraints as relationship between variables; solving puzzles (like sudoku)	8 hours
	Event-driven Programming (*in Python/.NET*) ● Events	8 hours

	• Main loop & callback • Scheduler & Event handlers; Triggers • Exception handling • Reliable eventing • Asynchronous triggers	
	Multi-Paradigms and more • Language support for multi paradigms; Benefits & issues • Parallel programming -- Data Parallelism (*in OpenMP*) and Message Passing (*in MPI*) • Reactive programming (*in Elm/ReactiveX for Java, JS*) • Meta programming (*in Lisp*) • Natural Language Programming (*in SciLab/MATLAB*)	8 hours
Pedagogy	Hands-on assignments / tutorials / peer-teaching / pair programming/ reading research papers/ presentations	
References/ Readings	• Terrance W. Pratt, Marvin V. Zelkowitz, "Programming Languages - Design & Implementation" • Robert L. Sebesta, "Concepts of Programming Languages" • Ravi Sethi, "Programming Languages Concepts & Constructs" • Bruce J. Mac Lennan, "Principles of Programming Languages: Design, Evaluation, and Implementation" • Kenneth C. Louden, "Programming Languages: Principles and Practice" • Allen Tucker, Robert Noonan, "Programming Languages: Principles and Paradigms" • Graham Hutton, "Programming in Haskell" • W. Clocksin, "Programming in Prolog" • Slim Abdennadher, Thom Frühwirth, "Essentials of Constraint Programming" • Roland Kuhn, Brian Hanafee, Jamie Allen, "Reactive Design Patterns"	
Learning Outcomes	1. Learner will be able to distinguish between different programming paradigms 2. Learner will be able to choose an adequate programming paradigm in solving specific software engineering problems 3. Learner will be able to recognize the similar concepts implemented in a different way across different programming languages and paradigms	

Programme: MCA
Course Code: CSO-4 **Title of Course:** Theory of computation
Number of Credits: 4 (4L-0T-0P) **Contact Hours:** 48 hours (48L-0T-0P)
Effective from AY: 2021-22

Prerequisites for the course	Program Prerequisites	
Objectives	1. To give an overview of the theoretical foundations of computer science from the perspective of formal languages 2. To illustrate finite state machines to solve problems in computing.	
Content	General Concepts of Automata Theory: Alphabets Strings, Languages, Grammars, Applications of Automata Theory.	2 hours
	Finite Automata (FA): Introduction, Deterministic Finite Automata (DFA) - definition and notations, language of a DFA. Nondeterministic Finite Automata (NFA)- Definition, language of an NFA, Equivalence of DFA and NFA, Applications of FA. Finite Automata with Epsilon Transitions, Eliminating Epsilon transitions, Minimization of DFA. Finite automata with output (Moore and Mealy machines) and inter-conversion.	10 hours
	Regular Expressions (RE): Introduction, Identities of RE. Finite Automata and Regular Expressions - conversions, Algebraic Laws for Regular Expressions, applications of RE. Regular grammars: Definition, regular grammars, and FA, Proving languages to be non-regular (Pumping lemma), Properties of Regular Language, applications.	8 hours
	Context-Free Grammar (CFG): Definition, Derivations Using a Grammar- Leftmost and rightmost derivation, Parse tree, Applications, Ambiguity in CFG. Minimization of CFG, CNF, GNF, Pumping Lemma for CFL‟s.	8 hours
	Pushdown Automata (PDA): Definition, Language of PDA- Acceptance by Final State and Acceptance by Empty stack, Equivalence of CFG and PDA, Deterministic PDA, Chmosky normal form of CFG Turing Machines (TM): Formal definition and behavior, Languages of a TM, TM as accepters, and TM as a computer	12 hours

	of integer functions, Types of TMs.	
	Recursive And Recursively Enumerable Languages (REL): Properties of recursive and recursively enumerable languages, Universal Turing machine, The Halting problem, Undecidable problems about TMs. Context-sensitive language and linear bounded automata (LBA), Chomsky hierarchy, Decidability.	8 hours
Pedagogy	lectures/ tutorials/assignments/self-study	
References/ Readings	1. John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, Introduction to Automata Theory Languages and Computation, Pearson Education, India (latest edition) 2. H.R.Lewis and C.H.Papadimitriou, Elements of the Theory of Computation, PHI, (latest edition) 3. J.Martin, Introduction to Languages and the Theory of Computation, TMH (latest edition)	
Learning Outcomes	At the end of the course students will be able to: ● To use basic concepts of formal languages of finite automata techniques ● To design Finite Automata for different Regular Expressions and Languages ● To Construct context-free grammar for various languages	

Programme: MCA
Course code: CSO-5 **Title of course:** Data Analytics
Number of credits: 4 (4L-0T-0P) **Total contact hours:** 48 hours (48L)
Effective from AY: 2021-22

Prerequisites for the course	Program Prerequisites; DBMS(CSC-202); Probability and statistics(CSC-203)	
Objectives	In this course, the learner will learn to ask the right questions to understand and analyze data by doing EDA and building statistical models. The learner will also learn to convert data into insights so as to be able to tell the story about data.	
Content		Hours
	1.Introduction:- what is data analytics and data science -areas of data sciences- data analytics intro	3L
	2. Revision of statistics:- Measures of central tendency-Measures of location of dispersions-Statistical hypothesis generation and testing-Chi-Square test-t-Test-Analysis of variance-Correlation analysis-Maximum likelihood test-Practice and analysis with R/python.	3L
	3. Data preparation-Business Intelligence tools-Datawarehouse-DataBase-setting up Microsoft SQL server for practice-ETL-BI tools-SQL programming for data science	7L
	4. Getting started with basics of python & R- library for data science	2L
	5. Visualization and Exploratory data analysis (EDA)-various plots using python , R and Tableau and sweetvize - histogram-kernel density plots- combining plot styles-box and violin plots-regression plots-heat maps and clustered matrices.	9L
	6. Data analysis techniques-Regression analysis-Classification techniques-Clustering-Association rules analysis-Practice and analysis with R/python.	9L

	7. Assessing the model:- Accuracy paradox- cumulative accuracy profile(CAP)-Drawing insights from the model- power insights from your CAP-coefficients of logistic regression-odd ration-odd ration vs coefficient-deriving insights from coefficients-Model maintenance . 8. Case studies and projects:- Understanding business scenarios-Feature engineering and visualization- Sensitivity Analysis- data analytics project in python or R in different domains like financial , agriculture, health, language etc.	8L 7L
Pedagogy	1. Problem solving approach with real life problems. 2. hands-on assignments 3. Learning theory and putting them into practice by doing projects in either python or R programming	
References/ Readings	1. Introduction to statistical learning by Trevor and Robert. 2. The Elements of Statistical Learning by Trevor Hastie. 3. Beginning R-the statistical programming language by Mark Gardener. 4. probability and statistics for Engineers and scientists by Ronald and Raymond. 5. Storytelling with data by Cole Nussbaumer Knaflic **6.** Python for data analysis by **Wes McKinney**	
Learning Outcomes	Having done this course equips any aspiring data analyst /data scientist to know the data –learn to read between lines and to see the hidden insights otherwise not visible to ordinary vision of the common man. The hidden insights collected would help make right decision for any business problem	

Programme: MCA
Course Code: CSO-6 **Title of Course:** Network Programming
Number of Credits: 4 (4L-0T-0P) **Contact Hours:** 48 hours (48L-0T-0P)
Effective from AY: 2021-22

Prerequisites for the course	Operating Systems(CSC-103), Internet technology(CSC-104),Linux (CSC-107)	
Objectives	To introduce the basic concept of network programming in UNIX and Windows OS environments.	
Content	Basic UNIX programming: Overview of process, signal handling, and related system calls. Named and unnamed pipes and related system calls.	6 hours
	Elementary Socket Programming: Berkley Sockets Overview, Introduction to sockets, Socket addresses, Basic Socket system calls, Error handling. Concept of Reserved ports, Elementary TCP and UDP socket programming. Socket options. Name and Address Conversion functions. Interface Operations using „ioctl“.	12 hours
	I/O Operations: Synchronous vs. Asynchronous I/O. I/O Multiplexing using „select“ and „pselect“., Sockets and signals, Signal driven I/O. Nonblocking I/O: Nonblocked „accept“ and „connect“. Broadcasting and Multicasting. Sending and Receiving Out of Band data using „select“ and signals. Advance I/O functions.	12 hours
	Daemon processes and Inetd Super Server	2 hours
	Network Programming in the .NET Framework: System.Net classes overview, working with URI, IP addresses, DNS class, Requests and responses, authentication, and permission.	4 hours
	Socket programming in .NET Working with sockets in .NET, Asynchronous programming, socket permission, support for IPv6, support for TCP, .NET Remoting, support for UDP, multicast sockets. Network tracing, network information, cache management, security.	6 hours
	Programming applications: Time and date routine, Ping, Trivial file transfer protocol, design of chat application using multicast socket programming.	6 hours

Pedagogy	lectures/ Hands-on assignment/tutorials	
References/ Readings	Main Reading: 1. Steven W.R., Unix Network Programming, Prentice Hall of India. 2. Microsoft Software Developers Network Documentation.	
Learning Outcomes	After completing the course, students will be able to: ● Analyze and write socket API based programs ● Design and implement client-server applications using TCP and UDP sockets	

Programme: MCA
Course Code: CSO-7 **Title of Course:** IoT architecture and protocols
Number of Credits: 4 (4L-0T-0P) **C ontact Hours:** 48 hours (48L-0T-0P)
Effective from AY: 2021-22

Prerequisites for the course	Program Prerequisites, Operating Systems(CSC-103),Internet Technology(CSC-104).	
Objectives	To understand the fundamentals of Internet of Things and the protocols and standards designed for IoT	
Content	Introduction to IoT: Introduction, IoT ecosystem, Applications, Challenges.	2 hours
	Fundamentals: IoT Devices - Sensors, Actuators, and gateways, Basics of the wireless sensor network.	4 hours
	IoT Architecture & Design: oneM2M, IoTWF, Additional Reference Models, Core functional stack, Data Management and compute stack.	6 hours
	Communicating smart objects: Communication criteria, communication models, IoT access technologies – 3GPP MTC, IEEE 802.11, IEEE 802.15, WirelessHART, ZWave, Bluetooth Low Energy, Zigbee Smart Energy, DASH7	10 hours
	IoT Network Layer: IP as IoT network layer, IPv6, 6LoWPAN, 6TiSCH, RPL, CORPL, CARP	8 hours
	IoT Transport and Application protocols: Transport Layer: TCP, UDP, DCCP, SCTP, TLS, DTLS IoT application transport methods, HTTP, CoAP, XMPP, MQTT, AMQP, DDS	12 hours
	Security in IoT: MAC802.15.4, 6LoWPAN, RPL, Application Layer security.	3 hours
	IoT Application case study: Discuss any 3 applications of IoT	3 hours
Pedagogy	lectures/ tutorials/Hands-on assignments/self-study	
References/ Readings	1. David Hanes, Gonzalo Salgueiro, Patrick Grossetete, Robert Barton, Jerome Henry, “IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things”, CISCO Press, 2017	

	2. Hersent, Olivier, David Boswarthick, and Omar Elloumi, The internet of things: Key applications and protocols. John Wiley & Sons, 2011. 3. Buyya, Rajkumar, and Amir Vahid Dastjerdi, eds. Internet of Things: Principles and Paradigms. Elsevier, 2016.	
Learning Outcomes	After completing the course, students will be able to: ● Understand the concepts of the IoT Architecture Reference model ● Identify the IoT networking components and protocols.	

Programme: MCA
Course Code: CSO-8 **Title of Course:** Quality Assurance and Usability
Number of Credits: 4 (4L-0T-0P) **Total contact hours:** 48 hours (48L-0T-0P)
Effective from AY: 2021-22

Prerequisites for the course	Object oriented programming(CSC-106), software engineering tools & processes(CSC-204),web development(CSC-205)	
Objectives	The course is aimed at providing learners with the necessary exposure to the job responsibilities of Software Quality Assurance (QA) engineers & User Experience (UX) designers in the IT industry.	
Content	Testing Web Applications ● Manual Testing ● Write test cases ● Automated test scripting (e.g. Selenium, QTP)	8 hours
	Developer-centric Testing ● API testing (e.g. Postman, Karate, SOAP-UI) ● Revisit to Unit Testing (e.g. NUnit, JUnit) ● Tests for Model, View & Controller (MVC)	8 hours
	Building test suites ● Design Patterns in Test Automation ○ Page-Object Model (Page-Object pattern) ○ Business-Layer Page-Object pattern ○ Using software development design patterns (creational, structural and behavioral) in test scripting ● Automated Testing Frameworks (e.g. Cucumber, Jasmine, Mocha, TestNG) ● Testing for Behavior Driven Development & Testing for Business Driven Development (e.g. Gherkin)	12 hours
	Non functional testing ● Performance Testing (e.g. Apache JMeter) ● Querying logs	6 hours
	Testing for Mobiles Apps	4 hours
	User Experience ● Gulf of evaluation and execution; 7 fundamental & universal design principles; Human error vs Bad design; Double-Diamond Model of Design ● Visual Design Elements (line, color, shape, form vs space, value, texture) and extended elements (dot, typography, movement)	10 hours

	• Visual Design Principles (scale, dominance/emphasis, balance, harmony) • Wireframing, Mockup & Prototype (Paper & Digital); Use of tools (e.g. Pencil, Adobe XD, Sketch and/or Figma); Interaction & Animation • Raster (e.g. GIMP, Adobe Photoshop) & Vector (e.g. Inkscape, Adobe Illustrator, CorelDraw) Graphic Editing • Maintaining your UX designs‟ portfolio (e.g. Behance, Dribble)	
Pedagogy	Hands-on assignments / tutorials / peer-teaching / mini-project / case studies/ presentations	
References/ Readings	1. Dorothy Graham, Rex Black, Erik van Veenendaal, “Foundations of Software Testing ISTQB Certification” 2. Don Norman, “The Design of Everyday Things” 3. Joseph A. Gatto, Albert W. Porter, Jack Selleck, “Exploring Visual Design: The Elements and Principles” 4. https://tutsplus.com 5. https://www.youtube.com/watch?v=Ib8UBwu3yGA 6. https://www.youtube.com/watch?v=IyR_uYsRdPs 7. https://www.youtube.com/watch?v=68w2VwalD5w	
Learning Outcomes	1. Learners will understand software testing and quality assurance as a fundamental component of software life cycle 2. Learners will efficiently perform quality assurance activities using modern software tools 3. Learners will prepare test plans and schedules for a quality assurance project 4. Learners will understand design workflows while building software products 5. Learner will efficiently create user experience (UX) designs and other deliverables using modern software tools	