UNIVERSITY OF MUMBAI

Syllabus for

Program: Bachelor of Science

Course: Computer Science

Choice Based Credit System (CBCS)
(Revised)

with effect from

Academic Year 2021-2022

Preamble

The rise of Information and Communication Technology (ICT) has profoundly affected modern society. Increasing applications of computers in almost all areas of human endeavor has led to vibrant industries with concurrent rapid change in technology.

As the computing field advances at a rapid pace, the students must possess a solid foundation that allows and encourages them to maintain relevant skills as the field evolves. Specific languages and technology platforms change over time. Thus students must continue to learn and adapt their skills throughout their careers. To develop this ability, students will be exposed to multiple programming languages, tools, paradigms and technologies as well as the fundamental underlying principles throughout this programme.

The programme offers required courses such as programming languages, data structures, computer architecture and organization, algorithms, database systems, operating systems, and software engineering; as well as specialized courses in artificial intelligence, computer-based communication networks, distributed computing, information security, graphics, human-computer interaction, multimedia, scientific computing, web technology, and other current topics in computer science.

The core philosophy of this programme is to –

- Form strong foundations of Computer Science
- Nurture programming, analytical & design skills for the real world problems.
- Introduce emerging trends to the students in gradual way.
- Groom the students for the challenges of ICT industry

The students these days not only aspire for a career in the industry but also look for research opportunities. The main aim of this programme is to deliver a modern curriculum that will equip graduates with strong theoretical and practical backgrounds to enable them to excel in the workplace and to be lifelong learners. Not only does it prepares the students for a career in Software industry, it also motivates them towards further studies and research opportunities. Graduating students, can thus take up postgraduate programmes in CS leading to research as well as R&D, can be employable at IT industries, or can adopt a business management career.

In the first year i.e. for semester I & II, basic foundation of important skills required for software development is laid. The syllabus proposes to have four core subjects of Computer science and two core courses of Mathematics-Statistics. All core subjects are proposed to have theory as well as practical tracks. While the Computer Science courses will form fundamental skills for solving computational problems, the Mathematics & Statistics course will inculcate research-oriented acumen. Ability Enhancement Courses on Soft Skill Development will ensure an overall and holistic development of the students. The syllabus design for further semesters encompasses more advanced and specialized courses of Computer Science.

We sincerely believe that any student taking this programme will get very strong foundation and exposure to basics, advanced and emerging trends of the subject. We hope that the students' community and teachers' fraternity will appreciate the treatment given to the courses in the syllabus.

We wholeheartedly thank all experts who shared their valuable feedbacks and suggestions in order to improvise the contents, we have sincerely attempted to incorporate each of them. We further thank Chairperson and members of Board of Studies for their confidence in us.

Special thanks to Department of Computer Science and colleagues from various colleges, who volunteered or have indirectly helped designing certain specialized courses and the syllabus as a whole.

Programme Structure for B.Sc. Computer Science

Programme Duration	06 Semesters *spread across 3 years*
Total Credits required for successful completion of the Course	120
Credits required from the Core Courses	76
Credits required for the Ability Enhancement Courses	04
Credits required for Skills Enhancement Courses	32
Credits for General Elective Courses	08
Minimum Attendance per Semester	75%

Progamme Objectives

The objectives of the 3 year B.Sc. Computer Science programme are as follows:

- To develop an understanding and knowledge of the basic theory of Computer Science with good foundation on theory, systems and applications.
- To fosternecessary skills and analytical abilities for developing computer based solutions of real-life problems.
- To provide training in emergent computing technologies which lead to innovative solutions for industry and academia.
- To develop the necessary study skills and knowledge to pursue further post-graduate study in computer science or other related fields.
- To develop the professional skillset required for a career in an information technology oriented business or industry.
- To enable students to work independently and collaboratively, communicate effectively, and become responsible, competent, confident, insightful, and creative users of computing technology

Progamme Learning Outcomes

At the end of three year Bachelor of Computer Science the students will be able:

- To formulate, to model, to design solutions, procedure and to use software tools to solve real world problems.
- To design and develop computer programs/computer -based systems in the areas such as networking, web design, security, cloud computing, IoT, data science and other emerging technologies.
- To familiarize with the modern-day trends in industry and research based settings and thereby innovate novel solutions to existing problems.
- To apply concepts, principles, and theories relating to computer science to new situations.
- To use current techniques, skills, and tools necessary for computing practice
- To apply standard Software Engineering practices and strategies in real-time software project development
- To pursue higher studies of specialization and to take up technical employment.
- To work independently or collaboratively as an effective tame member on a substantial software project.
- To communicate and present their work effectively and coherently.
- To display ethical code of conduct in usage of Internet and Cyber systems.
- To engage in independent and life-long learning in the background of rapid changing IT industry.

F.Y.B.Sc. Computer Science Syllabus

Choice Based Credit System (CBCS)

with effect from

Academic year 2021-2022

Semester – I				
Course Code	**Course Type**	**Course Title**	**Credits**	**Lectures/Week**
USCS101	Core Subject	Digital Systems & Architecture	2	3
USCSP101	Core Subject Practical	Digital Systems & Architecture – Practical	1	3
USCS102	Core Subject	Introduction to Programming with Python	2	3
USCSP102	Core Subject Practical	Introduction to Programming with Python – Practical	1	3
USCS103	Core Subject	LINUX Operating System	2	3
USCSP103	Core Subject Practical	LINUX Operating System – Practical	1	3
USCS104	Core Subject	Open Source Technologies	2	3
USCSP104	Core Subject Practical	Open Source Technologies – Practical	1	3
USCS105	Core Subject	Discrete Mathematics	2	3
USCSP105	Core Subject Practical	Discrete Mathematics – Practical	1	3
USCS106	Core Subject	Descriptive Statistics	2	3
USCSP106	Core Subject Practical	Descriptive Statistics – Practical	1	3
USCS107	Ability Enhancement Course	Soft Skills	2	3

F.Y.B.Sc. Computer Science Syllabus

Choice Based Credit System (CBCS)

with effect from

Academic year 2021-2022

Semester – II				
Course Code	**Course Type**	**Course Title**	**Credits**	**Lectures/Week**
USCS201	Core Subject	~~Design & Analysis of Algorithms~~	2	3
USCSP201	Core Subject Practical	Design & Analysis of Algorithms – Practical	1	3
USCS202	Core Subject	Advanced Python Programming	2	3
USCSP202	Core Subject Practical	Advanced Python Programming – Practical	1	3
USCS203	Core Subject	Introduction to OOPs using C++	2	3
USCSP203	Core Subject Practical	Introduction to OOPs using C++ – Practical	1	3
USCS204	Core Subject	Database Systems	2	3
USCSP204	Core Subject Practical	Database Systems – Practical	1	3
USCS205	Core Subject	Calculus	2	3
USCSP205	Core Subject Practical	Calculus – Practical	1	3
USCS206	Core Subject	Statistical Methods	2	3
USCSP206	Core Subject Practical	Statistical Methods – Practical	1	3
USCS207	Ability Enhancement Course	E-Commerce & Digital Marketing	2	3

Semester I

Course Code	Course Title	Credits	Lectures /Week
USCS101	**Digital Systems & Architecture**	**2**	**3**

About the Course:

This course introduces the principles of computer organization and the basic architecture concepts.The course emphasizes performance and cost analysis, instruction set design, pipelining, memory technology, memory hierarchy, virtual memory management, and I/O systems.

Course Objectives:

- To have an understanding of Digital systems and operation of a digital computer.
- To learn different architectures & organizations of memory systems, processor organization and control unit.
- To understand the working principles of multiprocessor and parallel organization's as advanced computer architectures

Learning Outcomes:

After successful completion of this course, students would be able to

- To learn about how computer systems work and underlying principles
- To understand the basics of digital electronics needed for computers
- To understand the basics of instruction set architecture for reduced and complex instruction sets
- To understand the basics of processor structure and operation
- To understand how data is transferred between the processor and I/O devices

Unit	Topics	No of Lectures
I	**Fundamentals of Digital Logic:** Boolean algebra, Logic Gates, Simplification of Logic Circuits: Algebraic Simplification, Karnaugh Maps. Combinational Circuits: Adders, Mux, De-Mux, Sequential Circuits: Flip-Flops (SR, JK & D), Counters: synchronous and asynchronous Counter **Computer System:** Comparison of Computer Organization & Architecture, Computer Components and Functions, Interconnection Structures. Bus Interconnections, Input / Output: I/O Module, Programmed I/O, Interrupt Driven I/O, Direct Memory Access	**15**
II	**Memory System Organization:** Classification and design parameters, Memory Hierarchy, Internal Memory: RAM, SRAM and DRAM, Interleaved and Associative Memory. Cache Memory: Design Principles, Memory mappings, Replacement Algorithms, Cache performance, Cache Coherence. Virtual Memory, External Memory: Magnetic Discs, Optical Memory, Flash Memories, RAID Levels **Processor Organization:** Instruction Formats, Instruction Sets, Addressing Modes, Addressing Modes Examples with Assembly Language [8085/8086 CPU], Processor Organization, Structure and Function. Register	**15**

	Organization, Basic Microprocessor operations: Data Transfer (Register / Memory) Operations, Arithmetic & Logical Operations, Instruction Cycle, Instruction Pipelining. Introduction to RISC and CISC Architecture, Instruction Level Parallelism and Superscalar Processors: Design Issues	
III	**Control Unit:** Micro-Operations, Functional Requirements, Processor Control, Hardwired Implementation, Micro-programmed Control**.** **Fundamentals of Advanced Computer Architecture:** Parallel Architecture: Classification of Parallel Systems, Flynn's Taxonomy, Array Processors, Clusters, and NUMA Computers. Multiprocessor Systems: Structure & Interconnection Networks, Multi-Core Computers: Introduction, Organization and Performance.	**15**

Textbooks:

1. M. Mano, Computer System Architecture 3rd edition, Pearson
2. Carl Hamacher et al., Computer Organization and Embedded Systems, 6 ed., McGraw-Hill 2012
3. R P Jain, Modern Digital Electronics, Tata McGraw Hill Education Pvt. Ltd. , 4th Edition, 2010

Additional References:

1. William Stallings (2010), Computer Organization and Architecture- designing for performance,8th edition, Prentice Hall, New Jersy.
2. Anrew S. Tanenbaum (2006), Structured Computer Organization, 5th edition, PearsonEducation Inc,
3. John P. Hayes (1998), Computer Architecture and Organization, 3rd edition, Tata McGrawHill

Course Code	**Course Title**	**Credits**	**Lectures /Week**
USCSP101	**Digital Systems & Architecture – Practical**	**1**	**3**
1	Study and verify the truth table of various logic gates (NOT, AND, OR, NAND, NOR, EX-OR, and EX-NOR).		
2	Simplify given Boolean expression and realize it.		
3	Design and verify a half/full adder		
4	Design and verify half/full subtractor		
5	Design a 4 bit magnitude comparator using combinational circuits.		
6	Design and verify the operation of flip-flops using logic gates.		
7	Verify the operation of a counter.		
8	Verify the operation of a 4 bit shift register		
9	Design and implement expression using multiplexers / demultiplexers.		
10	Design and implement 3-bit binary ripple counter using JK flip flops.		
11	Simple microprocessor programs for data transfer operations		
12	Simple microprocessor programs for arithmetic & logical transfer operations		
Note	Practical 1 – 10 can be performed using any open source simulator (like Logisim) (Download it from https://sourceforge.net/projects/circuit/)		

	Practical 11 – 12 can be performed on any simulation software like Jubin's 8085 simulator		
Course Code	**Course Title**	**Credits**	**Lectures /Week**
USCS102	**Introduction to Programming with Python**	**2**	**3**

About the Course:

This course is aims at introducing one of the fastest growing programming language of current time and enables learners to understand the fundamentals of programming with Python. Learners will be able to write programs to solve real-world problems, and produce quality code. It will help to develop strong skills of programming for implementing applications for emerging fields including data science and machine learning.

Course Objectives:

- To learn how to design and program Python applications.
- To explore the innards of Python Programming and understand components of Python Program
- To define the structure and components of a Python program.
- To learn how to write loops and decision statements in Python
- To learn about inbuilt input/output operations and compound data types in Python

Learning Outcomes:

After successful completion of this course, students would be able to:

- Ability to store, manipulate and access data in Python
- Ability to implement basic Input / Output operations in Python
- Ability to define the structure and components of a Python program.
- Ability to learn how to write loops and decision statements in Python.
- Ability to learn how to write functions and pass arguments in Python.
- Ability to create and use Compound data types in Python

Unit	Topics	No of Lectures
I	**Overview of Python:** History & Versions, Features of Python, Execution of a Python Program, Flavours of Python, Innards of Python, Python Interpreter, Memory Management in Python, Garbage Collection in Python, Comparison of Python with C and Java, Installing Python, Writing and Executing First Python Program, Getting Help, IDLE **Data Types, Variables and Other Basic Elements:** Comments, Docstrings, Data types- Numeric Data type, Compound Data Type, Boolean Data type, Dictionary, Sets, Mapping, Basic Elements of Python, Variables **Input and Output Operations:** Input Function, Output Statements, The print() function, The print("string") function, The print(variables list) function, , The print(object) function, The print(formatted string) function, Command Line Arguments	**15**

	Control Statements: The if statement, The if … else Statement, The 'if … elif … else' Statement, Loop Statement- while loop, for loop, Infinite loop, Nested loop, The else suite, break statement, continue statement, pass statement, assert statement, return statement	
II	**Operators:** Arithmetic operators, Assignment operators, Unary minus operator, Relational operators, Logical operators, Bitwise operators, Membership operators, Identity operators, Precedence of Operators, Associativity of Operators **Arrays:** Creating Arrays, Indexing and Slicing of Arrays, Basic Array Operations, Arrays Processing, Mathematical Operations on Array, Aliasing Arrays, Slicing and Indexing in NumPy Arrays, Basic slicing, Advanced Indexing, Dimensions of Arrays, Attributes of an Array, The ndim Attribute, The shape Attribute, The size Attribute, The itemsize Attribute **Functions:** Function definition and call, Returning Results, Returning Multiple Values from a Function, Built-in Functions, Difference between a Function and a Method, Pass Value by Object Reference, Parameters and Arguments, Formal and Actual Arguments, Positional Arguments, Keyword Arguments, Default Arguments, Arbitrary Arguments, Recursive Functions, Anonymous or Lambda Functions, Using Lambda with the filter() Function, Using Lambda with the map() Function, Using Lambda with the reduce() Function **Modules**:Introduction to Modules in Python	**15**
III	**Strings:** Creating Strings, Functions of Strings, Working with Strings, Length of a String, Indexing and Slicing, Repeating and Concatenating Strings, Checking Membership, Comparing Strings, Removing Spaces, Finding Substrings, Counting Substrings, Immutability, Splitting and Joining Strings, Changing Case, Checking Starting and Ending of a String, Sorting Strings, Searching in the Strings, Testing Methods, Formatting Strings, Finding the Number of Characters and Words, Inserting Substrings into a String **List and Tuples:** Lists, List Functions and Methods, List Operations, List Slices, Nested Lists, Tuples, Functions in Tuple **Dictionaries:** Creating a Dictionary, Operators in Dictionary, Dictionary Methods, Using for Loop with Dictionaries, Operations on Dictionaries, Converting Lists into Dictionary, Converting Strings into Dictionary, Passing Dictionaries to Functions, Sorting the Elements of a Dictionary using Lambda, Ordered Dictionaries	**15**

Textbooks:

1. Practical Programming: An Introduction to Computer Science Using Python 3, Paul Gries , Jennifer Campbell, Jason Montojo, Pragmatic Bookshelf, 2nd Edition, 2014

2. Programming through Python, M. T Savaliya, R. K. Maurya& G M Magar, Sybgen Learning India, 2020

Additional References:

1. Python: The Complete Reference, Martin C. Brown, McGraw Hill, 2018
2. Beginning Python: From Novice to Professional, Magnus Lie Hetland, Apress, 2017
3. Programming in Python 3, Mark Summerfield, Pearson Education, 2nd Ed, 2018
4. Python Programming: Using Problem Solving Approach, ReemaThareja, Oxford Univeristy Press, 2017
5. Let Us Python, Yashwant. B. Kanetkar, BPB Publication, 2019

Course Code	**Course Title**	**Credits**	**Lectures /Week**
USCSP102	**Introduction to Programming with Python – Practical**	**1**	**3**
1	Write a program to design and develop python program to implement various control statement using suitable examples		
2	Write program in Python to define and call functions for suitable problem.		
3	Write Python program to demonstrate different types of function arguments.		
4	Write a Python program to demonstrate the precedence and associativity of operators.		
5	Write suitable Python program to implement recursion for problems such as Fibonacci series, Factorial, Tower of Hanoi etc.		
6	Write Python program to implement and use lambda function in python		
7	Write a python program to create and manipulate arrays in Python. Also demonstrate use of slicing and indexing for accessing elements from the array.		
8	Write a program to implement list in Python for suitable problem. Demonstrate various operations on it.		
9	Write a program to implement tuple in Python for suitable problem. Demonstrate various operations on it.		
10	Write a program to implement dictionary in Python for suitable problem. Demonstrate various operations on it.		

Course Code	Course Title	Credits	Lectures /Week
USCS103	**LINUX Operating System**	**2**	**3**

About the Course:
This syllabus will help to train students in fundamental skills and build-up sustainable interest in Linux Operating System. It will improve necessary knowledge base to understand Linux Operating System and its practical implementation, it will also help to develop Linux based solutions for real life problems.

Course Objectives:

- To learn basic concepts of Linux in terms of operating system
- To learn use of various shell commands with regular expressions
- To set Linux Environment variables and learn setting file permissions to maintain Linux security implementation
- To learn various editors available in Linux OS
- To learn shell scripting.
- To learn installation of compilers and programming using C and Python languages on Linux platform

Learning Outcomes:
After successful completion of this course, students would be able to

- Work with Linux file system structure, Linux Environment
- Handle shell commands for scripting, with features of regular expressions, redirections
- Implement file security permissions
- Work with vi, sed and awk editors for shell scripting using various control structures
- Install softwares like compilers and develop programs in C and Python programming languages on Linux Platform

Unit	Topics	No of Lectures
I	**Linux operating system and Basics :** History, GNU Info and Utilities, Various Linux Distributions, The Unix/Linux architecture, Features of Unix/Linux, Starting the shell, Shell prompt, Command structure, File Systems and Directory Structure, man pages, more documentation pages **Basic Bash shell commands:** General purpose utility Commands, basic commands, Various file types, attributes and File handling Commands, Handling Ordinary Files. More file attributes **Advanced Bash shell commands:**Simple Filters, Filters using regular expressions. **The Linux environment variable:** Setting, Locating and removing environment variables like PATH etc, Default shell environment variables, Using command aliases.	**15**

II	**Understanding Linux file permission:** Linux security, Using Linux groups, Decoding file permissions, Changing security setting, Sharing files. **Linux Security:** Understanding Linux Security, uses of root, sudo command, working with passwords, Understanding ssh. **Networking:** TCP/IP Basics, TCP/IP Model, Resolving IP addresses, Applications, ping, telnet, ftp, DNS **Working withEditors:** awk, sed and Introduction to vi	**15**
III	**Basic script building:** Using multiple commands, Creating script files, Displaying messages, Using variables, Redirecting Input and Output, Pipes performing math, Exiting the script. **Using structured commands:** Working with if-then, if-then-else and nested if statements, test command, Compound condition testing, while command, until command, case command. **Script and Process control :** Handling signals, Running scripts in background mode, Running scripts without a console, Job control, Job scheduling commands: ps, nice, renice, at, batch, cron table, Running the script at boot	**15**

Textbooks:

1. "Linux Command line and Shell Scripting Bible", Richard Blum, Wiley India.
2. "Unix: Concepts and Applications", Sumitabha Das, 4th Edition, McGraw Hill.
3. "Official Ubuntu Book", Matthew Helmke& Elizabeth K. Joseph with Jose Antonio Rey and Philips Ballew, 8th Ed.

Additional References:

1. "Linux Administration: A Beginner's Guide", Fifth Edition, Wale Soyinka, Tata McGraw-Hill, 2008.
2. "Linux: Complete Reference", Richard Petersen, 6th Edition, Tata McGraw-Hill
3. "Beginning Linux Programming", Neil Mathew, 4th Edition, Wiley Publishing, 2008.

Course Code	Course Title	Credits	Lectures /Week
USCSP103	**LINUX Operating System – Practical**	**1**	**3**
1	**Installation of Ubuntu Linux operating system.** a) Booting and Installing from (USB/DVD) b) Using Ubuntu Software center / Using Synaptic c) Explore useful software packages.		
2	**Becoming an Ubuntu power user** a) Administering system and User setting b) Learning Unity keyboard c) Using the Terminal d) Working with windows programs		

3	**File System Commands:** touch, help, man, more, less, pwd, cd, mkdir, rmdir, ls, find, ls, etc **File handling Commands:** cat, cp, rm, mv, more, file, wc, od, cmp, diff, comm, chmod, chown, chgrp, gzip and gunzip, zip and unzip, tar, ln, umask,, chmod, chgrp, chown, etc
4	**General purpose utility Commands:**cal, date, echo, man, printf, passwd, script, who, uname, tty, stty, etc **Simple Filters and I/O redirection:** head, tail, cut paste, sort, grep family, tee, uniq, tr, etc. **Networking Commands:** who, whoami, ping, telnet, ftp, ssh, etc
5	**Editors:** vi, sed, awk
6	**Working and Managing with processes-** sh, ps, kill, nice, at and batch etc.
7	**Shell scripting I:** Defining variables, reading user input, exit and exit status commands, , expr, test, [], if conditional, logical operators
8	**Shell scripting II:** Conditions (for loop, until loop and while loop) arithmetic operations, examples
9	**Shell scripting III:** Redirecting Input / Output in scripts, creating your own Redirection
10	Installation of C/C++/Java/Python Compiler and creating an environment for app development. Basic programming using C and Python Languages.

Course Code	Course Title	Credits	Lectures /Week
USCS104	**Open Source Technologies**	**2**	**3**

About the Course:

Open Source Software is becoming an important resource for development, especially in developing countries. A working understanding of the economic and technical background of the Free / Open Source Software movement (FOSS) is essential for its effective use. The course takes students through the history and current status of the FOSS world, and starts them exploring it, by connecting their personal experiences with corresponding FOSS projects. Students will experience finding and using Open Source Software projects.

Course Objectives:

- Understand the difference between open-source software and commercial software.
- Understand the policies, licensing procedures and ethics of FOSS.
- Understand open-source philosophy, methodology and ecosystem.
- Awareness with Open-Source Technologies.

Learning Outcomes:

- Differentiate between Open Source and Proprietary software and Licensing.
- Recognize the applications, benefits and features of Open-Source Technologies
- Gain knowledge to start, manage open-source projects.

Unit	Topics	No of Lectures
I	**Introduction to Open-Source:** Open Source, Need and Principles of OSS, Open-Source Standards, Requirements for Software, OSS success, Free Software, Examples, Licensing, Free Vs. Proprietary Software, Free Software Vs. Open-Source Software, Public Domain. History of free software, Proprietary Vs Open-Source Licensing Model, use of Open-Source Software, FOSS does not mean no cost. History: BSD, The Free Software Foundation and the GNU Project. **Open-Source Principles and Methodology:** Open-Source History, Open-Source Initiatives, Open Standards Principles, Methodologies, Philosophy, Software freedom, Open-Source Software Development, Licenses, Copyright vs. Copy left, Patents, Zero marginal cost, Income-generation Opportunities, Internationalization. **Licensing:** What Is A License, How to create your own Licenses, Important FOSS Licenses (Apache, BSD, PL, LGPL), copyrights and copy lefts, Patent.	**15**
II	**Open-Source projects:** Starting and maintaining own Open-Source Project, Open-Source Hardware, Open-Source Design, Open-source Teaching, Open-source media. **Collaboration:** Community and Communication, Contributing to Open-Source Projects Introduction to GitHub, interacting with the community on GitHub, Communication and etiquette, testing open-source code, reporting	**15**

	issues, contributing code. Introduction to Wikipedia, contributing to Wikipedia or contributing to any prominent open-source project of student's choice. **Open-Source Ethics and Social Impact:** Open source vs. closed source, Open-source Government, Ethics of Open-source, Social and Financial impacts of open-source technology, Shared software, Shared source, Open Source as a Business Strategy	
III	**Understanding Open-Source Ecosystem**: Open-Source Operating Systems: GNU/Linux, Android, Free BSD, Open Solaris. Open-Source Hardware, Virtualization Technologies, Containerization Technologies: Docker, Development tools, IDEs, Debuggers, Programming languages, LAMP, Open-Source Database technologies **Case Studies:** Example Projects: Apache Web server, BSD, GNU/Linux, Android, Mozilla (Firefox), Wikipedia, Drupal, WordPress, Git, GCC, GDB, GitHub, Open Office, LibreOffice Study: Understanding the developmental models, licensing, mode of funding, commercial/non-commercial use.	**15**

Textbooks:

1. "Open-Source Technology", Kailash Vadera&Bhavyesh Gandhi, University Science Press, Laxmi Publications, 2009
2. "Open-Source Technology and Policy", Fadi P. Deek and James A. M. McHugh, Cambridge University Press, 2008.

Additional References:

1. "Perspectives on Free and Open-Source Software", Clay Shirky and Michael Cusumano, MIT press.
2. "Understanding Open Source and Free Software Licensing", Andrew M. St. Laurent, O'Reilly Media.
3. "Open Source for the Enterprise", Dan Woods, GautamGuliani, O'Reilly Media
4. Linux kernel Home: http://kernel.org4
5. Open-Source Initiative: https://opensource.org/5
6. The Linux Foundation: http://www.linuxfoundation.org/
7. The Linux Documentation Project: http://www.tldp.org/2
8. Docker Project Home: http://www.docker.com3.
9. Linux Documentation Project: http://www.tldp.org/6
10. Wikipedia: https://en.wikipedia.org/7.https://en.wikipedia.org/wiki/Wikipedia:Contributing_to_Wikipedia8
11. GitHub: https://help.github.com/9.
12. The Linux Foundation: http://www.linuxfoundation.org/

Course Code	Course Title	Credits	Lectures /Week
USCSP104	**Open Source Technologies– Practical**	**1**	**3**
1	**Open Source Operating Systems** • Learn the following open source operating system of your choice: Linux, Android, FreeBSD, Open Solaris etc. • Learn the installation. • Identify the unique features of these OS.		
2	Hands on with **LibreOffice** • Learn it from practical view-point • Give a brief presentation about it to the class		
3	Hands on with **GIMP** Photo Editing Tool • Learn it from practical view-point • Give a brief presentation about it to the class		
4	Hands on with **Shotcut** Video Editing Tool • Learn it from practical view-point • Give a brief presentation about it to the class		
5	Hands on with **Blender** Graphics and Animation Tool • Learn it from practical view-point • Give a brief presentation about it to the class		
6	Hands on with **Apache** Web Server • Learn it from practical view-point • Give a brief presentation about it to the class		
7	Hands on with **WordPress** CMS • Learn it from practical view-point • Give a brief presentation about it to the class		
8	Contributing to **Wikipedia**: • Introduction to wikipedia: operating model, license, how to contribute? • Create your user account on wikipedia • c. Identify any topic of your choice and contribute the missing information		
9	**Github** • Create and publish your own open source project: Write any simple program using your choice of programming language. • Create a repository on github and save versions of your project. You'll learn about the staging area, committing your code, branching, and merging, • Using GitHub to Collaborate: Get practice using GitHub or other remote repositories to share your changes with others and collaborate on multi-developer projects. You'll learn how to make and review a pull request on GitHub. • d. Contribute to a Live Project: Students will publish a repository containing		

	their reflections from the course and submit a pull request.
10	**Virtualization:** Open Source virtualization technologies: • Install and configure the following: VirtualBox, Zen, KVM • Create and use virtual machines
11	**Containerization:** • Install and configure the following containerization technologies: docker, rocket, LXD • Create and use containers using it

Course Code	Course Title	Credits	Lectures /Week
USCS105	**Discrete Mathematics**	**2**	**3**

About the Course:
Discrete Mathematics provides an essential foundation for virtually every area of Computer Science. The problem-solving techniques honed in Discrete Mathematics are necessary for writing complicated software. Discrete mathematics also builds the gateway to advanced courses in Mathematical Sciences, Data Science, Machine Learning, Software Engineering, etc.

Course Objectives:

- The purpose of the course is to familiarize the prospective learners with mathematical structures that are fundamentally discrete.
- This course will enhance prospective learners to reason and ability to articulate mathematical problems.
- This course will introduce functions, forming and solving recurrence relations and different counting principles. These concepts will be useful to study or describe objects or problems in computer algorithms and programming languages and these concepts can be used effectively in other courses.

Learning Outcomes:
After successful completion of this course, learners would be able to:

- Define mathematical structures (relations, functions, graphs) and use them to model real life situations.
- Understand, construct and solve simple mathematical problems.
- Solve puzzles based on counting principles.
- Provide basic knowledge about models of automata theory and the corresponding formal languages.
- Develop an attitude to solve problems based on graphs and trees, which are widely used in software.

Unit	Topics	No of Lectures
I	**Functions:** Definition of function; Domain, co-domain, range of a function; Examples of standard functions such as identity and constant functions, absolute value function, logarithmic and exponential functions, flooring and ceiling functions; Injective, surjective and bijective functions; Composite and inverse functions. **Relations:** Definition and examples of relation; Properties of relations, Representation of relations using diagraphs and matrices; Equivalence relation; Partial Order relation, Hasse Diagrams, maximal, minimal, greatest, least element, Lattices.	**15**

	Recurrence Relations: Definition and Formulation of recurrence relations; Solution of a recurrence relation; Solving recurrence relations- Back tracking method, Linear homogeneous recurrence relations with constant coefficients; Homogeneous solution of linear homogeneous recurrence relation with constant coefficients; Particular solution of non-linear homogeneous recurrence relation with constant coefficients; General solution of non-linear homogeneous recurrence relation with constant coefficients; Applications- Formulate and solve recurrence relation for Fibonacci numbers, Tower of Hanoi, Intersection of lines in a plane, Sorting Algorithms.	
II	**Counting Principles:** Basic Counting Principles (Sum and Product Rule); Pigeonhole Principle (without proof) - Simple examples; Inclusion Exclusion Principle (Sieve formula) (without proof); Counting using Tree diagrams. **Permutations and Combinations:** Permutation without and with repetition; Combination without and with repetition; Binomial numbers and identities: Pascal Identity, Vandermonde's Identity, Pascal triangle, Binomial theorem (without proof) and applications; Multionomial numbers, Multinomial theorem (without proof) and applications. **Languages, Grammars and Machines:** Languages and Grammars – Introduction, Phase structure grammar, Types of grammar, derivation trees; Finite-State Machines with Output; Finite-State Machines with No Output; Regular Expression and Regular Language.	**15**
III	**Graphs:** Graphs and Graph Models; Graph terminologies and Special types of graphs; Definition and elementary results; Representing graphs, Linked representation of a graph; Graph Isomorphism; Connectivity in graphs – path, trail, walk; Euler and Hamilton paths; Planar graphs, Graph coloring and chromatic number. **Trees:** Definition, Tree terminologies and elementary results; Linked representation of binary trees; Ordered rooted tree, Binary trees, Complete and extended binary trees, Expression trees, Binary Search tree, Algorithms for searching and inserting in binary search trees, Algorithms for deleting in a binary search tree; Traversing binary trees	**15**

Textbooks:

1. Discrete Mathematics and Its Applications, Seventh Edition by Kenneth H. Rosen, McGraw Hill Education (India) Private Limited. (2011)
2. Discrete Mathematics: SemyourLipschutz, Marc Lipson, Schaum's out lines, McGraw- Hill Inc.

3rd Edition
3. Data Structures Seymour Lipschutz, Schaum's out lines, McGraw- Hill Inc. 2017
4. Norman L. Biggs, Discrete Mathematics, Revised Edition, Clarendon Press, Oxford 1989.

Additional References:

1. Elements of Discrete Mathematics: C.L. Liu, Tata McGraw- Hill Edition.
2. Concrete Mathematics (Foundation for Computer Science): Graham, Knuth, Patashnik Second Edition, Pearson Education.
3. Discrete Mathematics: SemyourLipschutz, Marc Lipson, Schaum's out lines, McGraw- Hill Inc.
4. Foundations in Discrete Mathematics: K.D. Joshi, New Age Publication, New Delhi.

Course Code	Course Title	Credits	Lectures /Week
USCSP105	**Discrete Mathematics – Practical**	**1**	**3**
1	**Functions –** a. Identify if the given mapping is a function b. Finding domain and range of a given function c. Check if the given function is injective/surjective/bijective d. Find the inverse of a given function e. Operations on functions f. Graphs of functions using any online tool		
2	**Relations –** a. Representation of relations b. Determine if the given relation satisfies equivalence relation/partial order relation c. Draw Hasse diagrams d. Find maximal, minimal, greatest, least element in a poset e. Determine if a given poset is a lattice		
3	**Recurrence Relation –** a. Solve recurrence relation using backtracking method b. Solve linear homogeneous recurrence relations with constant coefficients c. Find homogeneous, particular, general solution of a recurrence relation d. Formulate and solving recurrence relation		
4	**Counting Principles –** a. Sum and product rule b. Pigeonhole Principle c. Inclusion Exclusion Principle d. Counting using Tree diagrams		
5	**Permutations and Combinations –** a. Permutations b. Permutations with repetitions c. Combinations d. Combinations with repetitions e. Binomial numbers and Identities		

	f. Applications on Binomial theorem g. Applications on Multinomial theorem
6	**Languages and Grammars –** a. Find the language generated by given grammar b. Check if a given string belongs or not to a given language/grammar c. Operations on languages d. Identify the type of grammar
7	**Finite State Machines –** a. Check if a given string is accepted or rejected by FSM without output b. Find the output for a FSM with output c. Describe a machine (diagram/table)
8	**Regular Expression and Regular Language –** a. Describe the regular expressions represented by given language b. Describe the language represented by given regular expression
9	**Graphs –** a. Types of graph b. Properties of graph c. Representation of graph d. Graph Isomorphism e. Connectivity in graphs – path, trail, walk f. Euler and Hamilton graphs g. Planar graphs h. Graph coloring and chromatic number
10	**Trees –** a. Tree terminologies b. Types of tree c. Properties of tree d. Representation of tree e. Expression tree f. Binary Search tree g. Tree traversal

Course Code	Course Title	Credits	Lectures /Week
USCS106	**Descriptive Statistics**	**2**	**3**

About the Course:
This course is designed to provide learners with an understanding of the data and to develop an understanding of the quantitative techniques from Statistics. It also provides the knowledge of different statistical tools used for primary statistical analysis of data.

Course Objectives:

1. To develop the learners ability to deal with different types of data.
2. To enable the use of different measures of central tendency and dispersion wherever relevant.
3. To make learner aware about the techniques to check the Skewness and Kurtosis of data.
4. To make learner enable to find the correlation between different variables and further apply the regression analysis to find the exact relation between them.
5. To develop ability to analyze statistical data through R software.

Learning Outcomes:
After successful completion of this course, learners would be able to

1. Organize, manage and present data.
2. Analyze Statistical data using measures of central tendency and dispersion.
3. Analyze Statistical data using basics techniques of R.
4. Study the relationship between variables using techniques of correlation and regression.

Unit	Topics	No of Lectures
I	**Data Types and Data Presentation: Data types:** Attribute, Variable, Discrete and Continuous variable, Univariate and Bivariate distribution. Types of Characteristics, Different types of scales: nominal, ordinal, interval and ratio. **Data presentation:** Frequency distribution, Histogram, Ogive curves. **Introduction to R:** Data input, Arithmetic Operators, Vector Operations, Matrix Operations, Data Frames, Built-in Functions. Frequency Distribution, Grouped Frequency Distribution, Diagrams and Graphs, Summary statistics for raw data and grouped frequency distribution. **Measures of Central tendency:** Concept of average/central tendency, characteristics of good measure of central tendency. Arithmetic Mean (A.M.), Median, Mode - Definition, examples for ungrouped and grouped data, effect of shift of origin and change of scale, merits and demerits. Combined arithmetic mean. Partition Values: Quartiles, Deciles and Percentiles - examples for ungrouped and grouped data	**15**
II	**Measures dispersion:** Concept of dispersion, Absolute and Relative	**15**

	measure of dispersion, characteristics of good measure of dispersion. Range, Semi-interquartile range, Quartile deviation, Standard deviation - Definition, examples for ungrouped and grouped data, effect of shift of origin and change of scale, merits and demerits. Combined standard deviation, Variance. Coefficient of range, Coefficient of quartile deviation and Coefficient of variation (C.V.) **Moments:** Concept of Moments, Raw moments, Central moments, Relation between raw and central moments. **Measures of Skewness and Kurtosis:** Concept of Skewness and Kurtosis, measures based on moments, quartiles.	
III	**Correlation:** Concept of correlation, Types and interpretation, Measure of Correlation: Scatter diagram and interpretation; Karl Pearson's coefficient of correlation (r): Definition, examples for ungrouped and grouped data, effect of shift of origin and change of scale, properties; Spearman's rank correlation coefficient: Definition, examples of with and without repetition. Concept of Multiple correlation. **Regression:** Concept of dependent (response) and independent (predictor) variables, concept of regression, Types and prediction, difference between correlation and regression, Relation between correlation and regression. Linear Regression - Definition, examples using least square method and regression coefficient, coefficient of determination, properties. Concept of Multiple regression and Logistic regression.	**15**

Textbooks:

1. Goon, A. M., Gupta, M. K. and Dasgupta, B. (1983). Fundamentals of Statistics, Vol. 1, Sixth Revised Edition, The World Press Pvt. Ltd., Calcutta.
2. Gupta, S.C. and Kapoor, V.K. (1987): Fundamentals of Mathematical Statistics, S. Chand and Sons, New Delhi

Additional References:

1. Sarma, K. V. S. (2001). Statistics Made it Simple: Do it yourself on PC. Prentce Hall of India, NewDelhi.
2. Agarwal, B. L. (2003). Programmed Statistics, Second Edition, New Age International Publishers, NewDelhi.
3. Purohit, S. G., Gore S. D., Deshmukh S. R. (2008). Statistics Using R, Narosa Publishing House, NewDelhi.
4. Schaum's Outline Of Theory And Problems Of Beginning Statistics, Larry J. Stephens, Schaum's Outline Series Mcgraw-Hill

Course Code	Course Title	Credits	Lectures /Week

USCSP106	Descriptive Statistics – Practical	1	3
Problem solving and implementation using R programming			
1	**Basics of R-** a. Data input, Arithmetic Operators b. Vector Operations, Matrix Operations c. Data Frames, Built-in Functions d. Frequency Distribution, Grouped Frequency Distribution e. Diagrams and Graphs		
2	**Frequency distribution and data presentation-** a. Frequency Distribution (Univariate data/ Bivariate data) b. Diagrams c. Graphs		
3	**Measures of Central Tendency-** a. Arithmetic Mean b. Median c. Mode d. Partition Values		
4	**Measures dispersion-** a. Range and Coefficient of range b. Quartile deviation and Coefficient of quartile deviation c. Standard deviation, Variance and Coefficient of variation (C.V.)		
5	**Moments-** a. Raw moments b. Central moments		
6	**Measures of Skewness -** a. Karl Pearson's measure of Skewness b. Bowley's measure of Skewness c. Moment coefficient of Skewness		
7	**Measures of Kurtosis-** a. Moment coefficient of Kurtosis (Absolute measure) b. Moment coefficient of Kurtosis (Relative measure)		
8	**Correlation-** a. Karl Pearson's correlation coefficient b. Spearman's Rank correlation		
9	**Regression-** a. Method of least squares b. Using regression coefficients c. Properties of regression lines & regression coefficients		
10	**Summary Statistics using R-** a. Summary statistics for raw data b. Summary statistics for grouped frequency distribution c. Simple Correlation & Regression using R		
Course Code	**Course Title**	**Credits**	**Lectures /Week**

USCS107	Soft Skills	2	3

About the Course:
To help learners develop their soft skills and develop their personality along with technical skills. Focus on various communication enhancement along with academic and professional ethics.

Course Objectives:

- Understand the significance and essence of a wide range of soft skills.
- Learn how to apply soft skills in a wide range of routine social and professional settings
- Learn how to employ soft skills to improve interpersonal relationships
- Learn how to employ soft skills to enhance employability and ensure workplace and career success

Learning Outcomes:

- Learners will be able to understand the importance and types soft skills
- Learners will develop skills for Academic and Professional Presentations.
- Learners will able to understand Leadership Qualities and Ethics.
- Ability to understand the importance of stress management in their academic & professional life.

Unit	Topics	No of Lectures
I	**Introduction to Soft Skills** **Soft Skills: An Introduction –** Definition and Significance of Soft Skills; Process, Importance and Measurement of Soft Skill Development. **Personality Development:** Knowing Yourself, Positive Thinking, Johari's Window, Physical Fitness **Emotional Intelligence:** Meaning and Definition, Need for Emotional Intelligence, Intelligence Quotient versus Emotional Intelligence Quotient, Components of Emotional Intelligence, Competencies of Emotional Intelligence, Skills to Develop Emotional Intelligence **Positivity and Motivation:** Developing Positive Thinking and Attitude; Driving out Negativity; Meaning and Theories of Motivation; Enhancing Motivation Levels **Etiquette and Mannerism:** Introduction, Professional Etiquette, Technology Etiquette **Ethical Values:** Ethics and Society, Theories of Ethics, Correlation between Values and Behavior, Nurturing Ethics, Importance of Work Ethics, Problems in the Absence of Work Ethics	15
II	**Basic Skills in Communication:** **Components of effective communication:** Communication process and handling them, Composing effective messages, Non – Verbal	15

	Communication: its importance and nuances: Facial Expression, Posture, Gesture, Eye contact, appearance (dress code). **Communication Skills:** Spoken English, Phonetics, Accent, Intonation **Employment Communication:** Introduction, Resume, Curriculum Vitae, Scannable Resume, Developing an Impressive Resume, Formats of Resume, Job Application or Cover Letter **Job Interviews:** Introduction, Importance of Resume, Definition of Interview, Background Information, Types of Interviews, Preparatory Steps for Job Interviews, Interview Skill Tips, Changes in the Interview Process, FAQ During Interviews **Group Discussion:** Introduction, Ambience/Seating Arrangement for Group Discussion, Importance of Group Discussions, Difference between Group Discussion, Panel Discussion and Debate, Traits, Types of Group Discussions, topic based and Case based Group Discussion, Individual Traits	
III	**Academic and Professional Skills:** **Professional Presentation:** Nature of Oral Presentation, planning a Presentation, Preparing the Presentation, Delivering the Presentation **Creativity at Workplace:** Introduction, Current Workplaces, Creativity, Motivation, Nurturing Hobbies at Work, The Six Thinking Hat Method. **Capacity Building: Learn, Unlearn and Relearn**: Capacity Building, Elements of Capacity Building, Zones of Learning, Ideas for Learning, Strategies for Capacity Building **Leadership and Team Building:** Leader and Leadership, Leadership Traits, Culture and Leadership, Leadership Styles and Trends, Team Building, Types of Teams. **Decision Making and Negotiation:** Introduction to Decision Making, Steps for Decision Making, Decision Making Techniques, Negotiation Fundamentals, Negotiation Styles, Major Negotiation Concepts **Stress and Time Management:** Stress, Sources of Stress, Ways to Cope with Stress	**15**
Textbooks: 1. Managing Soft Skills for Personality Development – edited by B.N.Ghosh, McGraw Hill India, 2017. 2. Soft Skills: An Integrated Approach to Maximize Personality, Gajendra S. Chauhan, Sangeeta Sharma, Wiley India **Additional References**: 1. Personality Development and Soft Skills, Barun K. Mitra, Oxford Press 2. Business Communication, ShaliniKalia, Shailja Agrawal, Wiley India 3. Cornerstone: Developing Soft Skills, Sherfield, Pearson India		

Semester II

Course Code	Course Title	Credits	Lectures /Week
USCS201	**Design & Analysis of Algorithms**	**2**	**3**

About the Course:

The course covers the concepts of - (i) calculating complexity of algorithms, (ii) the essential operations like searching, sorting, selection, pattern matching & recursion, and (iii) various algorithmic strategies like greedy, divide-n-conquer, dynamic programming, backtracking and implementations of all these on basic data structures like array, list and stack.

Course Objectives:

The objectives of this course are:

- To make students understand the basic principles of algorithm design
- To give idea to students about the theoretical background of the basic data structures
- To familiarize the students with fundamental problem-solving strategies like searching, sorting, selection, recursion and help them to evaluate efficiencies of various algorithms.
- To teach students the important algorithm design paradigms and how they can be used to solve various real world problems.

Learning Outcomes:

After successful completion of this course, students would be able to

- Students should be able to understand and evaluate efficiency of the programs that they write based on performance of the algorithms used.
- Students should be able to appreciate the use of various data structures as per need
- To select, decide and apply appropriate design principle by understanding the requirements of any real life problems

Unit	Topics	No of Lectures
I	**Introduction to algorithms** - What is algorithm, analysis of algorithm, Types of complexity, Running time analysis, How to Compare Algorithms, Rate of Growth, Types of Analysis, Asymptotic Notation, Big-O Notation, Omega-Ω Notation, Theta-Θ Notation, Asymptotic Analysis, Performance characteristics of algorithms, Estimating running time / number of steps of executions on paper, Idea of Computability **Introduction to Data Structures** - What is data structure, types, Introduction to Array(1-d & 2-d), Stack and List data structures, operations on these data structures, advantages disadvantages and applications of these data structures like solving linear equations, Polynomial Representation, Infix-to-Postfix conversion	**15**
II	**Recursion** - What is recursion, Recursion vs Iteration, recursion applications like Factorial of a number, Fibonacci series & their comparative analysis with respect to iterative version, Tower of hanoi	**15**

	problem **Basic Sorting Techniques -** Bubble, Selection and Insertion Sort & their comparative analysis **Searching Techniques -** Linear Search and its types, Binary Search and their comparative analysis **Selection Techniques** - Selection by Sorting, Partition-based Selection Algorithm, Finding the Kth Smallest Elements in Sorted Order & their comparative analysis **String Algorithms** - Pattern matching in strings, Brute Force Method & their comparative analysis	
III	**Algorithm Design Techniques -** Introduction to various types of classifications/design criteria and design techniques **Greedy Technique -** Concept, Advantages & Disadvantages, Applications, Implementation using problems like - file merging problem **Divide-n-Conquer -** Concept, Advantages & Disadvantages, Applications, Implementation using problems like - merge sort, Strassen's Matrix Multiplication **Dynamic Programming** - Concept, Advantages & Disadvantages, Applications, Implementation using problems like - Fibonacci series, Factorial of a number, Longest Common subsequence **Backtracking Programming -** Concept, Advantages & Disadvantages, Applications, Implementation using problems like N-Queen Problem	**15**

Textbooks:

1. "Data Structure and Algorithm Using Python", Rance D. Necaise, Wiley India Edition, 2016.
2. "Data Structures and Algorithms Made Easy", NarasimhaKarumanchi, CareerMonk Publications, 2016.
3. "Introduction to Algorithms", Thomas H. Cormen, 3rd Edition, PHI.

Additional References:

1. "Introduction to the Design and Analysis of Algorithms", Anany Levitin, Pearson, 3rd Edition, 2011.
2. "Design and Analysis of Algorithms", S. Sridhar, Oxford University Press, 2014.

Course Code	Course Title	Credits	Lectures /Week
USCSP201	**Design & Analysis of Algorithms – Practical**	**1**	**3**

1	Programs on 1-d arrays like - sum of elements of array, searching an element in array, finding minimum and maximum element in array, count the number of even and odd numbers in array. For all such programs, also find the time complexity, compare if there are multiple methods
2	Programs on 2-d arrays like row-sum, column-sum, sum of diagonal elements, addition of two matrices , multiplication of two matrices. For all such programs, also find the time complexity, compare if there are multiple methods
3	Program to create a list-based stack and perform various stack operations.
4	Program to perform linear search and binary search on list of elements. Compare the algorithms by calculating time required in milliseconds using readymade libraries.
5	Programs to sort elements of list by using various algorithms like bubble, selection sort, and insertion sort. Compare the efficiency of algorithms.
6	Programs to select the N^{th} Max/Min element in a list by using various algorithms. Compare the efficiency of algorithms.
7	Programs to find a pattern in a given string - general way and brute force technique. Compare the efficiency of algorithms.
8	Programs on recursion like factorial, fibonacci, tower of hanoi. Compare algorithms to find factorial/fibonacci using iterative and recursive approaches.
9	Program to implement file merging, coin change problems using Greedy Algorithm and to understand time complexity.
10	Program to implement merge sort, Straseen's Matrix Multiplication using D-n-C Algorithm and to understand time complexity.
11	Program to implement fibonacci series, Longest Common Subsequence using dynamic programming and to understand time complexity. Compare it with the general recursive algorithm.
12	Program to implement N-Queen Problem, Binary String generation using Backtracking Strategy and to understand time complexity.

Course Code	Course Title	Credits	Lectures /Week
USCS202	**Advanced Python Programming**	**2**	**3**

About the Course:
This course aims to explore and enable learners to master the skills of advanced topics in Python Programming. It helps learners develops advanced skills such as working with databases, matching patterns, implementing threads and exception handling and GUI in Python. It also highlights and why Python is a useful scripting language for all developers.

Course Objectives:

- To learn how to design object-oriented programs with Python classes.
- To learn about reading, writing and implementing other operation on files in Python.
- To implement threading concept and multithreading on Python
- To design GUI Programs and implement database interaction using Python.
- To know about use of regular expression and handling exceptions for writing robust python programs.

Learning Outcomes:
After successful completion of this course, students would be able to

- Ability to implement OOP concepts in Python including Inheritance and Polymorphism
- Ability to work with files and perform operations on it using Python.
- Ability to implement regular expression and concept of threads for developing efficient program
- Ability to implement exception handling in Python applications for error handling.
- Knowledge of working with databases, designing GUI in Python and implement networking in Python

Unit	Topics	No of Lectures
I	**Working with files:** Files, opening and closing a file, working with text files containing strings, knowing whether a file exists or not, working with binary files, the 'with' statement, the seek() and tell() methods, random accessing of binary files, zipping and unzipping files, working with directories, running other programs from python program **Regular expressions:** What is a regular expression?, sequence characters in regular expressions, quantifiers in regular expressions, special characters in regular expressions, using regular expression on files, retrieving information from an html file, **Threads in python:** Difference between process and thread, types of threads, benefits of threads, creating threads, single tasking and multitasking, thread synchronization, deadlock in threads, daemon threads **Date and time in python:** Date and time now, combining date and time,	**15**

	formatting dates and times, finding durations using “time delta”, comparing two dates, sorting dates, stopping execution temporarily, knowing the time taken by a program, calendar module	
II	**Database in python:** Using SQL with python, retrieving rows from a table, inserting rows into a table, deleting rows from a table, updating rows in a table, creating database tables through python, Exception handling in databases. **Exceptions in python:** Errors in a python program, compile & run-time errors, logical error, exceptions-exception handling, types of exceptions, the except block, the assert statement, user-defined exceptions, logging the exceptions **Networking:** Protocols,server-client architecture, tcp/ip and udp communication **Graphical user interface:** Creating a GUI in python, Widget classes, Working with Fonts and Colours, working with Frames, Layout manager, Event handling	**15**
III	**OOPs in python:** Features of Object Oriented Programming system (oops)-classes and objects, encapsulation, abstraction, inheritance, polymorphism, constructors and destructors **Classes and objects:** Creating a class, the self-variable, types of variables, namespaces, types of methods, instance methods, class methods, static methods, passing members of one class to another class, inner classes **Inheritance and polymorphism:** Inheritance in python, types of inheritance- single inheritance, multilevel inheritance, hierarchical inheritance, multiple inheritance, constructors in inheritance, overriding super class constructors and methods, the super() method, method resolution order (mro), polymorphism, duck typing, operator overloading, method overloading, method overriding**,** **Abstract classes and interfaces:** Abstract class, abstract method, interfaces in python, abstract classes vs. Interfaces	**15**

Textbooks:

1. Paul Gries , Jennifer Campbell, Jason Montojo, Practical Programming: An Introduction to Computer Science Using Python 3, Pragmatic Bookshelf, 3rd Edition, 2018
2. Programming through Python, M. T Savaliya, R. K. Maurya, G M Magar, Revised Edition, Sybgen Learning India, 2020

Additional References:

1. Advanced Python Programming, Dr. Gabriele Lanaro, Quan Nguyen, SakisKasampalis, Packt Publishing, 2019
2. Programming in Python 3, Mark Summerfield, Pearson Education, 2nd Ed, 2018
3. Python: The Complete Reference, Martin C. Brown, McGraw Hill, 2018
4. Beginning Python: From Novice to Professional, Magnus Lie Hetland, Apress, 2017

5. Programming in Python 3, Mark Summerfield, Pearson Education, 2nd Ed, 2018

Course Code	Course Title	Credits	Lectures /Week
USCSP202	**Advanced Python Programming – Practical**	**1**	**3**
1	Write a program to Python program to implement various file operations.		
2	Write a program to Python program to demonstrate use of regular expression for suitable application.		
3	Write a Program to demonstrate concept of threading and multitasking in Python.		
4	Write a Python Program to work with databases in Python to perform operations such as a. Connecting to database b. Creating and dropping tables **c.** Inserting and updating into tables.		
5	Write a Python Program to demonstrate different types of exception handing.		
6	Write a GUI Program in Python to design application that demonstrates a. Different fonts and colors b. Different Layout Managers **c.** Event Handling		
7	Write Python Program to create application which uses date and time in Python.		
8	Write a Python program to create server-client and exchange basic information		
9	Write a program to Python program to implement concepts of OOP such as a. Types of Methods b. Inheritance c. Polymorphism		
10	Write a program to Python program to implement concepts of OOP such as a. Abstract methods and classes b. Interfaces		

Course Code	Course Title	Credits	Lectures /Week
USCS203	**Introduction to OOPs using C++**	**2**	**3**

About the Course:

The course aims to introduce a new programming paradigm called Object Oriented Programming. This will be covered using C++ programming language. C++ is a versatile programming language, which supports a variety of programming styles, including procedural, object-oriented, and functional programming. This makes C++ powerful as well as flexible. It can be used to develop software such as operating systems, databases, and compilers.

Course Objectives:

Learning Outcomes:

After successful completion of this course, students would be able to

- Work with numeric, character and textual data and arrays.
- Understand the importance of OOP approach over procedural language.
- Understand how to model classes and relationships using UML.
- Apply the concepts of OOPS like encapsulation, inheritance and polymorphism.
- Handle basic file operations.

Unit	Topics	No of Lectures
I	**Introduction to Programming Concepts:** Object oriented programming paradigm, basic concepts of object oriented programming, benefits of object oriented programming, object oriented languages, applications of object oriented programming. Tokens-keywords, identifiers, constants-integer, real, character and string constants, backslash constants, features of C++ and its basic structure, simple C++ program without class, compiling and running C++ program. **Data Types, Data Input Output and Operators:** Basic data types, variables, rules for naming variables, programming constants, the type cast operator, implicit and explicit type casting, cout and cin statements, operators, precedence of operators. **Decision Making, Loops, Arrays and Strings:** Conditional statements-if, if...else, switch loops- while, do...while, for, types of arrays and string and string manipulations **Unified Modeling Language (UML):** Introduction to UML & class diagrams. **Classes, Abstraction & Encapsulation:** Classes and objects, Dot Operator, data members, member functions, passing data to functions, scope and visibility of variables in function.	**15**
II	**Constructors and Destructors:** Default constructor, parameterized	**15**

	constructor, copy constructor, private constructor, destructors. **Working with objects:**Accessor - mutator methods, static data and static function, access specifiers, array of objects. **Polymorphism -** Binding-static binding & overloading, constructor overloading function overloading, operator overloading, overloading unary and binary operators. **Modelling Relationships in Class Diagrams:** Association, Aggregation-Composition and examples covering these principles	
III	**Inheritance:** Defining base class and its derived class, access specifiers, types of inheritance-single, multiple, hierarchical, multilevel, hybrid inheritance, friend function and friend class, constructors in derived classes. **Modelling Relationships**: Generalization-Specialization and examples covering these principles **Run time Polymorphism -** Dynamic Binding, Function overriding, virtual function, pure virtual function, virtual base class, abstract class. **Pointers:** Introduction to pointers, * and & operators, assigning addresses to pointer variables, accessing values using pointers, pointers to objects & this pointer, pointers to derived classes **File Handling:** File Stream classes, opening and closing file-file opening modes, text file handling, binary file handling. **Applying OOP to solve real life applications:** To cover case studies like library management, order management etc. to design classes covering all relationships	**15**

Textbooks:

1. Object Oriented Programming with C++, Balagurusamy E., 8th Edition, McGraw Hill Education India.
2. UML & C++: A Practical Guide to Object Oriented Development, Lee/Tepfenhart, Pearson Education, 2nd Edition2015

Additional References:

1. Mastering C++ by Venugopal, Publisher: McGraw-Hill Education, 2017
2. Let Us C++ by KanetkarYashwant, Publisher: BPB Publications, 2020
3. Object Oriented Analysis and Design by Timothy Budd TMH, 2001

Course Code	Course Title	Credits	Lectures /Week
USCSP203	**Introduction to OOPs using C++ – Practical**	**1**	**3**
1	Program todemonstrate use of data members & member functions.		
2	Programs based on branching and looping statements using classes.		
3	Program to demonstrate one and two dimensional arrays using classes		
4	Program to use scope resolution operator. Display the various values of the same variables declared at different scope levels.		
5	Programs to demonstrate various types of constructors and destructors.		
6	Programs to demonstrate use of public, protected & private scope specifiers.		
7	Programs to demonstrate single and multilevel inheritance		
8	Programs to demonstrate multiple inheritance and hierarchical inheritance		
9	Programs to demonstrate inheritance and derived class constructors		
10	Programs to demonstrate friend function, inline function, this pointer		
11	Programs to demonstrate function overloading and overriding.		
12	Programs to demonstrate use of pointers		
13	Programs to demonstrate text and binary file handling		

Course Code	Course Title	Credits	Lectures /Week
USCS204	**Database Systems**	**2**	**3**

About the Course:
The course introduces the core principles and techniques required in the design and implementation of database systems. It includes ER Model, Normalization, Relational Model, and Relational Algebra. It also provides students with theoretical knowledge and practical skills of creating and manipulating data with an interactive query language (MySQL). It also provide student knowledge and importance of data protection.

Course Objectives:

- To make students aware fundamentals of database system.
- To give idea how ERD components helpful in database design and implementation.
- To experience the students working with database using MySQL.
- To familiarize the student with normalization, database protection and different DCL Statements.
- To make students aware about importance of protecting data from unauthorized users.
- To make students aware of granting and revoking rights of data manipulation.

Learning Outcomes:
After successful completion of this course, students would be able to

- To appreciate the importance of database design.
- Analyze database requirements and determine the entities involved in the system and their relationship to one another.
- Write simple queries to MySQL related to String, Maths and Date Functions.
- Create tables and insert/update/delete data, and query data in a relational DBMS using MySQL commands.
- Understand the normalization and its role in the database design process.
- Handle data permissions.
- Create indexes and understands the role of Indexes in optimization search.

Unit	Topics	No of Lectures
I	**Introduction to DBMS –** Database, DBMS – Definition, Overview of DBMS, Advantages of DBMS, Levels of abstraction, Data independence, DBMS Architecture **Data models** - Client/Server Architecture, Object Based Logical Model, Record Based Logical Model (relational, hierarchical, network) **Entity Relationship Model -** Entities, attributes, entity sets, relations, relationship sets, Additional constraints (key constraints, participation constraints, weak entities, aggregation / generalization, Conceptual Design using ER (entities VS attributes, Entity Vs relationship, binary Vs ternary, constraints beyond ER)	**15**

	ER to Table- Entity to Table, Relationship to tables with and without key constraints. **DDL Statements** - Creating Databases, Using Databases, datatypes, Creating Tables (with integrity constraints – primary key, default, check, not null), Altering Tables, Renaming Tables, Dropping Tables, Truncating Tables **DML Statements** – Viewing the structure of a table insert, update, delete, Select all columns, specific columns, unique records, conditional select, in clause, between clause, limit, aggregate functions (count, min, max, avg, sum), group by clause, having clause	
II	**Relational data model**– Domains, attributes, Tuples and Relations, Relational Model Notation, Characteristics of Relations, Relational Constraints - primary key, referential integrity, unique constraint, Null constraint, Check constraint **Relational Algebra** operations (selection, projection, set operations union, intersection, difference, cross product, Joins –conditional, equi join and natural joins, division) **Functions** – String Functions (concat, instr, left, right, mid, length, lcase/lower, ucase/upper, replace, strcmp, trim, ltrim, rtrim), Math Functions (abs, ceil, floor, mod, pow, sqrt, round, truncate) Date Functions (adddate, datediff, day, month, year, hour, min, sec, now, reverse) **Joining Tables** – inner join, outer join (left outer, right outer, full outer) **Subqueries** – subqueries with IN, EXISTS, subqueries restrictions, Nested subqueries, ANY/ALL clause, correlated subqueries	**15**
III	**Schema refinement and Normal forms:** Functional dependencies, first, second, third, and BCNF normal forms based on primary keys, lossless join decomposition. **Database Protection:** Security Issues, Threats to Databases, Security Mechanisms, Role of DBA, Discretionary Access Control, Backing Up and Restoring databases **Views** (creating, altering dropping, renaming and manipulating views) **DCL Statements** (creating/dropping users, privileges introduction, granting/revoking privileges, viewing privileges), Transaction control commands – Commit, Rollback **Index Structures of Files:** Introduction, Primary index, Clustering Index, Multilevel indexes	**15**

Textbooks:

1. "Fundamentals of Database System", ElmasriRamez, NavatheShamkant, Pearson Education, Seventh edition, 2017
2. "Database Management Systems", Raghu Ramakrishnan and Johannes Gehrke, 3rd Edition, 2014
3. "Murach's MySQL", Joel Murach, 3rd Edition, 3rd Edition, 2019

Additional References:

1. "Database System Concepts", Abraham Silberschatz,HenryF.Korth,S.Sudarshan, McGraw Hill, 2017
2. "MySQL: The Complete Reference", VikramVaswani , McGraw Hill, 2017
3. "Learn SQL with MySQL: Retrieve and Manipulate Data Using SQL Commands with Ease", AshwinPajankar, BPB Publications, 2020

Course Code	**Course Title**	**Credits**	**Lectures /Week**
USCSP204	**Database Systems – Practical**	**1**	**3**
1.	Conceptual Designing using ER Diagrams (Identifying entities, attributes, keys and relationships between entities, cardinalities, generalization, specialization etc.)		
2.	Perform the following: • Viewing all databases • Creating a Database • Viewing all Tables in a Database • Creating Tables (With and Without Constraints) • Inserting/Updating/Deleting Records in a Table		
3.	Perform the following: • Altering a Table • Dropping/Truncating/Renaming Tables • Backing up / Restoring a Database		
4.	Perform the following: • Simple Queries • Simple Queries with Aggregate functions		
5.	Queries involving • Date Functions • String Functions • Math Functions		
6.	Join Queries • Inner Join • Outer Join		

7.	Subqueries • With IN clause • With EXISTS clause
8.	Converting ER Model to Relational Model and apply Normalization on database. (Represent entities and relationships in Tabular form, Represent attributes as columns, identifying keys and normalization up to 3rd Normal Form).
9.	Views • Creating Views (with and without check option) • Dropping views • Selecting from a view
10.	DCL statements • Granting and revoking permissions • Saving (Commit) and Undoing (rollback)
11.	Creating Indexes on data tables.

Course Code	Course Title	Credits	Lectures /Week
USCS205	**Calculus**	**2**	**3**

About the Course:
Calculus is a branch of mathematics that involves the study of rates of change. In Computer Science, Calculus is used in Machine Learning, Data Mining, Scientific Computing, Image Processing, and creating the graphics and physics engines for video games, including the 3D visuals for simulations.

Course Objectives:

- The primary objective of this course is to introduce the basic tools of Calculus which are helpful in understanding their applications to the real world problems.
- The course is designed to have a grasp of important concepts of Calculus in a scientific way.
- It covers topics from as basic as definition of functions to partial derivatives of functions in a gradual and logical way.
- The learner is expected to solve as many examples as possible to a get compete clarity and understanding of the topics covered.

Learning Outcomes:
After successful completion of this course, learners would be able to:

- Develop mathematical skills and enhance thinking power of learners.
- Understand mathematical concepts like limit, continuity, derivative, integration of functions, partial derivatives.
- Appreciate real world applications which uses the learned concepts.
- Skill to formulate a problem through Mathematical modelling and simulation.

Unit	Topics	No of Lectures
I	**DERIVATIVES AND ITS APPLICATIONS:** **Review of Basic Concepts:** Functions, limit of a function, continuity of a function, derivative function. **Derivative In Graphing And Applications:** Increase, Decrease, Concavity, Relative Extreme; Graphing Polynomials, Rational Functions, Cusps and Vertical Tangents. Absolute Maxima and Minima, Applied Maximum and Minimum Problems, Newton's Method.	**15**
II	**INTEGRATION AND ITS APPLICATIONS:** **Integration:** An Overview of the Area Problem, Indefinite Integral, Definition of Area as a Limit; Sigma Notation, Definite Integral, Evaluating Definite Integrals by Substitution, Numerical Integration: Simpson's Rule. **Applications of Integration:** Area between two curves, Length of a plane curve. **Mathematical Modeling with Differential Equations**: Modeling with	**15**

	Differential Equations,Separation of Variables, Slope Fields, Euler's Method, First-Order Differential Equations and Applications.	
III	**PARTIAL DERIVATIVES AND ITS APPLICATIONS:** **Functions of Several Variables**: Functions of two or more variables, Limits and Continuity of functions of two or three variables. **Partial Derivatives:** Partial Derivatives, Differentiability, Differentials, and Local Linearity, Chain Rule, Implicit Differentiation, Directional Derivatives and Gradients, **Applications ofPartial Derivatives:** Tangent Planes and Normal Vectors, Maxima and Minima of Functions of Two Variables.	**15**
Textbooks: 1. Calculus: Early transcendental (10th Edition): Howard Anton, IrlBivens, Stephen Davis, John Wiley & sons, 2012. **Additional References**: 1. Calculus and analytic geometry (9th edition): George B Thomas, Ross L Finney, Addison Wesley, 1995 2. Calculus: Early Transcendentals (8th Edition): James Stewart, Brooks Cole, 2015. 3. Calculus (10th Edition): Ron Larson, Bruce H. Edwards, Cengage Learning, 2013. 4. Thomas' Calculus (13th Edition): George B. Thomas, Maurice D. Weir, Joel R. Hass, Pearson, 2014.		

Course Code	**Course Title**	**Credits**	**Lectures /Week**
USCSP205	**Calculus – Practical**	**1**	**3**
1	**Review of Basic Concepts –** a. Functions of one variable, its domain and range, Operations on functions b. Limits of functions of one variable c. Continuity of functions of one variable d. Derivatives of functions of one variable		
2	**Applications of Derivatives I –** a. Increasing and Decreasing functions b. Concavity and inflection points c. Relative Extrema d. Absolute Extrema		
3	**Applications of Derivatives II –** a. Analysis of polynomials b. Graphing rational functions c. Graphs With Vertical Tangents And Cusps d. Newton's method to find approximate solution of an equation		

4	**Integration –** a. Finding area using rectangle method and antiderivative method b. Indefinite and definite integrals c. Properties of integrals d. Numerical integration using Simpson's rule.
5	**Applications of Integration –** a. Area between two curves b. Length of a plane curve
6	**Differential Equations –** a. Solution of a first order first degree differential equation using variable separable method b. Solution of a first order linear differential equation using integrating factor c. Numerical solution of first-order equations using Euler's method d. Modeling using differential equation
7	**Functions of Several Variables –** a. Functions of two or more variables, its domain and range, Operations on functions, level curves b. Limits of functions of two or three variables c. Continuity of functions of two or three variables
8	**Partial Derivatives I –** a. Partial derivatives of functions, First and Second order partial derivatives, Mixed derivative theorem, Higher order partial derivatives b. Differential for functions of two or three variables c. Local linear approximation for functions of two or three variables
9	**Partial Derivatives II –** a. Chain rule for functions of two or three variables b. Implicit differentiation c. Directional derivatives and gradient
10	**Applications of Partial Derivatives–** a. Tangent Planes and Normal Vectors for functions of two or three variables b. Maxima and Minima of Functions of Two Variables
NOTE	**Above Practicals can also to be implemented using SageMath/ Geogebra.**

Course Code	Course Title	Credits	Lectures /Week
USCS206	**Statistical Methods**	**2**	**3**

About the Course:
This course introduces the key concepts in probability, conditional probabilities and distribution theory, including probability laws, random variables, expectation and variance, functions of random variables and its probability distributions. Emphasis is placed on theoretical understanding combined with problem solving using various statistical inferential techniques.

Course Objectives:

- To make learner aware about basic probability axioms and rules and its application.
- To understand the concept of conditional probability and Independence of events.
- To make learner familiar with discrete and continuous random variables as well as standard discrete and continuous distributions.
- To learn computational skills to implement various statistical inferential approaches.

Learning Outcomes:
After successful completion of this course, learners would be able to

- Calculate probability, conditional probability and independence.
- Apply the given discrete and continuous distributions whenever necessary.
- Define null hypothesis, alternative hypothesis, level of significance, test statistic and p value.
- Perform Test of Hypothesis as well as calculate confidence interval for a population parameter for single sample and two sample cases.
- Apply non-parametric test whenever necessary.
- Conduct and interpret one-way and two-way ANOVA.

Unit	Topics	No of Lectures
I	**Probability:** Random experiment, sample space, events types and operations of events, Probability definition: classical, axiomatic, Elementary Theorems of probability (without proof). Conditional probability, 'Bayes' theorem, independence, Examples on Probability. **Random Variables:** Concept and definition of a discrete random variable and continuous random variable. Probability mass function, Probability density function and cumulative distribution function of discrete and continuous random variable, Properties of cumulative distribution function.	**15**
II	**Mathematical Expectation and Variance:** Expectation of a function, Variance and S.D of a random variable, properties. **Standard Probability distributions:** Introduction, properties, examples and applications of each of the following distributions: Binomial distribution, Normal distribution, Chi-square distribution, t distribution, F distribution	**15**

III	**Hypothesis testing:** One sided, Two sided hypothesis, critical region, p-value, tests based on t, Normal and F, confidence intervals. **Analysis of Variance:** One-way, two-way analysis of variance. **Non-parametric tests:** Need of non-parametric tests, Sign test, Wilicoxon's signed rank test, run test, Kruskal-Walis tests, Chi square test.	**15**

Textbooks:

1. Gupta, S.C. and Kapoor, V.K. (1987): Fundamentals of Mathematical Statistics, S. Chand and Sons, New Delhi
2. Goon, A. M., Gupta, M. K. and Dasgupta, B. (1983). Fundamentals of Statistics, Vol. 1, Sixth Revised Edition, The World Press Pvt. Ltd., Calcutta.

Additional References:

1. Mood, A. M. and Graybill, F. A. and Boes D.C. (1974). Introduction to the Theory of Statistics, Ed. 3, McGraw Hill Book Company.
2. Hoel P. G. (1971). Introduction to Mathematical Statistics, John Wiley and Sons, New York.
3. Hogg, R.V. and Craig R.G. (1989). Introduction to Mathematical Statistics, Ed. MacMillan Publishing Co., New York.
4. Walpole R. E., Myers R. H. and Myers S. L. (1985), Probability and Statistics for Engineers and Scientists
5. Agarwal, B. L. (2003). Programmed Statistics, Second Edition, New Age International Publishers, New Delhi.

Course Code	**Course Title**	**Credits**	**Lectures /Week**
USCSP206	**Statistical Methods – Practical**	**1**	**3**
1	**Probability-** a. Examples based on Probability definition: classical, axiomatic b. Examples based on elementary Theorems of probability		
2	**Conditional probability and independence-** a. Examples based on Conditional probability b. Examples based on 'Bayes' theorem c. Examples based on independence		
3	**Discrete random variable-** a. Probability distribution of discrete random variable b. Probability mass function		
4	**Continuous random variable-** a. Probability distribution of continuous random variable b. Probability density function		

5	**Mathematical Expectation and Variance-** a. Mean of discrete and continuous Probability distribution b. S.D. and variance of discrete and continuous Probability distribution
6	**Standard probability distributions-** a. Calculation of probability, mean and variance based on Binomial distribution b. Calculation of probability based on Normal distribution
7	**Large Sample tests based on Normal (Z) -** a. Test of significance for proportion (Single proportion Ho: P = Po) b. Test of significance for difference between two proportions (Double proportion Ho: P1 = P2) c. Test of significance for mean (Single mean Ho: $\mu = \mu 0$) d. Test of significance for difference between two means. (Double mean Ho: $\mu 1 = \mu 2$)
8	**Small sample tests based on t and F-** a. t-test for significance of single mean, population variance being unknown (Single mean Ho : $\mu = \mu 0$) b. t-test for significance of the difference between two sample means (Independent samples) c. t-test for significance of the difference between two sample means (Related samples) d. F-Test to Compare Two Variances
9	**Analysis of variance -** a. Perform One-way ANOVA b. Perform Two-way ANOVA
10	**Non-parametric tests-** a. Sign test and Wilcoxon Sign rank test b. Run test c. Kruskal-Wallis (H) test d. Chi-square test
Note: Practical no. 6, 7, 8, 9 can also to be implemented using R programming.	

Course Code	Course Title	Credits	Lectures /Week
USCS207	**E-Commerce & Digital Marketing**	**2**	**3**

About the Course:
This course introduces the fundamental concepts of e-commerce, its types, the various legal and ethical issues of e-commerce and different e-commerce applications. The course also aims to introduce basic principles and types of digital marketing and web and Google analytics

Course Objectives:

- To understand increasing significance of E-Commerce and its applications in Business and Various Sectors
- To provide an insight on Digital Marketing activities on various Social Media platforms and its emerging significance in Business
- To understand Latest Trends and Practices in E-Commerce and Digital Marketing, along with its Challenges and Opportunities for an Organization

Learning Outcomes:
After successful completion of this course, students would be able to

- Understand the core concepts of E-Commerce.
- Understand the various online payment techniques
- Understand the core concepts of digital marketing and the role of digital marketing in business.
- Apply digital marketing strategies to increase sales and growth of business
- Apply digital marketing through different channels and platforms
- Understand the significance of Web Analytics and Google Analytics and apply the same.

Unit	Topics	No of Lectures
I	**Introduction to E-Commerce and E- Business:** Definition and competing in the digital economy, Impact of E-Commerce on Business Models, FactorsDriving e-commerce and e-Business Models, Economics and social impact of e-Business, opportunities and Challenges, e-Commerce vs m-Commerce, Different e-Commerce Models (B2B, B2C, C2B, C2C, B2E), e-Commerce Applications: e-Trading, e-Learning, e-Shopping, Virtual Reality & Consumer Experience, Legal and Ethical issues in e-Commerce. **Overview of Electronic Payment system**s: Types of Electronic payment schemes (Credit cards, Debit cards, Smartcards, Internet banking), E-checks, E-Cash Concepts and applications of EDI and Limitation **Introduction & origin of Digital Marketing**: Traditional v/s Digital Marketing. Digital Marketing Strategy, The P-O-E-M Framework, Segmenting & Customizing Messages, The Digital landscape, Digital Advertising Market in India. Skills required in Digital Marketing. Digital Marketing Plan.	**15**
II	**Social Media Marketing:** Meaning, Purpose, types of social media websites, Social Media Engagement, Target audience, Facebook Marketing: Business through Facebook Marketing, Creating Advertising Campaigns,	**15**

	Adverts, Facebook Marketing Tools, LinkedIn Marketing: Importance of LinkedIn Marketing, Framing LinkedIn Strategy, Lead Generation through LinkedIn, Content Strategy, Analytics and Targeting, Twitter Marketing: Framing content strategy, Twitter Advertising Campaigns, YouTube Marketing: Video optimization, Promoting on YouTube, Monetization, YouTube Analytics **Email Marketing:** Types of Emails, Mailing List, Email Marketing tools, Email Deliverability & Email Marketing automation **Mobile Marketing**: Introduction, Mobile Usage, Mobile Advertising, Mobile Marketing Types, Mobile Marketing Features, Mobile Campaign Development, Mobile Advertising Analytics **Content Marketing:** Introduction, Content marketing statistics, Types of Content, Types of Blog posts, Content Creation, Content optimization, Content Management & Distribution, Content Marketing Strategy, Content creation tools and apps, Challenges of Content Marketing.	
III	**Search Engine Optimization:** Meaning, Common SEO techniques, Understanding Search Engines, basics of Keyword search, Google rankings, Link Building, Steps to optimize website, On-page and off-page optimization **Search Engine Marketing:** Introduction to SEM, Introduction to Ad Words - Google Ad Words, Ad Words fundamentals, Ad Placement, Ad Ranks, Creating Ad Campaigns, Campaign Report Generation, Display marketing, Buying Models: Cost per Click (CPC), Cost per Milli (CPM), Cost per Lead (CPL), Cost per Acquisition (CPA). **Web Analytics:** Purpose, History, Goals & objectives, Web Analytic tools & Methods. Web Analytics Mistakes and Pitfalls. **Google Analytics:** Basics of Google Analytics, Installing Google Analytics in website, Parameters of Google Analytics, Reporting and Analysis	**15**

Textbooks:

1. "E-Commerce Strategy, Technologies and Applications", Whitley, David, Tata McGraw Hill, 2017
2. Digital Marketing, Seema Gupta, McGraw Hill Education, 2nd Edition

Additional References:

1. E-Commerce by S. Pankaj, A.P.H. Publication, New Delhi
2. Fundamentals of Digital Marketing, Punit Singh Bhatia, Pearson, 2nd Edition
3. "Understanding Digital Marketing: MarketingStrategies for Engaging the Digital Generation", Damian Ryan, Calvin Jone. Kogan Page, 4th Edition

EvaluationScheme

I. Internal Evaluation for Theory Courses – 25 Marks

(i) Mid-Term Class Test– 15Marks

- It should be conducted using any **learning management system** such as **Moodle**(Modularobject-orienteddynamiclearning environment)
- The test should have **15 MCQ's** which should be solved in a time duration of **30 minutes.**

(ii) Assignment/ Case study/ Presentations– 10 Marks

- Assignment / Case Study Report / Presentation can be uploaded onany **learning management system**.

II. External Examination for Theory Courses – 75 Marks

- Duration: **2.5 Hours**
- Theory question paper pattern:

	All questions are compulsory.		
Question	**Based on**	**Options**	**Marks**
Q.1	Unit I	*Any 4 out of 6*	20
Q.2	Unit II	*Any 4 out of 6*	20
Q.3	Unit III	*Any 4 out of 6*	20
Q.4	Unit I,II and III	*Any 5 out of 6*	15

- All questions shall be compulsorywith internal choicewithin thequestions.
- Each Question maybe sub-divided into subquestions as a, b, c, d, etc.&the allocation ofMarks dependson theweightage ofthetopic.

III. Practical Examination

- Each core subjectcarries50 Marks

  40 marks + 05 marks (journal) + 05 marks (viva)
- Duration: **2 Hours**for each practical course.
- Minimum **80% practical** from each core subjects are required to be completed.
- **Certified Journal is compulsory for appearing at the time of Practical Exam**
- The final submission and evaluation of **journal in electronic form** using a Learning Management System / Platform can be promoted by college.
