SHIV NADAR UNIVERSITY

School of Engineering

Department of
Computer Science and Engineering

Bachelor of Technology Programme

2018-2022

Computer Science & Engineering Department,
School of Engineering
UG Prospectus BTech 2019 Batch

Department of **Computer Science and Engineering** at Shiv Nadar University, Gautama Buddha Nagar offers a four-year Bachelor of Engineering Program in Computer Science and Engineering**.** The B.Tech programs at SNU prepare students with a broad foundation in fundamental principles of Mathematics, Science, and Engineering, and the ability to apply this knowledge to the design, analysis, and implementation of real-life complex systems.

The flexibility of SNU's curriculum allows students to customize their programs to meet their own interests. Students combine core math, science, and engineering with electives in the arts, humanities, social sciences, and business.

Students build on this foundation throughout their program of study by engaging in experiential learning in order to gain a hands-on experience in applying classroom concepts in the real world. A final year project provides the students the opportunity to work in multi-disciplinary teams to pursue an engineering idea from conception to design. Strong partnerships with industry and funded research provide great opportunities for undergraduate student research and internships, and other experiential learning programs provide opportunities for students to gain relevant real world experience during their program of study. SNU engineering graduates are trained to understand the broad social, economic, and ethical implications of their work, and to be cognizant of their professional responsibilities. The innovative curriculum offers exceptional preparation for an engineering career, as well as for many other professions.

The program is designed to meet the **Program Outcomes** as identified by Washington Accord given as:

1. Apply knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the conceptualization of engineering models.
2. Identify, formulate, research literature and solve *complex* engineering problems reaching substantiated conclusions using first principles of mathematics and engineering sciences.
3. Design solutions for *complex* engineering problems and *design* systems, components or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal, and environmental considerations.
4. Conduct investigations of *complex* problems including design of experiments, analysis and interpretation of data, and synthesis of information to provide valid conclusions.
5. Create, select and apply appropriate techniques, resources, and modern engineering tools, including prediction and modeling, to *complex* engineering activities, with an understanding of the limitations.
6. Function effectively as an individual, and as a member or leader in diverse teams and in multi-disciplinary settings.
7. Communicate effectively on *complex* engineering activities with the engineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
8. Demonstrate understanding of the societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to engineering practice.
9. Understand and commit to professional ethics and responsibilities and norms of engineering practice.

10. Understand the impact of engineering solutions in a societal context and demonstrate knowledge of and need for sustainable development.
11. Demonstrate a knowledge and understanding of management and business practices, such as risk and change management, and understand their limitations.
12. Recognize the need for, and have the ability to engage in independent and life-long learning.

Program Educational Objectives: Graduating Students of B.Tech. program of Computer Science and Engineering will be able to:
1. Specify, design, and develop software systems for various computing platforms, which behave reliably and efficiently, and satisfy all the requirements defined by the customer.
2. Specify, design, and develop system software to allow convenient and efficient use of computing systems.
3. Evaluate, select, and use appropriate computing environment (languages, operating systems, and other software tools) to meet the computing needs of various disciplines.
4. Develop software for intelligent systems.
5. Develop application software systems for the management of information and data in organizations.
6. Work in a team using common tools and environments to achieve project objectives.
7. Recognize their professional and personal responsibility to the community.
8. Pursue life-long learning as a means of enhancing the knowledge and skills necessary to contribute to the betterment of their profession and community.

Total creits: 160

Overall Credit Distribution

S No.	Category	Credits
1.	Core Common Curriculum (CCC)*	24
2.	University Wide Elective (UWE)*	18
3.	Basic Sciences (BS)	17
4.	Engineering Sciences (ES)	9
5.	Major Core	60
6.	Major Elective	12
7.	Project	20
Total Credits		**160**

** A student must complete 18 credits each in CCC and UWE categories.*
The total of CCC + UWE credits should be 42 credits.

Basic Sciences			**17 Credits**	
S No.	**Code**	**Course**	**L:T: P**	**Credits**
1.	PHY101	Introduction to Physics-I	3:1:0	4
2.		Maths-I	3:1:0	4
3.	PHY102	Introduction to Physics-II	3:1:1	5

4.		Maths-II	3:1:0	4

	Engineering Sciences		**9 Credits**	
	Code	**Course**	**L:T:P**	**Credits**
1.	CSD101	Introduction to Computing and programming	3:0:1	4
2.	EED101	Introduction to Electrical Engineering	3:1:1	5

S No.	**Major Core**		**60 Credits**	
	Code	**Course**	**L:T:P**	**Credits**
1.	CSD201	Data Structures	3:0:1	4
2.	EED201	Signals and Systems	3:1:0	4
3.	CSD207	Object Oriented Programming in Java	3:0:1	4
4.	CSD206	Computer Organization	3:1:0	4
5.	EED206	Digital Electronics	3:1:1	5
6.	CSD205	Discrete Mathematics	3:1:0	4
7.	CSD204	Operating Systems	3:0:1	4
8.	CSD208	Computer Architecture	3:0:1	4
9.	CSD202	Introduction to Database Systems	3:0:1	4
10.	CSD209	Introduction to Probability and Statistics	3:1:0	4
11.	CSD301	Software Engineering	3:0:1	4
12.	CSD304	Computer Networks	3:0:1	4
13.	CSD303	Theory of Computation	3:0:0	3
14.	CSD302	Design and Analysis of Algorithms	3:0:1	4
15.	CSD402	Internet and Web Systems	3:0:1	4

	Project	**20 Credits**		
S No.	**Code**	**Course**	**L:T:P**	**Credits**
1.	CSD428	Software Project Management	4:0:0	4

2.	CSD429	Research Methods in Computing	4:0:0	4
3.	CSD430	Project	0:0:12	12

Semester Wise Credit Distribution

First Semester

SI. No.	Course Title	L:T:P	Credits
1.	Mathematical Methods – I	3:1:0	4
2.	Introduction to Physics-I	3:1:0	4
3.	Introduction to Computing & Programming	3:0:1	4
4.	CCC 1	3:0:0	3
	Total Credits		**15**

Second Semester

SI. No.	Course Title	L:T:P	Credits
1.	Mathematical Methods – II	3:1:0	4
2.	Introduction to Physics-II	3:1:1	5
3.	Data Structures	3:0:1	4
4.	Introduction to Electrical Engineering	3:1:1	5
5.	CCC 2	3:0:0	3
6.	CCC 3	3:0:0	3
	Total Credits		**24**

Third Semester

SI. No.	Course Title	L:T:P	Credits
1.	Object Oriented Programming in Java	3:0:1	4
2.	Computer Organization	3:1:0	4
3.	Digital Electronics	3:1:1	5
4.	Discrete Mathematics	3:1:0	4
5.	CCC 4	3:0:0	3
	Total Credits		**20**

Fourth Semester

SI. No.	Course Title	L:T:P	Credits
1.	Operating Systems	3:0:1	4
2.	Computer Architecture	3:0:1	4
3.	Introduction to Database Systems	3:0:1	4
4.	Introduction to Probability and Statistics	3:1:0	4

5.	CCC 5	3:0:0	3
6.	UWE 1	3:0:0	3
	Total Credits		**22**

Fifth Semester

SI. No.	Course Title	L:T:P	Credits
1.	Software Engineering	3:0:1	4
2.	Computer Networks	3:0:1	4
3.	Signals & Systems	3:1:0	4
4.	Major Elective – 1	3:0:0	3
5.	UWE 2	3:0:0	3
6.	CCC 6	3:0:0	3
	Total Credits		**21**

Sixth Semester

SI. No.	Course Title	L:T:P	Credits
1.	Theory of Computation	3:0:0	3
2.	Design and Analysis of Algorithms	3:0:1	4
3.	Major Elective -2	3:0:0	3
4.	Major Elective -3	3:0:0	3
5.	CCC 7	3:0:0	3
6.	UWE 3	3:0:0	3
7.	Software Project Management	4:0:0	4
	Total Credits		**23**

Seventh Semester

SI. No.	Course Title	L:T:P	Credits
1.	Internet and Web Systems	3:0:1	4
2.	Major Elective – 4	3:0:0	3
3.	UWE 4	3:0:0	3
4.	UWE 5	3:0:0	3
5.	UWE 6	3:0:0	3
6.	CCC 8	3:0:0	3
7.	Research Methods in Computing	4:0:0	4
	Total Credits		**23**

Eighth Semester

Sl. No.	Course Title	L:T: P	Credits
1.	Project	0:0:12	12
Total Credits			**12**

Computer Science & Engineering Course Prerequisites

Course Code	Course	Prerequisites
CSD101	Introduction to Computing and Programming	NA
CSD201	Data Structures	CSD101
CSD202	Introduction to Database Systems	CSD201 and CSD205
CSD204	Operating Systems	CSD206 and CSD207*
CSD205	Discrete Mathematics	NA
CSD206	Computer Organization	CSD101
CSD207	Object Oriented Programming in Java	CSD101
CSD208	Computer Architecture	CSD206
CSD209	Introduction to Probability and Statistics	CSD205
CSD301	Software Engineering	CSD201 and CSD207*
CSD302	Design and Analysis of Algorithms	CSD201 and CSD205
CSD303	Theory of Computation	CSD201 and CSD205
CSD304	Computer Networks	CSD204 and CSD207*
CSD402	Internet and Web Systems	CSD304*
CSD428	Software Project Management	CSD301
CSD429	Research Methods in Computing	CSD428*
CSD430	Semester Long Project	NA

** This prerequisite requirement can be waived at the discretion of the instructor.*

Course Description of Core courses offered by Computer Science and Engineering department:

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD101**
Course Title	**Introduction to Computing and Programming**
Credits (L:T:P)	**3:0:1**
Contact Hours (L:T:P)	**3:0:2**
Prerequisites	**NA**
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

This course includes overview of Computer Structure, Algorithmic approach to problem solving and Basic programming abilities. Topics would include Basics of computer programming, Introduction to C programming, data types, operators, control statements, functions, arrays, pointers, strings, formatted I/O, structures, unions, bit manipulation, file processing, brief introduction to data structures.

Learning Outcomes:

On successful completion of the course, students will be able to achieve the following:

1. Understand the functioning of computer components.
2. Understand the process of problem solving using computer.
3. Understand the basic terminology used in computer programming.
4. Design an algorithmic solution for a given problem.
5. Write, compile and debug programs in C language.
6. Use different data types in a computer program.
7. Design programs involving decision structures, loops and functions.
8. Explain the difference between call by value and call by reference
9. Understand the dynamics of memory by the use of pointers.
10. Use different data structures and create/update basic data files.
11. Write C program for simple applications of real life using structures and files.

Curriculum Content:

Basics of computer system: Basic hardware components, definition of compilers, assemblers, linker, loader etc. Compiling multi-file programs

C Programming Basics: Data Types, Variables, Constants, Expressions, Operators, Operator precedence and associativity, Basic Input-Output statements, Control Structures, Simple Programs in C using all the operators and control structures,

Basic Algorithms: exchange of values of two variables, Summation of set of numbers, Factorial Computation, Generation of Fibonacci Sequence, Reversing of digits of an integer, Base conversions, Character to Number conversion, Finding Square Root, Factorial, GCD, Generating Prime numbers.

Functions: Concept of Functions, Parameters, Parameter passing method, Recursion, local and global variables, scope and extent of variables, Writing programs using recursive and non-recursive functions,

Arrays and Strings: Single and Multi-Dimensional arrays – Strings, String manipulations, Writing C programs using Strings.

Structures and Unions: Declaring and using Structures, operations on Structures, arrays of structures, user defined data type, nested structures, "sizeof" operator, Unions: Difference between Unions and structures, operations on a Union, Scope of a union

Pointers: Definition and use of pointers, address operator, pointer variable, pointer arithmetic, arrays of pointers, passing arrays to functions, pointers and functions, constant pointers, pointers to functions, Pointer to structure, Self-referencing structures.

Dynamic Memory Allocation: Library functions for Dynamic Memory Allocation, Dynamic multi-dimension arrays

Command-line arguments: using argc, agrv

Textbooks:

1. Byron Gottfried, Schaum's Outline of Programming with C, McGraw-Hill.
2. Rajaraman, "*Fundamentals of Computers"*, 5th Edition, PHI Learning
3. M. Felleisen, R. B. Findler, M. Flatt, S. Krishnamurthi , "*How To Design Programs: An Introduction To Programming And Computing*", 1st Edition, PHI Learning
4. Ivor Horton "*Beginning C Programming"*, 2nd Edition , Wrox Press
5. Herbert Schildt, *"Teach Yourself C*", 3rd Edition, McGraw-Hill Osborne Media
6. Paul J. Deitel ,"*C: How to Program"*, 6th Edition , Prentice Hall
7. Kamthane ,"*Programming in C*", 2nd Edition, Pearson
8. Mittal ,"*Programming in C –A practical approach"*, 2nd Edition, Pearson
9. Brian W. Kernighan and Dennis M. Ritchie, The C Programming Language, Prentice Hall of India

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD201**
Course Title	**Data Structures**
Credits (L:T:P)	**3:0:1**
Contact Hours (L:T:P)	**3:0:2**
Prerequisites	**CSD101**
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

This course will introduce students about problem solving techniques using programs and design of algorithms and their complexity. It also includes overview of elementary data structures such as Arrays, Stack, Queues, and Linked Lists and advanced data structures such as graphs, trees and hashing. Topics would include Time and Space Complexities, Lists, Stacks and Queues, Searching and Sorting, Hashing, Basic concepts in Trees, Advanced concepts in Trees, Priority Queues and Graphs.

Learning Outcomes:

On successful completion of the course, students will be able to achieve the following:

1. Ability to analyse algorithms and their complexities (Time and Space).
2. Understand and use the concept of Abstract Data Types such as Linked Lists, Stacks and Queues.
3. Understand and apply singly Linked List, Doubly Linked List and Circular Linked List.
4. Ability to implement stacks and queues as arrays and linked data structures.
5. Ability to handle operations like searching, sorting, insertion, deletion, traversing mechanism etc. on various data structures.
6. Understand the concept of hashing and different terms related to hashing including: hash function, double hashing, collision, collision resolution, etc.
7. Understand and implement Binary Trees, Complete Binary Tree, Threaded Binary Trees, Binary Search Trees, etc.
8. Understand and implement advanced concepts in trees including AVL Trees, Red Black Trees, B-Trees, etc.
9. Understand and use the concept of Priority Queues.
10. Understand and implement Graphs and Graph Algorithms.

Curriculum Content:

Module 1: Introduction: to Notions of data type, abstract data type, and data structures. Relation to the notion of classes and objects in object oriented programming. Importance of algorithms and data structures in programming. Notion of Complexity covering time complexity and space complexity. Worst case complexity, Average case complexity. Big Oh Notation.

Module 2: Iteration and Recursion- Problem solving using iteration and recursion with examples such as binary search, Fibonacci numbers, and Hanoi towers.Tradeoffs between iteration and recursion.

Module 3: List ADT. Implementation of lists using arrays and pointers. Stack ADT. Queue ADT. Implementation of stacks and queues. Dictionaries, Hash tables: open tables and closed tables. Analysis of hashing. Skip lists and analysis.

Module 4: Binary Trees- Definition and traversals: preorder, postorder, inorder. Common types and properties of binary trees. Counting of binary trees. Huffman coding using binary trees. Binary search trees: worst case analysis and average case analysis. AVL trees. Splay trees. Priority Queues -Binary heaps: insert and delete min operations and analysis. Binomial queues.
Module 6: Directed Graphs- Data structures for graph representation. Shortest path algorithms: Dijkstra (greedy algorithm) and Bellman-Ford (dynamic programming). Depth first search and Breadth-first search. Directed acyclic graphs. Undirected Graphs- Depth-first search and breadth-first search. Minimal spanning trees and algorithms and implementation. Application to the travelling salesman problem.
Module 7: Sorting- Bubblesort, selection sort, insertion sort, Shell sort; Quicksort; Heapsort; Mergesort; Radix sort; Analysis of the sorting methods. Selecting the top k elements. Lower bound on sorting.

Text Books:

1. Data Structures and Algorithms by Alfred V. Aho, Jeffrey D. Ullman and John E. Hopcroft , AddisonWesley Series (1983)
2. Data Structures and Algorithm Analysis in Java (3rd Edition) by Mark Allen Weiss, Addison Wesley, (2011).

Reference Books:

1. T.H. Cormen, C.E. Leiserson, and R.L. Rivest. Introduction to Algorithms. The MIT Press and McGraw-Hill Book Company, Cambridge, Massacusetts, 1990 (Available in Indian Edition).
2. Steven S. Skiena. The Algorithm Design Manual. Springer, Second Edition, 2008.

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD207**
Course Title	**Object Oriented Programming using Java**
Credits (L:T:P)	**3:0:1**
Contact Hours (L:T:P)	**3:0:2**
Prerequisites	**CSD101**
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

This course includes the introductory and advanced concepts and implementation of the Object Oriented Paradigm using the Java programming language. Topic would include Introduction, Elementary Programming, Selections, Loops, Methods, Arrays, Strings, Objects and Classes, Inheritance and Polymorphism, GUI Basics and Components, Graphics, Exceptions, Abstract Classes and Interfaces, Event-Driven Programming, Binary I/O, Recursion, and Java Collections Framework.

Learning Outcomes:

On successful completion of the course, students will be able to achieve the following:

1. Ability to understand, write and execute java programs using various input and output mechanisms.
2. Understand and use constants, variables and data types used in java.
3. Understand and apply selection and loop statements.
4. Understand and apply the concepts of methods.
5. Understand and apply string-handling mechanisms.
6. Understand object oriented programming concepts.
7. Understand and use abstract classes and interfaces.
8. Understand use Inheritance and Polymorphism.
9. Understand and create Graphical user interface.
10. Understand and perform event driven programming.
11. Understand and apply exception handling.
12. Understand and apply the components of Java Collections Framework.
13. Understand and apply java database programming.

Curriculum Content:

MODULE I: Object oriented thinking

Need for oop paradigm, A way of viewing world – Agents, responsibility, messages, methods, classes and instances, class hierarchies (Inheritance), method binding, overriding and exceptions, summary of oop concepts, coping with complexity, abstraction mechanisms.

MODULE II: Java Basics

History of Java, Java buzzwords, data types, variables, scope and life time of variables, arrays, operators, expressions, control statements, type conversion and casting, simple java program, concepts of classes, objects, constructors, methods, access control, this keyword, garbage collection, overloading methods and constructors, parameter passing, recursion, nested and inner classes, exploring string class.

MODULE III: Inheritance

Hierarchical abstractions, Base class object, subclass, subtype, substitutability, forms of inheritance-specialization, specification, construction, extension, limitation, combination, benefits of inheritance, costs of inheritance. Member access rules, super uses, using final with inheritance, polymorphism-method overriding, abstract classes, the Object class.

MODULE IV: Packages and Interfaces

Defining, Creating and Accessing a Package, Understanding CLASSPATH, importing packages, differences between classes and interfaces, defining an interface, implementing interface, applying interfaces, variables in interface and extending interfaces. Exploring java.io.

MODULE V: Exception handling

Concepts of exception handling, benefits of exception handling, Termination or resumptive models, exception hierarchy, usage of try, catch, throw, throws and finally, built in exceptions, creating own exception sub classes. String handling,Exploring java.util

MODULE VI: Multithreading

Differences between multi threading and multitasking, thread life cycle, creating threads, thread priorities, synchronizing threads, interthread communication, thread groups, daemon threads. Enumerations, autoboxing, annotations ,generics.

MODULE VII: Applets and GUI

Concepts of Applets, differences between applets and applications, life cycle of an applet, types of applets, creating applets, passing parameters to applets.

Swing – Introduction, limitations of AWT, MVC architecture, components, containers, exploring swing-JApplet, JFrame and JComponent, Icons and Labels, text fields, buttons – The JButton class, Check boxes, Radio buttons, Combo boxes, Tabbed Panes, Scroll Panes, Trees, and Tables.

MODULE VIII: Event Handling

Events, Event sources, Event classes, Event Listeners, Delegation event model, handling mouse and keyboard events, Adapter classes. The AWT class hierarchy, user interface components- labels, button, canvas, scrollbars, text components, check box, check box groups, choices, lists panels – scrollpane, dialogs, menubar, graphics, layout manager – layout manager types – border, grid, flow, card and grid bag.

MODULE IX: Java Collection Framework

Interfaces: Set, List, Queue, Deque and classes: ArrayList, Vector, LinkedList, PriorityQueue, HashSet, LinkedHashSet, TreeSet.

Text Books:

1. Introduction to Java Programming, by Y. Daniel Liang, Comprehensive version, 7th edition.

Reference Books:

2. Java: The Complete Reference, by Herbert Schildt, 7th edition

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD206**
Course Title	**Computer Organization**
Credits (L:T:P)	**3:0:1**
Contact Hours (L:T:P)	**3:0:2**
Prerequisites	**CSD101**
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

The course aims at providing an overview of computer organization and how computer design has evolved, analysing major components including internal and external memory, input–output of a computer and their interconnections, analysing internal organization & architecture of CPU, computer arithmetic operations, machine instructions, need of control unit and its various implementations.

Learning Outcomes:

On successful completion of the course, students will be able to:

1. Understand evolution and history of computers.
2. Understand and analyze performance of a computer using different metrics.
3. Understanding bus interconnection system.
4. Understand and apply computer arithmetic operations.
5. Understand and apply machine instructions and addressing modes in instruction sets.
6. Understand CPU structure and function.
7. Understand fundamental concepts of control unit - hardwired & microprogrammed.
8. Understand computer memory system and memory hierarchy.
9. Understand and analyze cache memory, internal and external memory.
10. Understand and analyze I/O modules, programmed I/O, interrupt driven I/O and DMA.

Curriculum Content:

Module 1: Computer Evolution & Arithmetic

A Brief History of computers, designing for Performance, Von Neumann Architecture, Computer Components, Interconnection Structures, Bus Interconnection, Fixed and Floating point numbers, Signed numbers, Integer Arithmetic, 2's Complement method for multiplication, Booths Algorithm, Hardware Implementation, Division, Restoring and Non Restoring algorithms

Module 2: The Central Processing Unit

Machine Instruction characteristics, types of operands, types of operations, Addressing modes, Instruction formats, Instruction types, Processor organization, Register Organization, Instruction cycles, Read Write cycles, CISC RISC, instruction pipelining

Module 3: The Control Unit

Single Bus Organization, Control Unit Operations: Instruction sequencing, Micro operations and Register Transfer. Hardwired Control, Micro-programmed Control: Basic concepts, Microinstructions and micro-program sequencing

Module 4: Memory Organization

Characteristics of memory systems, Internal and External Memory, Types of memories: ROM: PROM, EPROM, EEPROM, RAM: SRAM, DRAM, SDRAM, RDRAM
High-Speed Memories: Cache Memory, Organization and Mapping Techniques, Replacement Algorithms, Cache Coherence,.
Secondary Storage: Magnetic Disk, Tape, Optical memory, CDROM, DVD
Module 5: I/O Organization
Input/Output Systems, Programmed I/O, Interrupt Driven I/O, Direct Memory Access (DMA)

Textbooks:

1. "Computer Organization and Architecture: Designing for Performance", William Stallings, 9th Edition, Pearson India, 2013.
2. "Computer Organization", Zvonko Vranesic, Safwat Zaky, Carl Hamacher, 5th Edition, McGraw Hill Education (India) Private Limited, 2011.

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD204**
Course Title	**Operating Systems**
Credits (L:T:P)	**3:0:1**
Contact Hours (L:T:P)	**3:0:2**
Prerequisites	**CSD206 and CSD207***
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

The topics covered are introductory concepts, processes, threads, process synchronization, CPU scheduling, memory management, storage, file-system and I/O systems. The topics covered are generic and not tied to any particular operating system. Labs will be conducted in Java.

Learning Outcomes:

On successful completion of the course, students will be able to:

1. Identify the fundamental functions and basic structure of operating systems.
2. Understand process concept, process and CPU scheduling, and inter-process communication
3. Understand and apply multithreading
4. Understand and apply process synchronization
5. Understand deadlock management
6. Understand main memory and virtual memory
7. Understand mass storage structures
8. Understand File Systems
9. Understand I/O Systems

Curriculum Content:

Introduction: Definition, Design Goals, Evolution; Concept of User, job and Resources; Batch processing, Multi-programming, Time sharing; Structure and Functions of Operating System.

Process Management: Process states, State Transitions, Process Control Structure, Context Switching, Process Scheduling, Threads. Process Interaction, Shared Data and Critical Section, Mutual Exclusion, Busy form of waiting, Lock and unlock primitives, Synchronization, Classical Problems of Synchronization, Semaphores, Monitors, Conditional Critical Regions, System Deadlock, Wait for Graph, Deadlock Handling Techniques: Prevention, Avoidance, Detection and Recovery.

Memory Management: Address Binding, Dynamic Loading and Linking Concepts, Logical and Physical Addresses, Contiguous Allocation, Fragmentation, Paging, Segmentation, Combined Systems, Virtual Memory, Demand Paging, Page fault, Page replacement algorithms, Global Vs Local Allocation, Thrashing, Working Set Model,.

File and Secondary Storage Management: File Attributes, File Types, File Access Methods, Directory Structure, File System Organization and Mounting, Allocation Methods, Free Space management; Disk Structure, Logical and Physical View, Disk Head Scheduling, Formatting, Swap Management. Protection & Security.UNIX/ LINUX and WINDOWS as example systems.

Introduction to Distributed Systems.

Textbooks:

1. Abraham Silberschatz, Peter B galvin, Greg gagne , "*Operating systems Concepts*" 9th Edition, Wiley
2. William Stallings , "*Operating Systems Design and Implementation*" 5th Edition ,Prentice Hall
3. Harvey M. Deitel,Paul J. Deitel,David R. Choffnes ,"*Operating systems*" 3rd Edition, Prentice Hall
4. Haldar, Aravind ,"*Operating Systems* " ,2nd Edition, Pearson

Reference Books:

1. Operating System: A Design-oriented Approach, 1st Edition by Charles Crowley, Irwin Publishing
2. Operating Systems: A Modern Perspective, 2nd Edition by Gary J. Nutt, Addison-Wesley
3. Design of the Unix Operating Systems, 8th Edition by Maurice Bach, Prentice-Hall of India
4. Understanding the Linux Kernel, 3rd Edition, Daniel P. Bovet, Marco Cesati, O'Reilly and Associates

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD302**
Course Title	**Design and Analysis of Algorithms**
Credits (L:T:P)	**3:0:1**
Contact Hours (L:T:P)	**3:0:2**
Prerequisites	**CSD201 and CSD205**
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

This course includes the introductory and advanced concepts and implementation of the concepts of asymptotic notations, theoretical and empirical analysis of iterative and recursive algorithms, randomized algorithms, divide and conquer, greedy method, dynamic programming, graph algorithms, backtracking, NP-Hard and NP-Complete problems.

Learning Outcomes:

On successful completion of the course, students will be able to:

1. For a given algorithms analyze worst-case running times of algorithms based on asymptotic analysis and justify the correctness of algorithms.
2. Describe the greedy paradigm and explain when an algorithmic design situation calls for it. For a given problem develop the greedy algorithms.
3. Describe the divide-and-conquer paradigm and explain when an algorithmic design situation calls for it. Synthesize divide-and-conquer algorithms. Derive and solve recurrence relation.
4. Describe the dynamic-programming paradigm and explain when an algorithmic design situation calls for it. For a given problems of dynamic-programming
5. develop the dynamic programming algorithms, and analyze it to determine its computational complexity.
6. For a given model engineering problem model it using graph and write the corresponding algorithm to solve the problems.
7. Explain the ways to analyze randomized algorithms (expected running time, probability of error).
8. Explain what an approximation algorithm is. Compute the approximation factor of an approximation algorithm (PTAS and FPTAS).

Curriculum Content:

Module 1: Introduction- Fundamental characteristics of an algorithm. Basic algorithm analysis – Asymptotic analysis of complexity bounds – best, average and worst-case behaviour, standard notations for expressing algorithmic complexity. Empirical measurements of performance, time and space trade-offs in algorithms. Using recurrence relations to analyze recursive algorithms – illustrations using recursive algorithms.

Module 2: Fundamental Algorithmic Strategies: Brute-Force, Greedy, Branch-and-Bound, Backtracking and Dynamic Programming methodologies as techniques for design of algorithms – Illustrations of these techniques for Problem-Solving. Heuristics – characteristics and their domains of applicability. Design of algorithms for String matching problems, Huffman Code and Data compression problems, Subset-sum and Knapsack problems.

Module 3: Graph and Tree Algorithms: Depth- and Breadth- First traversals. Shortest path algorithms, Transitive closure, Minimum Spanning Tree, Topological sort, Network Flow problems.
Module 4: Tractable and Intractable Problems: Computability.The Halting problem. Computability classes – P, NP, NP-complete and NP-hard. Cook‟s theorem. Standard NPcomplete problems Reduction techniques.
Module 5: Advanced Topics: Approximation algorithms, Randomized algorithms, Class of problems beyond NP – P SPACE.

Text Books:

1. Algorithm Design – Jon Kleinberg and Eva Tardos
2. Introduction to Algorithms – T.H. Corman et. al.

Reference Books:

1. Fundamentals of Algorithms – E. Horowitz et al.
2. Combinatorial Optimization: Algorithms and Complexity – C.H. Papadimitriou et al.

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD208**
Course Title	**Computer Architecture**
Credits (L:T:P)	**3:0:1**
Contact Hours (L:T:P)	**3:0:2**
Prerequisites	**CSD206**
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

In this course, key areas addressed are assembly Language programming, processor architectures and system-level design processes, memory systems, parallel computer organization, pipelining and latest trends in multicore architectures

Learning Outcomes:

On successful completion of the course, students will have good knowledge of:

1. Different processor architectures and system-level design processes
2. The organization and operation of parallel computer systems
3. The Flynn's classification
4. The Multivector Architecture and SIMP Computer
5. The performance metrics of microprocessors
6. Pipelining, instruction set architectures, memory addressing.
7. Various techniques to enhance a processors ability like dynamic scheduling, branch prediction, caches, and vector processors.
8. Understand multithreading by using ILP and supporting thread-level parallelism
9. The components and operation of a memory hierarchy and the range of performance issues influencing its design.
10. Several advanced optimizations to achieve cache performance.
11. The virtual memory and virtual machines
12. The modern architectures such as RISC, Super Scalar, VLIW, multi-core systems.

Curriculum Content:

Introduction: Defining Computer Architecture, Flynn's Classification of Computers, Metrics for Performance Measurement.

Memory Hierarchy Introduction, Advanced Optimizations of Cache Performance, Memory Technology and Optimizations, Virtual Memory and Virtual Machines, The Design of Memory Hierarchy.

Instruction Level Parallelism Instruction-level Parallelism: Concepts and Challenges, Basic Compiler Techniques for Exposing ILP, Reducing Branch Costs with Advanced Branch Prediction, Dynamic Scheduling, Advanced Techniques for Instruction Delivery and Speculation, Limitations of ILP, Multithreading: Exploiting Thread-Level Parallelism to Improve Uniprocessor Throughput.

Thread Level Parallelism - Introduction, Shared-Memory Multicore Systems, Performance Metrics for Shared-Memory Multicore Systems, Cache Coherence Protocols, Synchronization, Memory Consistency.

Data Level Parallelism Introduction, Vector Architecture, SIMD Instruction Set Extensions for Multimedia, Graphics Processing Units, GPU Memory Hierarchy, Detecting and Enhancing Loop- Level Parallelism.

Textbooks:

1. David A. Patterson and John L. Hennessy, Computer Organization and Design: The Hardware and Software Interface, Morgan Kaufmann Publishers, Fourth Edition.(2009)

Reference Books:

2. John L. Hennessy and David A. Patterson, Computer Architecture: A Quantitative Approach, Morgan Kaufmann Publishers(2007)

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD202**
Course Title	**Introduction to Database Systems**
Credits (L:T:P)	**3:0:1**
Contact Hours (L:T:P)	**3:0:2**
Prerequisites	**CSD201 and CSD205**
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

This course is designed to equip students with knowledge about the fundamentals of Database Management Systems. The course also has a significant lab component. Through this lab component, students will gain exposure to SQL as well as procedural SQL. Topics include introduction to DBMS (Database Management Systems), ER model, relational data model, relational algebra, normalization, indexing, query processing & optimization, transaction processing, concurrency control & recovery, and an introduction to some advanced topics such as data mining, data warehousing and Big Data.

Learning Outcomes;

On successful completion of the course, students will be able to achieve the following:

1. Understand database architecture and database system environment
2. Understand data models and schemas
3. Write database requirements analysis for building a DBMS for a given organization or subset of an organization
4. Design ER model to satisfy database requirements
5. Design a relational database using SQL, while satisfying relational model constraints
6. Understand the importance of indexing
7. Design indexes based on application requirements
8. Understand query processing & optimization, while taking trade-offs into consideration
9. Understand transaction processing, concurrency control & recovery
10. Write programs in SQL

Curriculum Content:

Unit – I Introduction to Databases and Transactions

What is database system, purpose of database system, view of data, relational databases, database architecture, and transaction management.

Unit- II Data Models

The importance of data models, Basic building blocks, Business rules, The evolution of data models, Degrees of data abstraction.

Unit-III Database Design

ER-Diagram and Unified Modeling Language Database design and ER Model:overview, ER-Model, Constraints, ER-Diagrams, ERD Issues, weak entity sets, Codd's rules, Relational Schemas, Introduction to UML Relational database model: Logical view of data, keys, integrity rules. Relational Database design: features of good relational database design, atomic domain and Normalization (1NF, 2NF, 3NF, BCNF).

Unit- IV Relational Algebra

Introduction, Selection and projection, set operations, renaming, Joins, Division, syntax, semantics. Operators, grouping and ungrouping, relational comparison.

Unit- V Constraints, Views and SQL

What is constraints, types of constraints, Integrity constraints, Views: Introduction to views, data independence, security, updates on views, comparison between tables and views SQL: data definition, aggregate function, Null Values, nested sub queries, Joined relations. Triggers.

Unit-VI Transaction management, Concurrency control and Recovery

Transaction management: ACID properties, serializability and concurrency control, Lock based concurrency control (2PL, Deadlocks),Time stamping methods, database recovery management techniques.

Text Books:

1. Fundamentals of Database Systems (7th Edition), Ramez Elmasri and Shamkant B. Navathe, Publisher: Pearson (June 30, 2017);
2. Database System Concepts (7th Edition), Avi Silberschatz, Henry F. Korth, S. Sudarshan, McGraw-Hill.

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD301**
Course Title	**Software Engineering**
Credits (L:T:P)	**3:0:1**
Contact Hours (L:T:P)	**3:0:2**
Prerequisites	**CSD201 and CSD207***
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

Learning Outcomes:

On successful completion of the course, students will be able to:

Curriculum Content:

Module 1: Introduction- Notion of Software as a Product – characteristics of a good Software Product. Engineering aspects of Software production – necessity of automation. Job responsibilities of Programmers and Software Engineers as Software developers.

Module 2: Process Models and Program Design Techniques- Software Development Process Models – Code & Fix model, Waterfall model, Incremental model, Rapid Prototyping model, 61

Spiral (Evolutionary) model. Good Program Design Techniques – Structured Programming, Coupling and Cohesion, Abstraction and Information Hiding, Automated Programming, Defensive Programming, Redundant Programming, Aesthetics. Software Modelling Tools –Data flow Diagrams, UML and XML. Jackson System Development.

Module 3: Verification and Validation: Testing of Software Products – Black-Box Testing and White-Box Testing, Static Analysis, Symbolic Execution and Control Flow Graphs – Cyclomatic Complexity. Introduction to testing of Real-time Software Systems.

Module 4: Software Project Management: Management Functions and Processes, Project Planning and Control, Organization and Intra-team Communication, Risk Management. Software Cost Estimation – underlying factors of critical concern. Metrics for estimating costs of software products – Function Points. Techniques for software cost estimation – Expert judgement, Delphi cost estimation, Work break-down structure and Process breakdown structure, COCOMO and COCOMO-II.

Module 5: Advanced Topics: Formal Methods in Software Engineering – Z notation, Hoare‟s notation. Formalization of Functional Specifications – SPEC. Support environment for Development of Software Products. Representative Tools for Editors, Linkers, Interpreters, Code Generators, Debuggers. Tools for Decision Support and Synthesis, Configuration control and Engineering Databases, Project Management. Petrinets. Introduction to Design Patterns, Aspect-oriented Programming.

Text Books:

1. Fundamentals of Software Engineering – Carlo Ghezzi et. al.
2. Software Engineering – Design, Reliability Management – Pressman**.**

Reference Books:

1. Software Engineering – Ian Sommerville.

2. Software Engineering - Shoeman.
3. Software Engineering with Abstraction – Berzins and Luqi

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD304**
Course Title	**Computer Networks**
Credits (L:T:P)	**3:0:1**
Contact Hours (L:T:P)	**3:0:2**
Prerequisites	**CSD204 and CSD207***
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

This course develops an understanding of modern network architectures from a design and performance perspective. It introduces the student to the major concepts involved in wide-area networks (WANs), local area networks (LANs) and Wireless LANs (WLANs) and provides an opportunity to learn the practical aspects using network programming as well.

Learning Outcomes:

On successful completion of the course, students will be able to:

1. Explain the functions of the different layer of the OSI Protocol.
2. Draw the functional block diagram of wide-area networks (WANs), local area networks (LANs) and Wireless LANs (WLANs) describe the function of each block.
3. For a given requirement (small scale) of wide-area networks (WANs), local area networks (LANs) and Wireless LANs (WLANs) design it based on the market available component
4. For a given problem related TCP/IP protocol developed the network programming.
5. Configure DNS DDNS, TELNET, EMAIL, File Transfer Protocol (FTP), WWW, HTTP, SNMP, Bluetooth, Firewalls using open source available software and tools.

Curriculum Content:

Module1: Introduction -Hardware and software, Data communication, Networking, Protocols and Protocol architecture, standards. Data transmission concepts. Analog and digital transmission. Transmission impairments. Layered Architecture of Computer Networks, OSI and TCP/IP architectures

Module2: Physical Layer- Guided transmission media and wireless transmission, Data encoding - Digital and analog data and signals, spread spectrum. Data communication interface - asynchronous and synchronous transmission, line configuration and interfacing. Data link control - Flow control. Error detection and error control. HDLC and other data link

protocols. Multiplexing – Frequency-division, synchronous time-division, and statistical time-division multiplexing

Module3: Link Layer: Medium Access Control: CDMA, ALOHA, and Ethernet; Link Layer Addressing and Forwarding; Spanning Trees; The Channel Allocation Problem, Multiple Access Protocols, Ethernet, Wireless LANs, Broadband Wireless, Bluetooth, Data Link Layer Switching, Switched networks. Circuit-switched networks. Switching concepts. Routing in circuit-switched networks. Control signalling. Packet switching principles. Routing and congestion control, x.25 protocol standard. LAN Technology - LAN architecture. Bus/tree, ring, star, and wireless LANs. LAN Systems - Ethernet and Fast Ethernet (CSMA/CD) Token ring and FDDI, ATM LANs, Fiber channel, wireless LANs. Bridges - Bridge operation and routing with bridges.

Module4: Network Layer: Network layer design issues. Routing algorithms, Flooding, Shortest path routing, Link Sate routing, Hierarchical routing, Broadcast and multicast routings, Routing in the Internet, Path Vector routing, OSPF routing. The network layer in the Internet: IP protocol, ARP and RARP, BOOTP, ICMP, DHCP, Network Address Translation(NAT) Internetworking

Module5: Transport Layer: TCP introduction, Reliable/Un- Reliable Transport, TCP, UDP, Congestion Control, Intra-Domain Routing: Distance-Vector, Intra-Domain Routing: Link State, Wireless Networks: 802.11 MAC, Efficiency considerations

Module6: Application Layer: DNS-The Domain Name System, Electronic Mail, HTTP, FTP, Simple network management protocol (SNMP), The World Wide Web

Module7: Web and Multimedia: The World Wide Web – client and server side of www, HTML and webpages, JAVA language, Locating on the web. Multimedia- Audio & Video, Data compression, Video on demand, Multicast backbone.

Module 8: Security: Introduction, Cryptography and Cryptanalysis, Public Key Cryptography Algorithms, RSA Algorithm, DES, Authentication and Authorization.

Text Books:

1.Computer Networks, by Andrew S Tanenbaum, PHI. (2010)

2.Data and Computer Communications, by Walliam Stallings, PHI. (2002)

Reference Books:

1.Data Communications, Computer networking on OSI, by Fred Halsall, Addison Wesley Publishing Co.1998

2. Computer Networking -A Top-Down Approach Featuring the Internet, James F. Kurose and Keith W. Ross, Addison Wesley Publishing Co. 2004

3.Computer Networks: Protocols standards and interfaces, by Uyless Black, Prentice Hall.2002

4.Data communication & Networks, by Behrou A. Forouzan, Tata McGraw Hill. 2002

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD303**
Course Title	**Theory of Computation**
Credits (L:T:P)	**3:0:1**
Contact Hours (L:T:P)	**3:0:2**
Prerequisites	**CSD201 and CSD205**
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

1. Develop a formal notation for strings, languages and machines.
2. Design finite automata to accept a set of strings of a language.
3. Prove that a given language is regular and apply the closure properties of languages.
4. Design context free grammars to generate strings from a context free language and convert them into normal forms.
5. Prove equivalence of languages accepted by Push Down Automata and languages generated by context free grammars
6. Identify the hierarchy of formal languages, grammars and machines.
7. Distinguish between computability and non-computability and Decidability and decidability.

Learning Outcomes:

On successful completion of the course, students will be able to:

1. Write a formal notation for strings, languages and machines.
2. Design finite automata to accept a set of strings of a language.
3. For a given language determine whether the given language is regular or not.
4. Design context free grammars to generate strings of context free language.
5. Determine equivalence of languages accepted by Push Down Automata and languages generated by context free grammars
6. Write the hierarchy of formal languages, grammars and machines.
7. Distinguish between computability and non-computability and Decidability and undecidability.

Curriculum Content:

Regular expressions and finite automata. Context-free grammars and push-down automata. Regular and context-free languages, pumping lemma. Turing machines and undecidability.

Textbooks:

1. Theory of Computation by M. Sipser
2. Introduction to Automata Theory, Language and Computation by Hopcroft Motwani and Ullman
3. Introduction to formal languages and automata by Peter Linz

Reference Books:

1. Harry R. Lewis and Christos H. Papadimitriou, Elements of the Theory of Computation, Pearson Education Asia.
2. Dexter C. Kozen, Automata and Computability, Undergraduate Texts in Computer

Science, Springer.
3. Michael Sipser, Introduction to the Theory of Computation, PWS Publishing.
4. John Martin, Introduction to Languages and The Theory of Computation, Tata McGraw Hill.

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD209**
Course Title	**Introduction to Probability and Statistics**
Credits (L:T:P)	**3:1:0**
Contact Hours (L:T:P)	**3:1:0**
Prerequisites	**CSD205**
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

Uncertainty is ubiquitous and probability theory provides a rational description. These are several situations in computer engineering and other disciplines, where one tries to cope with probability and uncertainty.

Learning Outcomes:

On successful completion of the course, students will be able to:

1. Have a better understanding to probabilistic systems, such as reliability, performance related issues by assigning appropriate probability distribution
2. Some elementary understanding of generation of random numbers for solving problems.
3. Employ elements of information theory in quantification and uncertainty
4. Appreciate the concepts in statistical inference in relation to estimation of parameters, testing of hypothesis and regression analysis.

Curriculum Content:

Axioms of probability, Conditional probability and independence, Bayes theorem
Random variables, Distribution function, discrete random variable, Expectation, Variance, Bernoulli and Binomial random variable. Poisson random variable, Negative binomial random variable, Geometric random variable,
Continuous random variable: Expectation of random variable, Variance, Distribution: Uniform, Normal and Exponential
Jointly distributed random variables, Independent random variable, Sum of independent random variable, Conditional distribution, Joint probability distribution, Covariance, Correlation coefficient
Generation of random numbers and elements of Monte Carlo simulation.
Elements of information theory: Entropy, Mutual information

References:

1. A first course in probability by Sheldon Ross
2. Probability and Statistics with Reliability, Queuing, and Computer Science Applications by Kishor S. Trivedi
3. Computer Networks and Systems: Queuing Theory and Performance Evaluation, by Robertazzi

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD205**
Course Title	**Discrete Mathematics**
Credits (L:T:P)	**3:1:0**
Contact Hours (L:T:P)	**3:1:0**
Prerequisites	**NA**
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

Throughout the course, students will be expected to demonstrate their understanding of Discrete Mathematics by being able to do each of the following:

1. Use mathematically correct terminology and notation.
2. Construct correct direct and indirect proofs.
3. Use division into cases in a proof.
4. Use counterexamples.
5. Apply logical reasoning to solve a variety of problems

Learning Outcomes:

On successful completion of the course, students will be able to:

1.For a given logic sentence express it in terms of predicates, quantifiers, and logical connectives

2. For a given a problem, derive the solution using deductive logic and prove the solution based on logical inference

3. For a given a mathematical problem, classify its algebraic structure

4. Evaluate Boolean functions and simplify expressions using the properties of Boolean algebra

5.Develop the given problem as graph networks and solve with techniques of graph theory.

Curriculum Content:

Sets, Operations on sets, Cartesian product of sets, General proofs of some fundamental identities on sets. Relations and Digraphs, Paths in relations and digraphs, Properties of relations, Equivalence relations and equivalence classes, Operations on relations, Connection between relations and some data structures, Transitive Closure and Warshall's algorithm. Functions: Definition, Classification of functions, Operations on functions, Recursively defined functions. Growth of Functions.Recurrence relations, Partial order relations Propositions and Logical operations, Conditional statements, Methods of proof, Mathematical induction. First order predicate, well-formed formula of predicate, quantifiers, Inference theory of predicate logic. Counting Techniques, Pigeonhole principle, Algebraic Structures: Definition, Groups, Subgroups and order, Cyclic Groups, Cosets, Lagrange's theorem, Normal Subgroups, Permutation and Symmetric groups, Group Homomorphisms, Definition and elementary properties of Rings and Fields, Integers Modulo n. Partial order sets: Definition, Partial order sets, Combination of partial order sets, Hasse diagram. Lattices: Definition, Properties of lattices – Bounded, Complemented, Modular and Complete lattice.Trees, Labeled trees, Tree searching, Undirected trees, Isomorphic trees, Minimal spanning trees, Prim's algorithm.Graphs, Euler paths and circuits, Hamiltonian paths and circuits, isomorphic graphs, Transport networks, Matching problems, Colouring graphs.

Textbooks:

1. C.L.Liu and Mohpatra, “Elements of Discrete Mathematics”, 3rd Edition ,McGraw Hill Companies.
2. Kolman, Busby, and Ross, “Discrete Mathematical Structures”, 6th Edition, Prentice Hall of India.
3. Joseph A Gallan, “Contemporary abstract algebra”, 8th Edition ,Narosa Publishing House.

Reference books:

4. J.P. Tremblay and R. Manohar, Discrete Mathematical Structure and It’s Application to Computer Science”, TMG Edition, TataMcgraw-Hill
5. Norman L. Biggs, Discrete Mathematics, 2nd Edition, Oxford University Press.
6. Schaum’s Outlines Series, Seymour Lipschutz, Marc Lipson,
7. 3. Discrete Mathematics, Tata McGraw – Hill

School	**SoE**
Department	**Computer Science and Engineering**
Course Code	**CSD402**
Course Title	**Internet and Web Systems**
Credits (L:T:P)	**3:0:1**
Contact Hours (L:T:P)	**3:0:2**
Prerequisites	**NA**
Major Core for	**CSE**
Major Elective for	**NA**

Course Summary:

This course provides an introduction to the Internet and Web Systems. The course first describes the three-tier architecture of the web and discusses about building web applications. The course also includes the study of some important topics in Computer Networks like congestion control, traffic engineering, peer to peer system and content distribution network.

Learning Outcomes:

On successful completion of the course, students will be able to:

1. Students should be able to know basics of HTML and hands on programming
2. Students should be able to CSS style sheet.
3. Students should know basics of scripting language like Java script
4. Know the concept of event programming and DOM structure
5. Understand the concept of JSON, XML and Ajax
6. Understand the basic concepts server-side programming using Node.JS
7. Understand the concept of callback, Event Loop, Event Emitter, Buffers, Streams, File System
8. Know the programming of MongoDB using node.js
9. Understand the concepts of congestion control and flow control in the Internet
10. Know the advanced routing concepts like MPLS and traffic engineering
11. Knowledge about peer to peer system and content distribution network
12. Know about the latest multimedia networking and audio and video streaming
13. Understand the basic concepts of Simple Network Management Protocol

Curriculum Content:

Week 1 Introduction to Internet and three tier architectures of the Internet and web systems, introduction to client side programming, HTML 5 Introduction, Basic, Elements, Attributes, Styles, Formatting, Colours, HTML CSS, Links, Images, Tables, Lists, Blocks, Classes, Id, Iframes, File Paths, Layout, Forms, Form Elements, Input Types, Input Attributes

Week 2 CSS Introduction, Syntax, Colors, Backgrounds, Borders, Margins, Padding, Height/Width, Box Model, Outline, Text, Fonts, Icons, Links, Lists, Tables, Display

Week 3 JS Introduction, Output, Statements, Syntax, Comments, Variables, Operators, Arithmetic, Assignment, Data Types, Functions, Objects, Scope, Events, Strings, String Methods, Numbers, Arrays, Array Methods, Array Sort, Conditions, Switch, Loop For, Loop While, Break, Type Conversion, Bitwise, RegExp, Errors, Debugging, JS this Keyword

Forms, Forms, Forms API, Objects, Object Definitions, Object Properties, Object Methods, Object Constructors, Object Prototypes, JS Functions, Function Definitions, Function Parameters, Function Invocation, Function Call, Function Apply, Function Closures

JS HTML DOM, DOM Intro, DOM Methods, DOM Document, DOM Elements, DOM HTML, DOM CSS, DOM Animations, DOM Events, DOM Event Listener, DOM Navigation, DOM Nodes, DOM Collections, DOM Node Lists, JS Timing, JS Cookies

Node.js Tutorial, Node.js HOME, Node.js Intro, Node.js Get Started, Node.js Modules, Node.js HTTP Module, Node.js File System, Node.js URL Module, Node.js NPM, Node.js Events, Node.js Upload Files Node.js Email

Node.js - Callbacks Concept, Node.js - Event Loop, Node.js - Event Emitter, Node.js - Buffers, Node.js - Streams, Node.js - File System, Node.js - Global Objects, Node.js - Utility Modules, Node.js - Web Module, Node.js - Express Framework, Node.js - RESTFul API, Node.js - Scaling Application, Node.js – Packaging

Node.js, MongoDB, MongoDB Get Started, Create Database, MongoDB Create Collection, MongoDB Insert, Find, Query, Sort, Delete, Drop Collection, Update, Limit, Join

Refresher of Computer Networks which would brush the networking concepts of Physical, data link and network layer in brief

Routing, Border Gateway Protocol and Path Vector Routing, MPLS and Traffic Engineering (Routing as optimization problem)

TCP Congestion Control and Flow Control: Fair and Efficient Use of a Resource, Max-Min Fairness, Resource Allocation over Paths, TCP Congestion Control (TCP Reno, TCP Vegas),

TCP Congestion Control and Flow Control: Congestion in Drop Tail FIFO Queue, TCP Congestion Window, Additive Increase Multiplicative Decrease, Receiver Window vs Congestion Window

Application Layer - Peer to Peer Network, Distributed Hash Table, Content Distribution Network

Multimedia Networking, Multimedia Audio and Video, Streaming audio and video, Client-Side Buffering (Netflix), Voice over IP (Skype), Real Time Protocol and Session Initiation Protocol

Network Management Protocol: Internet network management framework, MIB: management information base, SMI: data definition language, SNMP: protocol for network management, security and administration

Lab

Books/References:

1. https://www.w3schools.com/
2. https://courses.engr.illinois.edu/cs538/sp2018/
3. https://www.systems.ethz.ch/node/489
4. Peterson, Larry L., and Bruce S. Davie. Computer networks: a systems approach. Elsevier, 2007.
5. Robbins, Jennifer Niederst. Learning web design: A beginner's guide to HTML, CSS, JavaScript, and web graphics. " O'Reilly Media, Inc.", 2012.
6. Computer Networking: A Top-Down Approach 6th ed. J.F. Kurose and K.W. Ross