SHIV NADAR

INSTITUTION OF EMINENCE DEEMED TO BE UNIVERSITY

DELHI NCR

School of Engineering

Department of Computer Science and Engineering

Bachelor of Technology Program (B.Tech. – CSE with Specialization) (2022 onwards)

Department of Computer Science & Engineering

School of Engineering

UG Prospectus B.Tech. CSE with Specialization

Department of Computer Science and Engineering at Shiv Nadar University, Gautam Buddh Nagar offers a four-year Bachelor of Engineering Program in Computer Science and Engineering. The program is designed to meet the Program Outcomes as identified by the Washington Accord:

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 the 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 multidisciplinary 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 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 an 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.

<u>Structure of B.Tech. Computer Science & Engineering with Specialization</u>

Programme Structure for B.Tech. (4 year)

Overall Credit Structure - 160 Credits

S. No.	Category	Credits
1	Core Common Curriculum (CCC)	18-24*
2	University Wide Elective (UWE)	18-24*
3	Basic Sciences (BS)	17
4	Engineering Sciences (ES)	13
5	Major Core	61
6	Major Elective	15
7	Project1 + Project2/Internship	12
Total Credits		**160**

***CCC and UWE with minimum 18 credits in each category and overall CCC+UWE should be 42 credits**

All undergraduate students at SNU must take a core group of common subjects designated as Core Common Curriculum (or CCC) courses. The CCC is designed to provide students a broad-based understanding of the world, its physical, biological and social systems, the development of human civilization and culture, and the historical development and modern formation of global society with a special emphasis on the history and development of India.

A UWE course for a student is any non-CCC course outside the student's major from any department of SNU. The UWE credits for a student cannot come from courses that is either core course or elective course of the student's major. A student may use the UWE category in any desired way, with no interference from the major department. For example, a student may use UWE to pursue a variety of interests in dance, media, communication, history and sociology. Alternately, a student may concentrate the UWE credits in one direction and use them to earn a minor degree in other department.

The CCC consists of courses in 8 Topic Areas. Each student must earn at least 1.5 credits each from any six of eight topic areas listed below:

Core Common Curriculum

S. No.	Category
1	Indian History and Society (IHS)
2	World History and Society (WHS)
3	Culture and Communication (CAS)
4	Physical and Living Systems (PLS)
5	Cognition and Intelligence (CAI)
6	Technology and Society (TAS)
7	Environment and Ecology (EAE)
8	Reasoning and Analysis (RAA)
Total Credits: 18 – 24	

Basic Sciences (BS) - 17 Credits

S. No.	Course Code	Course Title	L-T-P	Credits
1	PHY101	Introduction to Physics-1	3-1-0	4
2	PHY102	Introduction to Physics-II	3-1-1	5
3	MAT103	Mathematical Methods –I	3-1-0	4
4	MAT161	Applied Linear Algebra	3-1-0	4
Total Credits				**17**

Engineering Sciences (ES) - 13 Credits

S. No.	Course Code	Course Title	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
3	MED201	Material Science and Engineering	3-0-1	4
Total Credits				**13**

Core Projects- 12 Credits

S. No.	Course Code	Course Title	L-T-P	Credits
1	CSD493	Project-1	0-0-6	6
2	CSD494	Project-2/Internship	0-0-6	6
Total Credits				**12**

Major Core- 61 Credits

S. No.	Course Code	Course Title	L-T-P	Credits	Prerequisites
1	CSD102	Data Structures	3-0-1	4	CSD101
2	CSD204	Operating Systems	3-0-1	4	CSD102/201
3	CSD205	Discrete Mathematics	3-1-0	4	CSD101
4	CSD210	Introduction to Probability and Statistics	3-1-0	4	CSD102/201
5	CSD211	Computer Organization and Architecture	3-1-1	5	CSD101
6	CSD213	Object Oriented Programming	3-0-1	4	CSD101
7	CSD304	Computer Networks	3-0-1	4	CSD102/201
8	CSD311	Artificial Intelligence	3-0-1	4	CSD102/201
9	CSD317	Introduction to Database Systems	3-0-1	4	CSD102/201
10	CSD319	Design and Analysis of Algorithms	3-0-1	4	CSD102/201
11	CSD326	Software Engineering	3-0-1	4	CSD102/201
12	CSD334	Theory of Computation	3-0-0	3	CSD102/201
13	CSD	Optimization	3-0-1	4	CSD102/201
14	CSD	Digital Image Processing	3-0-1	4	
15	EED206	Digital Electronics	3-1-1	5	
Total Credits				**61**	

Major Electives - 15 credits

S. No.	Course Code	Course Title	L-T-P	Credits	Prerequisite
1	CSD350	Natural Language Processing	2-0-1	3	CSD102/201
2	CSD351	Advanced Database Management System	2-0-1	3	CSD317/202
3	CSD352	Computational Neuroscience	2-0-1	3	CSD102/201
4	CSD353	Computer Graphics	2-0-1	3	CSD102/201
5	CSD355	Foundation of Data Sciences	2-0-1	3	CSD317/202
6	CSD356	Foundation of Information Security	2-0-1	3	CSD102/201
7	CSD357	Image Processing and Its Applications	2-0-1	3	CSD102/201
8	CSD358	Information Retrieval	2-0-1	3	CSD102/201
9	CSD360	Introduction to Logic and Functional Programming	2-0-1	3	CSD102/201
10	CSD361	Introduction to Machine Learning	2-0-1	3	CSD102/201, CSD210/209
11	CSD363	Social and Information Networks	2-0-1	3	CSD102/201
12	CSD450	Algorithms for Big Data	3-0-0	3	CSD102/201, CSD210/209
13	CSD451	Applied Cryptography	2-0-1	3	CSD101, CSD205
14	CSD452	Big data Analytics	2-0-1	3	CSD317/202
15	CSD454	Computer Vision	2-0-1	3	CSD102/201
16	CSD455	Data Mining and Warehousing	2-0-1	3	CSD317/202, CSD210/209
17	CSD456	Deep Learning	2-0-1	3	CSD361
18	CSD457	Internet of things	2-0-1	3	CSD304
19	CSD458	Introduction to Geometric Algorithms	3-0-0	3	CSD319/302
20	CSD459	Performance Modeling and Queuing Theory	3-0-0	3	CSD210/209/ MAT205/284
21	CSD462	Virtualization and Cloud Computing	2-0-1	3	CSD102/201
22	CSD463	Wireless and Mobile Systems	3-0-0	3	CSD304
23	CSD464	Wireless Sensor Networks	2-0-1	3	CSD304
24	CSD481	Special Topics in Artificial Intelligence	3-0-0	3	CSD311
25	CSD482	Special Topics in Applications	3-0-0	3	CSD102/201
26	CSD483	Special Topics in Systems	3-0-0	3	CSD102/201
27	CSD484	Special Topics in Theoretical Computer Science	3-0-0	3	CSD102/201
28	CSD485	Special Module in Artificial Intelligence	1-0-0	1	CSD311
29	CSD486	Special Module in Applications	1-0-0	1	CSD102/201
30	CSD487	Special Module in Systems	1-0-0	1	CSD102/201
31	CSD488	Special Module in Theoretical Computer Science	1-0-0	1	CSD102/201

Semester wise Course offering

The CSE department also revised the existing scheduling of courses within the approved structure template. This has been done to create the balance of course loads each semester and difficulty level along with satisfying the useful unspecified prerequisites.

First Semester – 20 Credits				
S. No.	**Course Code**	**Course Title**	**L-T-P**	**Credits**
1	CSD101	Introduction to Computing and Programming	3-0-1	4
2	MAT103	Mathematical Methods-I	3-1-0	4
3	MED201	Material Science and Engineering	3-0-1	4
4	PHY101	Introduction to Physics –I	3-1-0	4
5	CCC704	Environmental Studies (CCC 1)	3-0-1	4

Second Semester – 21 Credits				
S. No.	**Course Code**	**Course Title**	**L-T-P**	**Credits**
1	CSD102	Data Structures	3-0-1	4
2	EED101	Introduction to Electrical Engineering	3-1-1	5
3	MAT161	Applied Linear Algebra	3-1-0	4
4	PHY102	Introduction to Physics –II	3-1-1	5
5	CCC	CCC 2		3

Third Semester – 24 Credits				
S. No.	**Course Code**	**Course Title**	**L-T-P**	**Credits**
1	CSD205	Discrete Mathematics	3-1-0	4
2	CSD211	Computer Organization and Architecture	3-1-1	5
3	CSD213	Object Oriented Programming	3-0-1	4
4	EED206	Digital Electronics	3-1-1	5
5	ESD201	UWE 1: Engineering Science and Design	2-0-1	3
6	CCC	CCC 3		3

Fourth Semester – 22 Credits				
S. No.	**Course Code**	**Course Title**	**L-T-P**	**Credits**
1	CSD204	Operating Systems	3-0-1	4
2	CSD210	Introduction to Probability and Statistics	3-1-0	4
3	CSD	Digital Image Processing	3-0-1	4
4	CSD317	Introduction to Database Systems	3-0-1	4
5		UWE 2		3
6	CCC	CCC 4		3
Summer Internship (15th May to 15th July) after 4th semester -Non Mandatory				

Fifth Semester – 24 Credits				
S. No.	**Course Code**	**Course Title**	**L-T-P**	**Credits**
1	CSD304	Computer Networks	3-0-1	4
2	CSD311	Artificial Intelligence	3-0-1	4
3	CSD319	Design and Analysis of Algorithms	3-0-1	4
4	CSD	Major Elective 1		3
5		UWE 3		3
6		UWE 4		3
7	CCC	CCC 5		3

Sixth Semester – 22 Credits				
S. No.	**Course Code**	**Course Title**	**L-T-P**	**Credits**
1	CSD	Optimization Techniques	3-0-1	4
2	CSD326	Software Engineering	3-0-1	4
3	CSD334	Theory of Computation	2-1-0	3
4	CSD	Major Elective 2		3
5	CSD	Major Elective 3		3
6		UWE 5		3
7	CCC	CCC 6		2
Summer Internship (15th May to 15th July) after 6th semester -Non Mandatory				

Seventh Semester – 21 Credits			
S. No.	**Course Code**	**Course Title**	**Credits**
1	CSD	Major Elective 4	3
2	CSD	Major Elective 5	3
3		UWE 6	3
4		UWE 7	3
5	CCC	CCC 7	3
6	CSD493	Project-1	6

Eighth Semester – 6 Credits			
S. No.	**Course Code**	**Course Title**	**Credits**
1	CSD494	Project-2/ Internship	6

Areas of Specialization

The students enrolled in B. Tech. Computer Science and Engineering (4 year) would have an option to specialize in one the following emerging areas-

- **Artificial Intelligence and Machine Learning**
- **Data Science and Big Data Analytics**
- **Cyber Security and Privacy**

Minimum Requirement for Specialization

The student wishing to do specialization must have overall CGPA $\geq$ 7 and CGPA in Specialization component $\geq$ 8. The student must complete minimum of 12 credits in the area of specialization as follows-

- Electives courses from the chosen specialization bucket- 12 Credits

OR

- Electives courses from the chosen specialization bucket- 6 Credits
- Project-I in the area of specialization- 6 Credits

At the time of graduation (end of 8th semester before convocation), students who have completed the specialization requirement may apply for a specialization in CSE to UG advisor for further processing. A student can apply for a specialization only in one of the mentioned areas.

List of Elective courses in specialization buckets

Given below are the list of courses in above specialization buckets.

Artificial Intelligence and Machine Learning-

Course Code	Course Name	L-T-P	Credits
CSD350	Natural Language Processing	2-0-1	3
CSD352	Computational Neuroscience	2-0-1	3
CSD357	Image Processing and Its Applications	2-0-1	3
CSD360	Introduction to Logic and Functional Programming	2-0-1	3
CSD361	Introduction to Machine learning	2-0-1	3
CSD454	Computer Vision	2-0-1	3
CSD456	Deep Learning	2-0-1	3
CSD481	Special Topics in Artificial Intelligence	3-0-0	3
CSD485	Special Module in Artificial Intelligence	1-0-0	1
CSD	Reinforcement Learning	2-0-1	3
CSD	Evolutionary Computing	2-0-1	3

Data Science and Big Data Analytics-

Course Code	Course Name	L-T-P	Credits
CSD350	Natural Language Processing	2-0-1	3
CSD351	Advanced Data Management Systems	2-0-1	3
CSD355	Foundation of Data Sciences	2-0-1	3
CSD358	Information Retrieval	2-0-1	3

CSD361	Introduction to Machine learning	2-0-1	3
CSD363	Social and Information Networks	2-0-1	3
CSD450	Algorithms for Big Data	3-0-0	3
CSD452	Big Data Analytics	2-0-1	3
CSD455	Data Mining and Warehousing	2-0-1	3
CSD462	Virtualization and Cloud Computing	2-0-1	3
CSD482	Special Topics in Applications	3-0-0	3
CSD486	Special Module in Applications	1-0-0	1
CSD	Stochastic Simulation and Modelling	2-0-1	3

Cyber Security and Privacy-

Course Code	Course Name	L-T-P	Credits
CSD356	Foundation of Information Security	2-0-1	3
CSD451	Applied Cryptography	2-0-1	3
CSD457	Internet of Things	2-0-1	3
CSD459	Performance Modeling and Queuing Theory	3-0-0	3
CSD463	Wireless and Mobile Systems	3-0-1	3
CSD464	Wireless Sensor Networks	2-0-1	3
CSD483	Special Topics in Systems	3-0-0	3
CSD487	Special Module in Systems	1-0-0	1
CSD	Ethical Hacking	2-0-1	3
CSD	Security Analytics	2-0-1	3
CSD	Secure Coding	2-0-1	3

Above list is tentative and can be suitably updated based on needs of the Industry and availability of relevant course in the university.

Core Course Descriptions

Course: Introduction to Computing and Programming

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD101**
Course Title	**Introduction to Computing and Programming**
Credits	**4**
L-T-P (Contact Hours)	**3-0-1 (L:3H - T:0H - P:2H)**
Prerequisites	**-**
Category	**ES**

Course Summary:

This course briefs about Computer Structure, the Algorithmic approach to solve a problem, basic introduction to computers and its corresponding concepts for the benefit of students. Apart from this, programming concepts are also discussed in this course using C programming language.

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

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

Curriculum Content:

1. Basics of a computer system: Basic hardware components, definition of compilers, assemblers, linker, loader, etc. Compiling multi-file programs.
2. 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.
3. 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.
4. 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.
5. Arrays and Strings: Single and Multi-Dimensional arrays – Strings, String manipulations, Writing C programs using Strings.
6. 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.
7. 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.
8. Dynamic Memory Allocation: Library functions for Dynamic Memory Allocation, Dynamic multi-dimension arrays, Command-line arguments: using argc, agrv.

Textbooks and References:

1. Byron Gottfried, Schaum's Outline, *Programming with C*, 3rd Edition, 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,* 6^{th}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.

Course: Data Structures

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD102**
Course Title	**Data Structures**
Credits	**4**
L-T-P (Contact Hours)	**3-0-1 (L:3H - T:0H - P:2H)**
Prerequisites	**CSD101**
Category	**Major Core**

Course Summary:

This course introduces problem-solving techniques using programs and the design of algorithms and their complexity. It includes an overview of elementary data structures and advanced data structures. Topics would include Time and Space Complexities, Searching, Sorting, Hashing, Basic and Advance 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 analyze algorithms and their complexities (Time and Space).
2. Understand and use the concept of Abstract Data Types with their respective applications.
3. Ability to handle operations like searching, sorting, insertion, deletion, traversing mechanism, etc. on various data structures.
4. Understand the concept of hashing and different terms related such as hash function, double hashing, collision, collision resolution, etc.
5. Understand and implement Basic and Advance concepts in Trees, Priority Queues, Graphs and Graph Algorithms.

Curriculum Content:

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

5. 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.
6. Sorting- Bubble sort, selection sort, insertion sort, Shell sort; Quicksort; Heapsort; Merge sort; Radix sort; Analysis of the sorting methods. Selecting the top k elements. Lower bound on sorting.

Textbooks and References:

1. Alfred V. Aho, Jeffrey D. Ullman, John E. Hopcroft, *Data Structures and Algorithms*, Addison Wesley Series, 1983.
2. Mark Allen Weiss, *Data Structures and Algorithm Analysis in Java*, 3rd Edition, Addison Wesley, 2011.
3. T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms*. The MIT Press andMcGraw-Hill Book Company, Cambridge, Massachusetts, 1990.
4. Steven S. Skiena,*The Algorithm Design Manual*, 2nd Edition, Springer, 2008.

Course: Operating Systems

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD204**
Course Title	**Operating Systems**
Credits	**4**
L-T-P (Contact Hours)	**3-0-1 (L:3H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Core**

Course Summary:
The topics covered are introductory concepts on 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.

Learning Outcomes: On successful completion of the course, students will be able to achieve the following:
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, synchronization, Memory management.
4. Understand File Systems and I/O Systems.

Curriculum Content:
1. Definition, Design Goals, Evolution; Concept of User, job and Resources; Batch processing, Multi-programming, Time sharing; Structure and Functions of Operating System.
2. 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.

3. Deadlock Handling Techniques: Prevention, Avoidance, Detection and Recovery.
4. 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.
5. 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.
6. Introduction to Distributed Systems.

Textbooks and References :

1. Abraham Silberschatz, Peter B Galvin, Greg Gagne,*Operating systems Concepts*,9th Edition, Wiley.
2. William Stallings,*Operating Systems Design and Implementation*,5thEdition, Prentice-Hall.
3. Harvey M. Deitel, Paul J. Deitel, David R. *Operating systems*,3rd Edition, Prentice-Hall.
4. Haldar, Aravind, *Operating Systems,* 2nd Edition, Pearson.
5. Charles Crowley, *Operating System: A Design-oriented Approach*, 1st Edition, Irwin Publishing.
6. Gary J. Nutt, *Operating Systems: A Modern Perspective*, 2nd Edition, Addison-Wesley.
7. Maurice Bach, *Design of the Unix Operating Systems*, 8th Edition, Prentice-Hall of India.
8. Daniel P. Bovet, Marco Cesati, *Understanding the Linux Kernel*, 3rd Edition, O'Reilly and Associates.

Course: Discrete Mathematics

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD205**
Course Title	**Discrete Mathematics**
Credits	**4**
L-T-P (Contact Hours)	**3-1-0 (L:3H - T:1H - P:0H)**
Prerequisites	**CSD101**
Category	**Major Core**

Course Summary:

Throughout the course, students will be expected to demonstrate their understanding of Discrete Mathematics by being able to use mathematically correct terminology and notation, construct correct direct and indirect proofs, use division into cases in proof, use counterexamples and apply logical reasoning to solve a variety of problems

Learning Outcomes: On successful completion of the course, students will be able to achieve the following:
1. For a given logic sentence express it in terms of predicates, quantifiers, and logical connectives.
2. For a given 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:
1. 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.

2. 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,
3. 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.
4. Partial order sets: Definition, Partial order sets, Combination of partial order sets, Hasse diagram.
5. 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 and References:

1. C.L. Liu and Mohapatra, *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.
4. Kenneth Rosen, *Discrete Mathematics and its applications*, 7th edition, McGraw Hill Education.
5. J.P. Tremblay and R. Manohar, *Discrete Mathematical Structure and Its Applications to Computer Science*, 1st Edition, TataMcgraw-Hill.
6. Norman L. Biggs, *Discrete Mathematics*, 2nd Edition, Oxford University Press.
7. Seymour Lipschutz, Marc Lipson, *Schaum's Outlines Series of linear Algebra*, 3rd edition, McGraw Hill Education, 2017.
8. Kenneth Rosen,*Discrete Mathematics and its Applications*, 7th Edition, Tata McGraw Hill.

Course: Introduction to Probability and Statistics

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD210**
Course Title	**Introduction to Probability and Statistics**
Credits	**4**
L-T-P (Contact Hours)	**3-1-0 (L:3H - T:1H - P:0H)**
Prerequisites	**CSD102/201**
Category	**Major Core**

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 achieve the following:

1. Have a better understanding of probabilistic systems, such as reliability, performance-related issues by assigning an appropriate probability distribution
2. Some elementary understanding of the 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:
1. Axioms of probability, Conditional probability and independence, Bayes theorem.
2. Random variables, Distribution function, discrete random variable, Expectation, Variance, Bernoulli and Binomial random variable. Poisson random variable, Negative binomial random variable, Geometric random variable.
3. 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.
4. Generation of random numbers and elements of Monte Carlo simulation.
5. Elements of information theory: Entropy, Mutual information.

Textbooks and References:

1. Sheldon Ross, *A first course in probability*, 9th edition, Pearson Education India.
2. Kishor S. Trivedi, *Probability and Statistics with Reliability, Queuing, and Computer Science Applications*, 2nd edition, Wiley.
3. Robertazzi, *Computer Networksand Systems: Queuing Theory and Performance Evaluation*, 3rd edition, Springer.

Course: Computer Organization and Architecture

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD211**
Course Title	**Computer Organization and Architecture**
Credits	**5**
L-T-P (Contact Hours)	**3-1-1 (L:3H - T:1H - P:2H)**
Prerequisites	**CSD101**
Category	**Major Core**

Course Summary:
This course includes the working of Computer Systems, Instruction Level Architecture, Instruction Execution, current state of the art in memory system design, and I/O devices. It also includes the concept of microprogramming, parallel architecture and pipelining techniques.

Learning Outcomes: On successful completion of the course, students will be able to achieve the following:
1. Draw the functional block diagram of a single bus architecture of a computer and describe the function of the instruction execution cycle, RTL interpretation of instructions, addressing modes, instruction set.
2. Write an assembly language program for specified microprocessor for computing 16-bit multiplication, division, and I/O device interface (ADC, Control circuit, serial port communication).

3. Draw a flowchart for Concurrent access to memory and cache coherency in Parallel Processors and describe the process.
4. Design a memory and analyse its operation by interfacing with the CPU by given CPU organization and instruction.

Curriculum Content:

1. Functional blocks of a computer: CPU, memory, input-output subsystems, control unit.
2. Instruction set architecture of a CPU–register, instruction execution cycle, RTL
3. Interpretation of instructions, addressing modes, instruction set. Case study – instruction sets of some common CPUs.
4. Data representation: signed number representation, fixed and floating-point representations, character representation. Computer arithmetic – integer addition and subtraction, ripple carry adder, carry look-ahead adder, etc. multiplication – shift-and add, Booth multiplier, carry-save multiplier, etc. Division restoring and non-restoring techniques, floating-point arithmetic.
5. Introduction to x86 architecture. CPU control unit design: hardwired and micro-programmed design approaches, Case study – design of a simple hypothetical CPU. Memory system design: semiconductor memory technologies, memory organization. Peripheral devices and their characteristics: Input-output subsystems, I/O device interface, I/O transfers–program-controlled, interrupt-driven and DMA, privileged and non-privileged instructions, software interrupts and exceptions. Programs and processes–the role of interrupts in-process state transitions, I/O device interfaces – SCII, USB
6. Pipelining: Basic concepts of pipelining, throughput and speedup, pipeline hazards. Parallel Processors: Introduction to parallel processors, Concurrent access to memory and cache coherency.
7. Memory organization: Memory interleaving, the concept of hierarchical memory organization, cache memory, cache size vs. block size, mapping functions, replacement algorithms, write policies.

Textbooks and References:

1. David A. Patterson and John L. Hennessy, *Computer Organization and Design: The Hardware/Software Interface*, 5th Edition, Elsevier.
2. CarlHamacher, *Computer Organization and Embedded Systems*, 6th Edition, McGraw Hill Higher Education.
3. John P. Hayes, *Computer Architecture and Organization*, 3rd Edition, WCB McGraw-Hill.
4. William Stallings, *Computer Organization and Architecture: Designing for Performance*, 10th Edition, Pearson Education.
5. Vincent P. Heuring and Harry F. Jordan, *Computer System Design and Architecture*, 2nd Edition, Pearson Education.

Course: Object Oriented Programming

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD213**
Course Title	**Object Oriented Programming**
Credits	**4**
L-T-P (Contact Hours)	**3-0-1 (L:3H - T:0H - P:2H)**
Prerequisites	**CSD101**
Category	**Major Core**

Course Summary:

This course includes the introductory and advanced concepts and implementation of the Object Oriented Paradigm using any 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, etc.

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

1. Ability to understand, write and execute object oriented programs using various input and output mechanisms.
2. Understand and apply string-handling mechanisms.
3. Understand object oriented programming concepts.
4. Understand and use abstract classes and interfaces.
5. Understand use Inheritance and Polymorphism.
6. Understand and create Graphical user interface.
7. Understand and perform event driven programming.
8. Understand and apply exception handling.
9. Understand and apply database programming.

Curriculum Content:

Introduction to Computers, Programming, Elementary Programming, Selections, Loops, Methods, Single-Dimensional Arrays, Multidimensional Arrays, String handling, Object oriented programming: Classes, Objects, Inheritance and Polymorphism, Abstract Classes and Interfaces, GUI, Event-Driven Programming, Exception Handling, collection Framework: Lists, Stacks, Queues, and Priority Queues

Textbooks and References:

1. Y. Daniel Liang, *Introduction to Java Programming*, , Comprehensive version, 7th edition.
2. Herbert Schildt, *The Complete Reference*, 7th edition.

Course: Computer Networks

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD304**
Course Title	**Computer Networks**
Credits	**4**
L-T-P (Contact Hours)	**3-0-1 (L:3H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Core**

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 achieve the following:

1. Explain the functions of the different layers of the OSI Protocol.
2. Draw the functional block diagram of wide-area networks (WANs), local area networks (LANs) and Wireless LANs (WLANs) to 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 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:

1. 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
2. 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
3. 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.
4. Network Layer: Network layer design issues. Routing algorithms, Flooding, Shortest path routing, Link State routing, Hierarchical routing, Broadcast and multicast routings, Routing in the Internet, Path Vector routing, OSPF routing. The network layer on the Internet: IP protocol, ARP and RARP, BOOTP, ICMP, DHCP, Network Address Translation(NAT) Internetworking
5. 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
6. Application Layer: DNS-The Domain Name System, Electronic Mail, HTTP, FTP, Simple network management protocol (SNMP), The World Wide Web
7. 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.
8. Security: Introduction, Cryptography and Cryptanalysis, Public Key Cryptography Algorithms, RSA Algorithm, DES, Authentication and Authorization.

Textbooks and References:

1. Andrew S Tanenbaum, *Computer Networks*,PHI, 2010.
2. William Stallings, *Data and Computer Communications*, PHI, 2002.
3. Fred Halsall, *Data Communications, Computer networking on OSI*, Addison Wesley Publishing Co, 1998.
4. James F. Kurose and Keith W. Ross,*Computer Networking -A Top-Down Approach Featuring the Internet*, Addison Wesley Publishing Co., 2004.
5. Uyless Black, *Computer Networks: Protocols standards and interfaces*,Prentice-Hall, 2002.
6. Behrouz A. Forouzan, *Data communication & Networks*,Tata McGraw Hill. 2002.

Course: Artificial Intelligence

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD311**
Course Title	**Artificial Intelligence**
Credits	**4**
L-T-P (Contact Hours)	**3-0-1 (L:3H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Core**

Course Summary:
This course introduces the concepts and techniques in the field of artificial intelligence. It is aimed for undergraduate students who have knowledge of Data structures and any imperative programming language such as C, C++, Java, etc. AI is a broad area consisting of various courses under its umbrella such as Neural Network, Soft Computing, Machine Learning, Natural Language Processing, Vision, etc., But this course imparts broad overview, both of traditional and modern AI, and prepares a student for advanced elective courses as mentioned above.

Learning Outcomes: On successful completion of the course, students will be able to achieve the following:
1. Use language Prolog (Programming in Logic) to develop software systems
2. Develop programs for 2-player games
3. Develop production system and expert system
4. Develop Neural Network and Fuzzy systems

Curriculum Content:
1. Introduction**:** Overview of AI, AI problems and characteristics, Problem-solving: Production systems, control strategies, Reasoning: forward & backward chaining exhaustive search strategies (Breadth-first search, Depth-first search, Iterative deepening, Uniform cost search)
2. Intelligent searching: Best first search, A* algorithm, AO* algorithm, Measures of performance
3. Game playing: Minimax & game trees, Refining Minimax, Alpha-Beta pruning
4. Knowledge Representation: First order predicate Calculus Resolution, Unification, Natural deduction system, Resolution Refutation, Logic Programming, PROLOG, Semantic Networks, Frame System, Value inheritance, Conceptual Dependency.
5. Advanced Problem solving using Planning, Handling uncertainty: using probabilistic models and Fuzzy Logic.
6. Machine learning: Inductive and deductive learning, Artificial Neutral Networks, Support Vector Machines, Expert Systems and Applications of Expert System.
7. Soft computing: genetic algorithms, swarm intelligence.
8. Intelligent agents: Classification of agents, architecture and multi-agent system design and applications, Natural language Processing

Textbooks and References:

1. Saroj Kaushik, *Artificial Intelligence*,1st Edition, Cengage Learning, 2019 (reprint)
2. Stuart Russell & Peter Norvig, *Artificial Intelligence: A Modern Approach*, 3rd Edition, Prentice-Hall, 2009.
3. Tom Mitchell., *Machine Learning*, 1st Edition, McGraw-Hill, 1997

Course: Introduction to Database Systems

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD317**
Course Title	**Introduction to Database Systems**
Credits	**4**
L-T-P (Contact Hours)	**3-0-1 (L:3H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Core**

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. It includes an 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 and design indexes based on application requirements
7. Understand query processing & optimization, while taking trade-offs into consideration
8. Understand transaction processing, concurrency control & recovery
9. Write programs in SQL

Curriculum Content:
1. Introduction to Databases and Transactions: database system definition, the purpose of the database system, view of data, relational databases, database architecture, and transaction management.
2. Data Models: The importance of data models, Basic building blocks, Business rules, The evolution of data models, Degrees of data abstraction.
3. Database Design: ER-Diagram and Unified Modelling 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).

4. Relational Algebra: Introduction, Selection and projection, set operations, renaming, Joins, Division, syntax, semantics. Operators, grouping and ungrouping, relational comparison.
5. Constraints, Views, and SQL: 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 subqueries, Joined relations. Triggers.
6. 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.

Textbooks and References:

1. Ramez Elmasri and Shamkant B. Navathe, *Fundamentals of Database Systems*, 7th Edition, Pearson,2017.
2. Avi Silberschatz, Henry F. Korth, S. Sudarshan, *Database System Concepts*, 7th Edition, McGraw-Hill.

Course: Design and Analysis of Algorithms

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD319**
Course Title	**Design and Analysis of Algorithms**
Credits	**4**
L-T-P (Contact Hours)	**3-0-1 (L:3H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Core**

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 achieve the following:

1. Demonstrate knowledge of how to measure the complexity of an algorithm, including best-case, worst-case, and average complexities as functions of the input size, as well as classification in terms of asymptotic complexity classes.
2. Recognize different algorithmic design strategies which include recursion, divide-and-conquer, the greedy method, dynamic programming, and backtracking and branch-and-bound, etc.
3. Compare different algorithm design strategies for any computational problem.
4. Understand the concepts of NP-Completeness.

Curriculum Content:

1. Introduction- Fundamental characteristics of an algorithm. Basic algorithm analysis –Asymptotic analysis of complexity bounds – best, average and worst-case behavior, 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.
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.

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

Textbooks and References:

1. Jon Kleinberg and Eva Tardos, *Algorithm Design*, 1st Edition, Pearson Education India.
2. T.H. Corman et. al., *Introduction to Algorithms*, 3rd Edition, PHI Learning Pvt. Ltd.
3. E. Horowitz et al.,*Fundamentals of Algorithms*, Universities Press.
4. C.H. Papadimitriou et al., *Combinatorial Optimization: Algorithms and Complexity*, Dover Publications, 1998.

Course: Software Engineering

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD326**
Course Title	**Software Engineering**
Credits	**4**
L-T-P (Contact Hours)	**3-0-1 (L:3H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Core**

Course Summary:

Software engineering is the branch of computer science that creates practical, cost-effective solutions to computing and information processing problems, preferentially by applying scientific knowledge, developing software systems in the service of mankind. This course covers the fundamentals of software engineering, including understanding system requirements, finding appropriate engineering compromises, effective methods of design, coding, and testing, team software development, and the application of engineering tools. The course will combine a strong technical focus with a capstone project providing the opportunity to practice engineering knowledge, skills, and practices in a realistic development.

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

1. Understand various phases of the software development lifecycle.
2. Analyze the requirements systematically and develop the model using standard tools and techniques.
3. Apply key aspects of software engineering processes for the development of a complex software system.
4. Develop a quality software project through effective team-building, planning, scheduling, and risk assessment.
5. Keep abreast of current trends in the area of software engineering

Curriculum Content:

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

2. Process Models and Program Design Techniques- Software Development Process Models – Code & Fix model, Waterfall model, Incremental model, Rapid Prototyping model, 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.
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.
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.
5. Advanced Topics: Formal Methods in Software Engineering – Z notation, Hoare's notation. Formalization of Functional Specifications – SPEC. Support environment for the 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, Petri nets. Introduction to Design Patterns, Aspect-oriented Programming.

Textbooks and References:

1. Carlo Ghezzi, *Fundamentals of Software Engineering*, 2nd Edition, PHI, 2002.
2. Ian Sommerville, *Software Engineering*, 9th Edition, Pearson, 2011.
3. Berzins and Luqi, *Software Engineering with Abstraction*, 1st Edition, Addison-Wesley, 1991.
4. Martin L. Shooman, *Software Engineering – Design, Reliability and Management*, McGraw-Hill Education, 1984.

Course: Theory of Computation

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD334**
Course Title	**Theory of Computation**
Credits	**3**
L-T-P (Contact Hours)	**3-0-0 (L:3H - T:0H - P:0H)**
Prerequisites	**CSD102/201**
Category	**Major Core**

Course Summary:

The theory of computation teaches how efficiently problems can be solved on a model of computation, using an algorithm. It is also necessary to learn the ways in which the computer can be made to think. Finite state machines can help in natural language processing which is an emerging area.

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

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 and References:

1. M. Sipser, *Theory of Computation*, 3rd Edition, Cengage.
2. Hopcroft Motwani and Ullman, *Introduction to Automata Theory, Languages and Computation*, Pearson.
3. Peter Linz, *Introduction to formal languages and automata*, Jones & Bartlett.
4. Harry R. Lewis and Christos H. Papadimitriou, *Elements of the Theory of Computation*, 1st edition, Pearson Education Asia.
5. Dexter C. Kozen, *Automata and Computability, Undergraduate Texts in Computer Science,* Springer, 2007.
6. Michael Sipser, *Introduction to the Theory of Computation*, 3rd edition, Cengage Learning.
7. John Martin, *Introduction to Languages and The Theory of Computation*, 3rd edition, Tata McGraw Hill.

Course: Software Design Lab

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD345**
Course Title	**Software Design Lab**
Credits	**2**
L-T-P (Contact Hours)	**0-0-2 (L:0H - T:0H - P:4H)**
Prerequisites	**CSD102/201**
Category	**Major Core**

Course Summary:
Students will learn to develop the large programs such as editor, parser, Lex and Yacc etc.

Course: Seminar

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD346**
Course Title	**Seminar**
Credits	**2**
L-T-P (Contact Hours)	**0-0-2 (L:0H - T:0H - P:4H)**

Prerequisites	**CSD102/201**
Category	**Major Core**

Course Summary:
Students will learn to read, understand and present latest research papers in their chosen area of interest.

Elective Course Descriptions

Course: Natural Language Processing

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD350**
Course Title	**Natural Language Processing**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
Natural Language Processing (NLP) is a core area for advancement of Artificial Intelligence Systems and Humanoids, so that these systems can converse like humans. The course introduces fundamental concepts and techniques of natural language processing. Students will gain an in-depth understanding of the computational properties and algorithms for processing linguistic information. NLP has various industry applications like semantic search engines, conversational engines (AI chatbots, virtual agents and humanoids), document summarization systems, knowledge generation systems, speech recognition, text generation and language translators. Students will get an exposure to NLP requirements of Artificial Intelligence industry.

Learning Outcomes: On successful completion of the course, students will be able to achieve the following:
1. Have a thorough understanding of NLP theory and algorithms
2. Develop NLP based AI systems
3. Have a good exposure to industry applications and requirements for NLP

Curriculum Content:
1. Regular Expressions, Text Normalization, Edit Distance, N-gram Language Models, Naïve Byes and Sentiment Classification, Logistic Regression, Vector Semantics and Embeddings,
2. Neural Networks and Neural Language Models, Part-of-Speech Tagging, Sequence Processing with Recurrent Networks (Simple RNN, Applications, Deep Networks stacked and bidirectional, Managing context in RNN: LSTM and GRUs, Words, Subwords and Characters),
3. Encoder-Decoder Models, Attention and Contextual Embeddings (Neural Language Models and Generation, Encode-Decoder networks, Attention, Transformer networks),
4. Machine Translation, Constituency Grammars (Constituency, Context Free Grammars, Treebanks, Equivalence and Normal Form, Lexicalized Grammars), Constituency Parsing (Ambiguity, CKY Parsing, Partial, Chunking), Statistical Constituency Parsing (Probabilistic CFGs, Evaluating Parsers), Dependency Parsing (Transition based, Graph based),

5. Logical Representation of Sentence Meaning (Model-Theoretic, Event and State), Computational Semantics and Semantic Parsing, Information Extraction, Word Sense and WordNet (Disambiguation, algorithms, thesauruses, induction),
6. Semantic Role Labelling (Proposition Bank, FrameNet, Selectional Restrictions, Primitive decomposition of predicates), Lexicon for Sentiment, Affect and Connotation (Emotion, Human Labelling, semi-supervised induction, personality, connotation frames),
7. Coreference resolution (Tasks and datasets, mention detection, algorithms, neural mention ranking, entity linking, Winograd schema, gender bias), Discourse Coherence (Relations, parsing, centering and entity based, learning models, global coherence),
8. Summarization, Question Answering (IR based factoid, knowledge based), Dialogue systems and chatbots (rule based, corpus based, slot filling, dialogue state tracking, natural language generation), Phonetics (Speech sounds, transcription, articulatory, prosodic prominence, structure and tune, acoustic phonetics and signals), Speech Recognition and Synthesis

Textbooks and References:

1. Bernstein and Newcomer, *Principles of Transaction processing*, Morgan and Kaufmann.
2. Ramez Elmasri and Shamkant Navathe, *Fundamentals of Database Systems*, 6th edition, Addison-Wesley Publishing Company, USA, 2010.
3. Abraham Silberschatz, Henry F. Korth, and S. Sudarshan, *Database Systems Concepts* 6th edition, McGraw-Hill Higher Education, 2010.
4. Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom, *Database Systems: The Complete Book*, 2nd edition, Prentice-Hall, 2008.
5. Jim Grey, *On Database Operating System and Transaction Execution: Database operating Systems*, Springer – Verlag, 1979.

Course: Advanced Database Management System

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD351**
Course Title	**Advanced Database Management System**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD317/202**
Category	**Major Elective**

Course Summary:

The course discusses the system level issues, serializability, concurrency control, transaction management, and recovery. It addresses the issue of a database implementation, query processing and query optimization for relational databases. Different file structures, indexing, and hashing techniques will also be addressed. The course will also introduce the management of Big Data and data warehouse.

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

1. The architecture and internals of the Relational DBMS system.

2. Concurrency control, its limitations, and algorithms to deal with the concurrent execution of transactions.
3. The failures that can occur during concurrent execution of transactions and different algorithms to recover from failures.
4. Query processing and optimization, performance, and cost of execution, and able to write optimized queries.
5. Fundamental concepts of data warehousing and OLAP techniques, Index Management.
6. The four dimensions of big data, challenges involved and its applications.

Curriculum Content:

1. Introduction to database Concepts, Serializability**:** Concurrency Control Problems, Serializable Executions, Consistency Preservation, Ordering Transactions, Limitations of Serializability.
2. Recoverability: Rollback, Roll forward, Recoverable Histories, Avoiding Cascading Aborts, Strict Executions, The Recovery Manager, Schedulers.
3. Concurrency Control: Two phase locking, Deadlocks, Multi-granularity locking, Non Locking Schedulers: Timestamp ordering, Serializable graph testing.
4. Centralized Recovery: Failures, The recovery Manager, The Undo/redo algorithm, The Undo/no-redo algorithm, The no-undo/redo algorithm, The no-undo/no-redo algorithm.
5. Query Optimization: Algorithms for select, join, project, outer joins, aggregate and set operations, Combining operations using pipelining, Selectivity and Cost estimation.
6. Data Warehousing: What is Data Warehousing, dimensional modelling, slowly changing dimensions, fact tables, OLAP objects, business intelligence, and extract, transform, and load technologies.
7. View Maintenance: Materialized views and their applications, What is view maintenance
8. File Management: Placing file record on disk, Operation on Files, Files of unordered records, Files of ordered record, Single-level indexes, Multilevel Indexes, B-Tree, B+ Tree.
9. Big Data:Four Dimensions of Big Data, Why big data is fast and noisy, Applications in Different Domains.

Textbooks and References:

1. Bernstein and Newcomer, *Principles of Transaction processing*, Morgan and Kaufmann.
2. Ramez Elmasri and Shamkant Navathe, *Fundamentals of Database Systems*, 6th edition, Addison-Wesley Publishing Company, USA, 2010.
3. Abraham Silberschatz, Henry F. Korth, and S. Sudarshan, *Database Systems Concepts* 6th edition, McGraw-Hill Higher Education, 2010.
4. Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom, *Database Systems: The Complete Book*, 2nd edition, Prentice-Hall, 2008.
5. Jim Grey, *On Database Operating System and Transaction Execution: Database operating Systems*, Springer – Verlag, 1979.

Course: Computational Neuroscience

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD352**
Course Title	**Computational Neuroscience**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
This course introduces basic computational methods for understanding how nervous systems function. Computational principles underlying various aspects of vision, sensory-motor control, learning and memory are studied. Topics such as goal directed behaviour, sleep and consciousness would be discussed. The course has implications in advancement of artificial intelligence, machine learning and robotics as these fields are inspired by how human brain works.
Learning Outcomes: On successful completion of the course, students will be able to:
1. Appreciate the complexity involved in computations within the brain
2. Understand how to develop models of neurons and their networks
3. Decide to pursue research in the subject for final year thesis work or higher studies.

Curriculum Content:
1. **Introduction to CNS and basic neurobiology:** Introduction to CNS, Neuroanatomy: Frontal, Parietal, Temporal, Occipital, Insula and Limbic. Neurophysiology: Sensory Systems, Sensorimotor Control, Neural plasticity, Motivational Systems, Memory Systems.

2. **Building Neural Models:** Integrate-and-Fire model, Hodgkin-Huxley model, Compartmental models, Hebbian plasticity, Neural encoding and decoding, Population models.

3. **Project:** Students to undertake a mini project involving literature survey and building models using tools like MATLAB and NEURON.

Course: Computer Graphics

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD353**
Course Title	**Computer Graphics**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Curriculum Content:
1. Introduction, CG system, Recursive Fractals, Geometric Objects, Affine Transformations - Translation, Rotation, Scaling, Homogeneous Coordinates, Concatenation.
2. OpenGL Transformations, Projection, Parallel, Perspective, extended Homogenous, Viewing Volumes, Frame Transformations, Clipping, View-Port transformation, Stereo Viewing, Artistic Projection, Non linear projection, Introduction to OpenGL and GLUT.
3. Modelling curves and surfaces, Parametric polynomial curves, Bezier curves, Hermite curves, Splines, B-spline subdivisions schemes, Tensor product surfaces, Surface of revolution, Polygonal meshes. 3D formats: obj and md2, Texture coordinates, Half edge data structures, Back/front faces, hidden line removal using depth buffer.
4. Rendering faces: Gouraud and Phong shading, Ray tracing, Ray casting, Recursive ray-tracing, Ray mesh intersection, Bounding objects, Scene description, Anti-Aliasing, Distributed ray tracing.

Textbooks and References:
Interactive Computer Graphics: A Top-Down Approach with OpenGL, 2nd Edition, Edward Angel

Course: Foundation of Data Sciences

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD355**
Course Title	**Foundation of Data Sciences**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD317/202**
Category	**Major Elective**

Course Summary:
Course topics will cover data collection, cleaning and visualization. Data modeling and basics of databases. Mathematical foundations of data science including linear algebra, (multivariate) calculus and convex optimization. Topics in data mining, such as similarity and distance functions, clustering, ranking, networks. Introduction to machine learning. Prediction methods, e.g. regression and common measures.

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

1. Understand the key concepts in data science, including their real-world applications
2. Understand how data is collected, managed and stored for data science
3. Implement and understand statistics and machine learning concepts that are vital for data science
4. Critically evaluate data visualisations based on their design and generate visualisations from data

Curriculum Content:
1. Data handling in Python: collecting, cleaning visualizing using Python tools.
2. SQL and data modelling: relational algebra, schemes, indexing basics
3. Similarity and distance functions, clustering
4. Linear algebra, dimensionality reduction: SVD, least squares
5. Probability and statistics: interpreting results
6. Convex optimization: gradient (one and multi dimensional), gradient descent, regressions
7. Machine Learning basics: linear and ridge regression, SVM
8. Data Visualization: Basic principles, ideas and tools for data visualization.
9. Graphs: node importance, connectivity, centrality – Page rank, social and web graphs, community detection

Textbooks and References:

1. Cathy O'Neil and Rachel Schutt, “Doing Data Science, Straight Talk From The Frontline”, O'Reilly, 2014.
2. Jiawei Han, Micheline Kamber and Jian Pei, “Data Mining: Concepts and Techniques”, Third Edition. ISBN 0123814790, 2011.
3. Mohammed J. Zaki and Wagner Miera Jr, “Data Mining and Analysis: Fundamental Concepts and Algorithms”, Cambridge University Press, 2014.

4. Matt Harrison, “Learning the Pandas Library: Python Tools for Data Munging, Analysis, and Visualization, O'Reilly, 2016.
5. Joel Grus, “Data Science from Scratch: First Principles with Python”, O’Reilly Media, 2015.
6. Wes McKinney, “Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython”, O'Reilly Media, 2012.

Course: Foundation of Information Security

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD356**
Course Title	**Foundation of Information Security**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
This course will introduce students to fundamentals of information security, cryptography, access control mechanisms, system attacks and defenses against them.

Learning Outcomes:
1. Learn information security basics
2. Learn to use and apply various security mechanisms to real world problems

Curriculum Content:
1. Security Overview, CIA model, Threats, Policy and Mechanisms, Security Policies, Confidentiality Policies, Integrity Policies, Hybrid Policies, Cryptography Basics, Classical Cryptosystems, Stream Ciphers and Block Ciphers, Public Key Cryptography: RSA
2. Cryptographic Checksums , Authentication Basics, Password management, Challenge Response, Biometrics, Key Exchange, Certificate Chains, X.509, Digital Signatures, Access Control Lists: Creation and Maintenance, Revocation of Rights, Ring based Access Control
3. Stream Ciphers, LFSR based stream ciphers, Message Authentication Codes, Hash functions, Hash algorithms, Digital Signatures and Authentication Protocols, Firewalls, Malicious Logic, Trojan Horses, Viruses, Worms, Logic Bombs, Defenses, Sandboxing, Intrusion Detection: Principles and Basics, Anomaly modelling, Architecture: Host and network based Information Gathering, Organization of Intrusion Detection Systems, Intrusion Response
4. Firewalls and Proxies, DMZ server, User Security: Policy, Access, Files and Devices, Processes, Electronic Communications, Program Security: Requirements and Policy, Design, common security related programming problems, Virtual Machines Structure

Textbooks and References:

7. Matt Bishop, S.S. Venkatramanayya, *Introduction to Computer Security*, 3/e, Pearson Education
8. W Stallings, *Cryptography and Network Security: Principles and Practice*, 6/e, Prentice Hall
9. B. Forouzan, D. Mukhopadhyay, *Cryptography and Network Security* 2/e, Tata-McGraw Hill

Course: Image Processing and Its Applications

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD357**
Course Title	**Image Processing and Its Applications**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
Fundamentals of digital image processing, image enhancement using point processing, edge detection, noise removal, line detection, corner detection, morphological operations on binary images, texture determination, video processing and motion estimation, image processing in frequency domain, filtering, image compression, DCT, JPEG, object detection and classification, digit recognition, face recognition using Machine Learning and Deep Learning,

Learning Outcomes: On successful completion of the course, students will be able to:
1. Apply Hough transforms and be familiar with image representation using textures
2. Compute Motion using optic flow, and understand methods for image description and morphological operations
3. Analyze images in frequency domain using various transforms
4. Demonstrate knowledge of various compression techniques
5. Using Deep learning for object detection and classification

Curriculum Content:
1. Image Enhancement: Contrast stretching using point processing, Histogram Equalization. Enhancement using Spatial filters, concept of convolution, smoothing, Gaussian filters, Edge detection using Prewitt, Sobel, Laplace Filters. Canny Edge Detector, Harris Corner Detector.
2. Lines and Texture: Line detection using Hough transform, polar form, Circle Detection, Texture analysis, Texture from histogram, Texture from GLCM matrices.
3. Morphology: Morphological Operations – Dilation, Erosion, Opening, Closing, Boundary detection, Hole filling, Hit and Miss transform. Shape representation using moments.
4. Motion: Video Processing, Motion Detection: Concept of Optical Flow, optical flow equation, Lucas Kanade method
5. Frequency Domain transformation: 2-D Fourier Transform, Low pass and Hi pass Filtering, Gaussian filters, Homomorphic Filtering, Image Compression: Run length Encoding, Huffman Coding, DCT, zigzag coding, JPEG, MPEG.
6. Feature Detection: Text recognition, Face Detection – Viola Jones method, Face Recognition using Eigenface, PCA, SIFT and HOG parameters
7. Applications: Using Machine Learning and Deep learning for plant disease identification, human activity recognition

Textbooks and References:

1. Gonzales, R. and R. E. Woods, Digital Image Processing, 4th ed. Pearson
2. Sridhar, S. Digital Image Processing, 2nd ed. Oxford University Press

---

Course: Information Retrieval

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD358**
Course Title	**Information Retrieval**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
This is an undergraduate-level introductory course for information retrieval. It will cover algorithms, design, and implementation of modern information retrieval systems. Topic includes retrieval system design and implementation, text analysis techniques, retrieval models (e.g., Boolean, vector space, probabilistic, and learning-based methods), search evaluation, retrieval feedback, search log mining, and applications in web information management.

Learning Outcomes: On successful completion of the course, students will be able to achieve the following:
1. Information Retrieval process and impact of the web on IR.
2. Different information retrieval models
3. Performance metrics to evaluate any IR model
4. Query languages used for IR
5. Knowledge of text mining techniques
6. Understand the working of web search engines
7. Implement their own search engine

Curriculum Content:
1. Introduction: Goals and history of IR. The impact of the web on IR.
2. Basic IR Models: Boolean and vector-space retrieval models; ranked retrieval; text-similarity metrics; TF-IDF (term frequency/inverse document frequency) weighting; cosine similarity.
3. Basic Tokenizing, Indexing, and Implementation of Vector-Space Retrieval: Simple tokenizing, stop-word removal, and stemming; inverted indices; efficient processing with sparse vectors;
4. Experimental Evaluation of IR: Performance metrics: recall, precision, and F-measure; Evaluations on benchmark text collections.
5. Query Operations and Languages: Relevance feedback; Query expansion; Query languages.
6. Text Classification and Cluster: K-Nearest Neighbour classification, Naïve Bayes Classifier, K-Means Clustering, HAC clustering.
7. Web Search: Search engines; spidering; metacrawlers; directed spidering; link analysis (e.g. Google PageRank).

Textbooks and References:

1. C. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval,* Cambridge University Press, 2008.
2. Ricardo Baeza -Yates and Berthier Ribeiro – Neto, *Modern Information Retrieval: The Concepts and Technology behind Search*, ACM Press Books.
3. Bruce Croft, Donald Metzler and Trevor Strohman, *Search Engines: Information Retrieval in Practice*, Pearson.

Course: Introduction to Logic and Functional Programming

School	**School of Engineering**
Department	**CSE**
Course Code	**CSD360**
Course Title	**Introduction to Logic and Functional Programming**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
The course introduces declarative/applicative style of computing. Declarative programming is about describing what to achieve without instructing how to do it. In this category there are mainly two computing paradigms. One is based on resolution and the other on reduction. Logic programming is based on resolution and Functional programming is based on reduction. This course discusses mathematical foundations of these paradigms along with logic language called Prolog and functional language SML.

Learning Outcomes: On successful completion of the course, students will be able to:
1. Learn and appreciate the declarative style of computing which is most suitable for building the structure and elements of computer programs and allows to express the logic of a computation without describing its control flow in contrast with imperative programming where actual flow of algorithm is stated and implemented.
2. Learn to model, or mathematical representations of physical systems which may be implemented in declarative languages.
3. Learn and code in Prolog (Programming in Logic) and SML functional languages.

Curriculum Content:
1. **Introduction of computing paradigms:** Imperative verses declarative computing; Introduction of Logic and Functional Paradigm
2. **Propositional Logic:** Propositional Concepts; Natural Deduction and Axiomatic system; Semantic Tableaux and Resolution
3. **First Order Predicate Logic(FOL):** Predicate Calculus; Prenex normal forms and Skolemization; Clauses in FOL; Semantic Tableaux and Resolution
4. **Logic programming:** Conversion of Clauses to Clausal representation; Interpretation of Logic program (LP); Execution of a Query in Logic Program; Abstract interpreter for LP
5. **Prolog Programming:** Programming in Prolog (Overview); Meta Level Programming and Meta interpreters; Nondeterministic Programming
6. **Functional Programming (FP) Concepts:** Functions; Mathematical notion of function; Multi-argument functions; Expression composition & equality; Recursive Definitions; Higher Order Functions; Functions as data objects; Curried Functions
7. **SML, a functional language:** Introduction to SML**;** Value and Function Declaration; Bindings and Environments; Polymorphic Function Declarations; Records and Tuples; Local declarations; List and Advanced

Features in SML**;** Manipulation of List; Tree manipulation in SML; Graphs as an Application of a List; Structures declaration; Recursive Datatype Declarations; Exception Handling

8. **Lambda Calculus:** Pure Lambda Calculus; Currying of function (λ - function with more arguments); Applied Lambda Calculus; Function definition using λ-notation; Recursive Definitions in λ - Notation
9. **Lazy and Eager Evaluation:** Evaluation Strategies; Lazy Evaluation; Evaluation Order and strictness of function; Programming with lazy evaluation; Interactive functional program; Delay of unnecessary computation; Eager Evaluation and Reasoning

Textbooks and References:

1. Saroj Kaushik, *Logic and Prolog Programming*, New Age International 2002.
2. J. W. Lloyd. *Foundations of Logic programming*. Springer-Verlag. New York.
3. Laurence C. Paulson. *ML for the Working Programmer*. Cambridge University Press.
4. Chris Reade. *Elements of Functional Programming*. Addison-Wesley.
5. John Kelly, *The Essence of Logic, Prentice Hall of India*, 1997.
6. Anil Nerode and Richard A. Shore. *Logic for Applications*. Springer-Verlag.
7. Leon Sterling and Ehud Shapiro. *The Art of Prolog (Advanced Programming Techniques)*, Prentice Hall of India, 1996.
8. Peter Henderson. *Functional Programming: Applications and Implementation*. Prentice Hall.

Course: Introduction to Machine Learning

School	**School of Engineering**
Department	**CSE**
Course Code	**CSD361**
Course Title	**Introduction to Machine Learning**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD102/201, CSD210/209**
Category	**Major Elective**

Course Summary:
The course introduces the basic concepts, techniques and tools for designing programs that learn from data.

Learning Outcomes: On successful completion of the course, students will be able to:
1. Build models for prediction and data organization from data.
2. Learn to use basic ML libraries.
3. Understand the basic theories and concepts that underly machine learning.

Curriculum Content:
The learning problem, Types of learning, Training, validation, testing, generalization, overfitting, Features and feature engineering, dimensionality reduction, Bayesian decision theory, Parametric methods, Tree models, Linear models, SVMs and kernel based models, Nearest neighbour models, Markov models, Neural network models, Ensemble methods - boosting, bagging, voting schemes, Distance metrics and cluster based models.

The topics in the course will not be covered in linear order. They will be inter-twined to make machine learning easy to understand and hopefully the progression will be fairly logical.

Textbooks and References:

1. Ethem Alpaydin, *Introduction to Machine Learning*, 3rd Ed., MIT Press, 2014.
2. Peter Flach, *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*, CUP, 2012.
3. Kevin Murphy, *Machine Learning: A Probabilistic Perspective*, MIT Press, 2012.
4. S Kulkarni, G Harman, *An Elementary Introduction to Statistical Learning Theory*, Wiley, 2011.

Course: Applied Cryptography

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD451**
Course Title	**Applied Cryptography**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD101, CSD205**
Category	**Major Elective**

Course Summary:

This course will introduce students to basic building blocks of cryptography and applications of cryptographic protocols in real world. The focus will be on how cryptography and its application can maintain privacy and security in electronic communications and computer networks.

Learning Outcomes:

1. Learn applied cryptographic basics
2. Learn to apply and use cryptographic concepts to real world problems

Curriculum Content:

1. Course Introduction and terminology, Conventional Cryptography: Definitions, Classical encryption techniques, One-time pad, Perfect Secrecy, DES, Triple DES, Finite fields, AES, Modes of Encryption
2. Asymmetric Cryptography: Number Theory, public key cryptography: RSA, ElGamal, and Elliptic Curve Cryptography, Diffie Hellman Key management, Digital Certificates: X.509
3. Stream Ciphers, LFSR based stream ciphers, Message Authentication Codes, Hash functions, Hash algorithms, Digital Signatures and Authentication Protocols, Firewalls
4. Intrusion Detection, PGP, S/MIME, Kerberos, IPSec, SSL/TLS, Password Hashing and Management

Textbooks and References:

1. *Modern cryptography*: Jonathan Katz, Yehuda Lindell, 2nd Ed., CRC Press
2. W Stallings, *Cryptography and Network Security: Principles and Practice*, *6/e*, Prentice Hall
3. Douglas R. Stinson, *Cryptography: Theory and Practice 3/e*, CRC Press, 2006
4. Christof Paar, Jan Pelzl, *Understanding Cryptography: A textbook for students and practitioners*, *1/e*, Springer
5. A. Menezes, P. van Oorschot, S. Vanstone. *Handbook of Applied Cryptography*, CRC press, 1997.
6. B. Schneier. *Applied Cryptography*. Second Edition. John Wiley & Sons, Inc., 1996
7. B. Forouzan, D. Mukhopadhyay, *Cryptography and Network Security 2/e*, Tata-McGraw Hill
8. Bernard Menezes, *Network Security and Cryptography 2/e*, Cenege Learning

---

Course: Big Data Analytics

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD452**
Course Title	**Big Data Analytics**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD317/202**
Category	**Major Elective**

Course Summary:
This course is designed to equip students with knowledge about the fundamentals of concepts of big data analytics, and make them understand the various search methods and visualization techniques. The course will cover the various techniques for mining data stream and understand the applications using Map Reduce Concepts.

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

1. Work with big data platform and explore the big data analytics techniques.
2. Design efficient algorithms for mining the data from large volumes.
3. Analyze the HADOOP and Map Reduce technologies associated with big data analytics.
4. Understand the fundamentals of various big data analytics techniques.

Curriculum Content:

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

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

Hadoop: History of Hadoop - the Hadoop Distributed File System – Components of Hadoop Analysing the Data with Hadoop- Scaling Out- Hadoop Streaming- Design of HDFS-Java interfaces to HDFS Basics- Developing a Map Reduce Application-How Map Reduce Work.

Predictive Analytics: Simple linear regression- Multiple linear regression- Interpretation 5 of regression coefficients. Visualizations - Visual data analysis techniques- interaction techniques - Systems and applications.

Textbooks and References:

1. Michael Berthold, David J. Hand, "Intelligent Data Analysis", Springer, 2007.
2. Tom White "Hadoop: The Definitive Guide" Third Edition, O'reilly Media, 2012.
3. Chris Eaton, Dirk DeRoos, Tom Deutsch, George Lapis, Paul Zikopoulos, "Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data", McGrawHill Publishing, 2012.

4. Anand Rajaraman and Jeffrey David Ullman, “Mining of Massive Datasets”, CUP, 2012.

Course: Computer Vision

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD454**
Course Title	**Computer Vision**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
The goal of the Computer Vision course is to provide hands-on knowledge on applying popular Computer Vision techniques to handle images and videos. The students of this course will be given opportunities to do one research project and a set of assignments. The course curriculum is designed to equip the students with the recent advances in Computer Vision.

Learning Outcomes: This course will enable the students to acquire a deep understanding on the recent techniques for computer vision. On successful completion of the course, students will be able to achieve the following:

1. Understand the basic and traditional ideas of computer vision problems.
2. Implement different popular computer vision techniques.
3. Have a deep knowledge about the recent advances of Computer Vision based on deep learning techniques.
4. Solve real life problems on computer vision.

Curriculum Content:

1. Low-level vision: image processing, edge detection, feature detection, image matching, RANSAC, etc.
2. Geometry and Photometry: projective geometry, stereo, structure from motion, etc.
3. Recognition and Learning: Recognition, Machine Learning, ANN, CNN, etc., CNN and Computer Vision

Textbooks and References:
5. Rick Szeliski, Computer Vision: Algorithms and Applications
Online at: http://szeliski.org/Book/
6. Ian Goodfellow and Yoshua Bengio and Aaron Courville, Deep Learning, MIT Press, 2016
Online at: http://www.deeplearningbook.org/
7. Online course CS231n of Stanford University.

Course: Data Mining and Warehousing

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD455**
Course Title	**Data Mining and Warehousing**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD317/202, CSD210/209**
Category	**Major Elective**

Course Summary:
In this course, we would explore the fundamental data mining methodology, OLTP and OLAP, data pre-processing, association rules mining, clustering, classification, and other advanced topics in the field such as Social impact of Data mining, Recent trends in Data mining research, Challenges and Future Scope, need for Security and Privacy preserving in Data Mining.

Learning Outcomes: On successful completion of the course, students will be able to achieve the following:
1. Demonstrate a comprehensive understanding of different tasks associated in Data Warehousing and Data Mining.
2. Understand the concepts of Data Pre-processing in the Data Mining process.
3. Understand the concepts of Association rules mining.
4. Understand the concepts of Data Mining techniques, such as Clustering, Classification, etc.
5. Formulate and design solutions for real-world Data Mining applications and Implement the designed solutions in the most suitable programming language.
6. Evaluate and Appraise/Criticize the solution designed for any data mining task.

Curriculum Content:

1. Introduction to data mining – A bird's eye view, Data Mining issues, Social Implications Introduction to data warehousing, Database/OLTP systems, OLAP Data Pre-processing - Data Cleaning, Feature extraction.
2. Data Reduction and Transformation, Data Visualization, Mining Association Rules, From Association Mining to Correlation Analysis.
3. Classification – Feature selection for classification, Decision trees, Advanced Classification techniques.
4. Clustering – Cluster analysis, Advanced Clustering techniques.
5. Mining complex data, Text data mining, Mining Time-series data, Spatial data mining, Multimedia data mining, Mining the web.
6. Social impact of Data mining, Recent trends in Data mining research, Challenges and Future Scope, Security and Privacy preserving in Data Mining, Review.

Textbooks and References:

1. J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 3rd Edition, Elsevier.
2. J. Leskovec, A. Rajaraman, *Mining Massive Datasets*, 2nd Edition, Dreamtech Press.
3. Charu C. Agarwal, *Data Mining: The Textbook*, 1st Edition, Springer.

4. M. Gopal, *Applied Machine Learning*, 1st Edition, McGraw Hill Education.

Course: Deep Learning

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD456**
Course Title	**Deep Learning**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD361/316**
Category	**Major Elective**

Course Summary:
The goal of deep learning course is to provide a hands on knowledge on applying deep learning techniques to handle large data. The students of this course will be given opportunities to do one research project and a set of assignments.

Learning Outcomes: On successful completion of the course, students will be able to achieve the following:
1. Understand the basic Neural Networks and layers.
2. Implement different deep architectures.
3. Generate new deep architectures specific to a data.
4. Solve real life problems based on deep learning.

Curriculum Content:
Overview of Deep Learning techniques: Deep forward network, Regularization, Optimization, CNN, RNN, Methodology, Applications, Advanced topics: Autoencoders, Probabilistic models for deep learning, Generative models, GANs

Textbooks and References:
Ian Goodfellow et al., *Deep Learning*, MIT Press.

Course: Internet of Things

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD457**
Course Title	**Internet of Things**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD304**
Category	**Major Elective**

Course Summary:

Internet of Things have attracted a wide range of disciplines where close interactions with the physical world are essential. The distributed sensing capabilities and the ease of deployment provided by a wireless communication paradigm make IoT an important component of our daily lives. The course covers the basic concepts of IoT from system perspective and application development.

Learning Outcomes: On successful completion of the course, students will be able to achieve the following:
1. Understand IoT architecture and applications.
2. Understand IoT design requirements.
3. Understand IoT communication technologies.
4. Understand IoT coverage & connectivity aspects.
5. Understand the security issues in IoT.

Curriculum Content:
Introduction to IoT and IoT Domains; IoT Networking Technologies; IoT Hardware Platforms, Programming and Sensing; Introduction to IoT Analytics, Visualization, and Security; IoT Case Studies.

Course: Performance Modeling and Queuing Theory

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD459**
Course Title	**Performance Modeling and Queuing Theory**
Credits	**3**
L-T-P (Contact Hours)	**3-0-0 (L:3H - T:0H - P:0H)**
Prerequisites	**CSD210/209/ MAT205/284**
Category	**Major Elective**

Course Summary:
The course will enable the students to appreciate the power of analytical models in the analysis of the performance of computer communication networks.

Learning Outcomes: On successful completion of the course, students will be able to achieve the following:
1. The probabilistic evolution of the computer communication network.
2. Estimate various performance metrics like mean delay, loss probability, overflow probability, mean buffer size.
3. Understating of different statistical characteristics of computer communication network traffic.
4. Applicability of non-linear Markov chain for performance evaluation of ALOHA protocol.

Curriculum Content:
1. Introduction to Queueing, The Queueing Paradigm, Motivating Examples - Power of analytical modeling and queueing theory. Review of Probability.
2. Introduction to stochastic process, The Poisson process, The inter-arrival times, Exponential service times, Foundation of M/M/1 queueing system, Little's Law, Reversibility and Burke's theorem, State-dependent M/M/1 queueing system, Performance measures.
3. The M/M/1/N Queueing system; The finite buffer case, The M/M/m Queueing system: m Parallel servers with a queue, The M/M/m/m queue: A loss system; Erlang's B and C formulae. The M/G/1 queueing system, Mean number The Recursion; Pollaczek-Khinchin mean value formula.
4. Networks of queues, Open networks, The product from solution, closed queueing network.

5. Discrete time Markov chains and Aloha protocol analysis, Properties of Aloha Markov chain.
6. Real-world workloads: High variability and heavy tails, Properties of the Pareto distribution.

Textbooks and References:

1. Robertazzi T.G., *Computer Networks and Systems*, 3rd Edition, Springer, 2000.
2. Mor Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*, Cambridge university press, 2013.
3. Trivedi K.S, *Probability and Statistics, with Reliability, Queueing and Computer Science Applications*, 2nd Edition, Wiley.
4. Bertsekas D. and Gallager R., *Data Networks*, 2nd Edition, Prentice-Hall, 1992.

Course: Wireless Sensor Networks

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD464**
Course Title	**Wireless Sensor Networks**
Credits	**3**
L-T-P (Contact Hours)	**2-0-1 (L:2H - T:0H - P:2H)**
Prerequisites	**CSD304**
Category	**Major Elective**

Course Summary:
Wireless sensor networks (WSNs) have attracted a wide range of disciplines where close interactions with the physical world are essential. The distributed sensing capabilities and the ease of deployment provided by a wireless communication paradigm make WSNs an important component of our daily lives. The course covers the basic concepts of WSN from a system perspective and application development. This course deals with comprehensive knowledge about wireless sensor networks. It provides insight into different layers and their design considerations.

Learning Outcomes: On successful completion of the course, students will be able to achieve the following:
1. Understand WSN architecture and applications.
2. Understand WSN design requirements.
3. Understand and analyze various WSN protocols.
4. Understand WSN communication technologies.
5. Understand WSN coverage & connectivity aspects.
6. Understand the security issues in WSN.

Curriculum Content:
1. Introduction to Wireless sensor networks, Node and Network Architecture, Applications of WSN, WSN Protocol Stack.
2. WSN MAC protocols, Technologies for WSN, Sensor Deployment Mechanisms, Node Addressing.
3. Localization schemes, Time Synchronization, Network clustering, Query Models.
4. In-network data aggregation, QoS Management, Security.

Textbooks and References:
1. Holger Karl & Andreas Willig, *Protocols and Architectures for Wireless Sensor Networks*, John Wiley, 2005.

2. Kazem Sohraby, Daniel Minoli, & Taieb Znati, *Wireless Sensor Networks: Technology, Protocols, and Applications*, John Wiley, 2007.
3. Ian F. Akyildiz & Mehmet Can Vuran, *Wireless Sensor Networks*, John Wiley, 2010.
4. Waltenegus W. Dargie & Christian Poellabauer, *Fundamentals of Wireless Sensor Networks: Theory and Practice*, John Wiley, 2010.

Course: Special Topics in Artificial Intelligence

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD481**
Course Title	**Special Topics in Artificial Intelligence**
Credits	**3**
L-T-P (Contact Hours)	**3-0-0 (L:3H - T:0H - P:0H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
The course emphasizes on special topics and research problems in the emerging areas.

Learning Outcomes:
Students will learn new and recent trends in AI.

Curriculum Content:
The detailed content will be provided by the faculty conducting the course as and when required.

Textbooks and References:
Research papers

Special Topics in Applications

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD482**
Course Title	**Special Topics in Applications**
Credits	**3**
L-T-P (Contact Hours)	**3-0-0 (L:3H - T:0H - P:0H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
The course emphasis is on special topics and research problems in the emerging areas.

Learning Outcomes:
Students will learn new and recent trends in this area.

Curriculum Content:
The detailed content will be provided by the faculty conducting the course as and when required.

Textbooks and References:
Research papers

Course: Special Topics in Systems

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD483**
Course Title	**Special Topics in Systems**
Credits	**3**
L-T-P (Contact Hours)	**3-0-0 (L:3H - T:0H - P:0H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
The course emphasis is on special topics and research problems in the emerging areas.

Learning Outcomes:
Students will learn new and recent trends in this area.

Curriculum Content:
The detailed content will be provided by the faculty conducting the course as and when required.

Textbooks and References:
Research papers

Course: Special Topics in Theoretical Computer Science

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD484**
Course Title	**Special Topics in Theoretical Computer Science**
Credits	**3**
L-T-P (Contact Hours)	**3-0-0 (L:3H - T:0H - P:0H)**
Prerequisites	**CSD302**
Category	**Major Elective**

Course Summary:
The course emphasis is on special topics and research problems in the emerging areas.

Learning Outcomes:
Students will learn new and recent trends in the theoretical computer science.

Curriculum Content:
The detailed content will be provided by the faculty conducting the course as and when required.

Textbooks and References:
Research papers

Course: Special Module in Artificial Intelligence

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD485**
Course Title	**Special Module in Artificial Intelligence**
Credits	**1**
L-T-P (Contact Hours)	**1-0-0 (L:1H - T:0H - P:0H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
The students will learn about topics, which are currently at the forefront of Artificial Intelligence research. Each semester, the theme of the course may change depending on the instructor.

Learning Outcomes:
Student will understand the insight into current research problems in this area.

Curriculum Content:
The detailed content will be provided by the faculty conducting the course as and when required.

Textbooks and References:
Research papers

Course: Special Module in Applications

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD486**
Course Title	**Special Module in Applications**
Credits	**1**
L-T-P (Contact Hours)	**1-0-0 (L:1H - T:0H - P:0H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
To provide insight into current research problems in the area of Applications of Computer Science. The exact contents are of computer application may differ every year depending on the course run under this category.

Learning Outcomes:
Student will understand the insight into current research problems in this area.

Curriculum Content:
The detailed content will be provided by the faculty conducting the course as and when required.

Textbooks and References:
Research papers

Course: Special Module in Systems

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD487**
Course Title	**Special Module in Systems**
Credits	**1**
L-T-P (Contact Hours)	**1-0-0 (L:1H - T:0H - P:0H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
To provide insight into current research problems in the area of systems. The exact contents are of theoretical computer science may differ every year depending on the course run under this category.

Learning Outcomes:

Student will understand the insight into current research problems in this area.

Curriculum Content:

The detailed content will be provided by the faculty conducting the course as and when required.

Textbooks and References:
Research papers

Course: Special Module in Theoretical Computer Science

School	**School of Engineering**
Department	**Computer Science and Engineering**
Course Code	**CSD488**
Course Title	**Special Module in Theoretical Computer Science**

Credits	**1**
L-T-P (Contact Hours)	**1-0-0 (L:1H - T:0H - P:0H)**
Prerequisites	**CSD102/201**
Category	**Major Elective**

Course Summary:
To provide insight into current research problems in the area of theoretical of Computer Science. The exact contents are of theoretical computer science may differ every year depending on the course run under this category.

Learning Outcomes:

Student will understand the insight into current research problems in this area.

Curriculum Content:

The detailed content will be provided by the faculty conducting the course as and when required.

Textbooks and References:
Research papers
