DEPARTMENT OF COMPUTER SCIENCE
B.Sc. (H) Computer Science

CATEGORY-I

DISCIPLINE SPECIFIC CORE COURSE – 1: PROGRAMMING USING PYTHON

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Programming using Python	4	3	0	1	Class XII pass	Nil

Learning Objectives

This course is designed as the first course that:

- Introduces programming concepts using Python to Computer Science students.
- Focuses on the development of Python programming to solve problems of different domains.
- Introduces the concept of object- oriented programming.

Learning Outcomes:

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

- Understand the basics of programming language
- Develop, document, and debug modular Python programs.
- Apply suitable programming constructs and built-in data structures to solve a problem.
- Use and apply various data objects in Python.
- Use classes and objects in application programs and handle files.

SYLLABUS OF DSC-1

Theory

Unit – 1 **(6 hours)**

Introduction to Programming

Problem solving strategies; Structure of a Python program; Syntax and semantics; Executing simple programs in Python.

Unit – 2 **(12 hours)**

Creating Python Programs

Identifiers and keywords; Literals, numbers, and strings; Operators; Expressions; Input/output statements; Defining functions; Control structures (conditional statements, loop control statements, break, continue and pass, exit function), default arguments.

Unit – 3 **(15 hours)**

Built-in Data Structures

Mutable and immutable objects; Strings, built-in functions for string, string traversal, string operators and operations; Lists creation, traversal, slicing and splitting operations, passing list to a function; Tuples, sets, dictionaries and their operations.

Unit – 4 **(6 hours)**

Object Oriented Programming

Introduction to classes, objects and methods; Standard libraries.

Unit – 5 **(6 hours)**

File and Exception Handling

File handling through libraries; Errors and exception handling.

Practical **(30 hours)**

List of Practicals:

1. WAP to find the roots of a quadratic equation
2. WAP to accept a number 'n' and
    j. Check if 'n' is prime
    k. Generate all prime numbers till 'n'
    l. Generate first 'n' prime numbers This program may be done using functions
3. WAP to create a pyramid of the character '*' and a reverse pyramid

```
    *
   ***
  *****
 *******
*********
```

```
*********
 *******
  *****
   ***
    *
```

8. WAP that accepts a character and performs the following:
    a. print whether the character is a letter or numeric digit or a special character
    b. if the character is a letter, print whether the letter is uppercase or lowercase
    c. if the character is a numeric digit, prints its name in text (e.g., if input is 9, output is NINE)

9. WAP to perform the following operations on a string
    a. Find the frequency of a character in a string.
    b. Replace a character by another character in a string.
    c. Remove the first occurrence of a character from a string.
    d. Remove all occurrences of a character from a string.
10. WAP to swap the first n characters of two strings.
11. Write a function that accepts two strings and returns the indices of all the occurrences of the second string in the first string as a list. If the second string is not present in the first string then it should return -1.
12. WAP to create a list of the cubes of only the even integers appearing in the input list (may have elements of other types also) using the following:
    a. 'for' loop
    b. list comprehension
13. WAP to read a file and
    m. Print the total number of characters, words and lines in the file.
    n. Calculate the frequency of each character in the file. Use a variable of dictionary type to maintain the count.
    o. Print the words in reverse order.
    p. Copy even lines of the file to a file named 'File1' and odd lines to another file named 'File2'.
14. WAP to define a class Point with coordinates x and y as attributes. Create relevant methods and print the objects. Also define a method distance to calculate the distance between any two point objects.
15. Write a function that prints a dictionary where the keys are numbers between 1 and 5 and the values are cubes of the keys.
16. Consider a tuple t1=(1, 2, 5, 7, 9, 2, 4, 6, 8, 10). WAP to perform following operations:
    a. Print half the values of the tuple in one line and the other half in the next line.
    b. Print another tuple whose values are even numbers in the given tuple.
    c. Concatenate a tuple t2=(11,13,15) with t1.
    d. Return maximum and minimum value from this tuple
17. WAP to accept a name from a user. Raise and handle appropriate exception(s) if the text entered by the user contains digits and/or special characters.

Essential Readings

- Taneja, S., Kumar, N. Python Programming- A modular Approach, 1st edition, Pearson Education India, 2018.

- Balaguruswamy E. Introduction to Computing and Problem Solving using Python, 2nd edition, McGraw Hill Education, 2018.

Suggestive Readings

- Brown, Martin C. Python: The Complete Reference, 2nd edition, McGraw Hill Education, 2018.
- Guttag, J.V. Introduction to computation and programming using Python, 2nd edition, MIT Press, 2016.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DISCIPLINE SPECIFIC CORE COURSE – 2: COMPUTER SYSTEM ARCHITECTURE

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

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

Learning Objectives

The Learning Objectives of this course are as follows:

- Introduces the students to the fundamental concepts of digital computer organization, design and architecture.
- Develop a basic understanding of the building blocks of the computer system and highlights how these blocks are organized together to architect a digital computer system.

Learning Outcomes

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

- Design Combinational Circuits using basic building blocks. Simplify these circuits using Boolean algebra and Karnaugh maps. Differentiate between combinational circuits and sequential circuits.
- Represent data in binary form, convert numeric data between different number systems and perform arithmetic operations in binary.
- Determine various stages of instruction cycle, pipelining and describe interrupts and their handling.
- Explain how CPU communicates with memory and I/O devices and distinguish between different types of processors.
- Simulate the design of a basic computer using a software tool.

SYLLABUS OF DSC - 2

Theory

Unit – 1 **(6 hours)**

Digital Logic Circuits

Logic Gates, Truth Tables, Boolean Algebra, Digital Circuits, Combinational Circuits, Introduction to Sequential Circuits, Circuit Simplification using Karnaugh Map, Don't Care Conditions, Flip-Flops, Characteristic Tables, Excitation Table.

Unit – 2 **(9 hours)**

Digital Components (Fundamental building blocks)

Designing of combinational circuits- Half Adder, Full Adder, Decoders, Encoders, Multiplexers, Registers and Memory (RAM, ROM and their types), Arithmetic Microoperations, Binary Adder, Binary Adder-Subtractor.

Unit – 3 **(6 hours)**

Data Representation and Basic Computer Arithmetic

Number System, r and (r-1)'s Complements, data representation and arithmetic operations.

Unit – 4 **(9 hours)**

Basic Computer Organization and Design

Bus organization, Microprogrammed vs Hardwired Control, Instruction Codes, Instruction Format, Instruction Cycle, Instruction pipelining, Memory Reference, Register Reference and Input Output Instructions, Program Interrupt and Interrupt Cycle..

Unit – 5 **(6 hours)**

Processors

General register organization, Stack Organization, Addressing Modes, Overview of Reduced Instruction Set Computer (RISC), Complex Instruction Set Computer (CISC), Multicore processor and Graphics Processing Unit (GPU).

Unit – 6 **(9 hours)**

Memory and Input-Output Organization

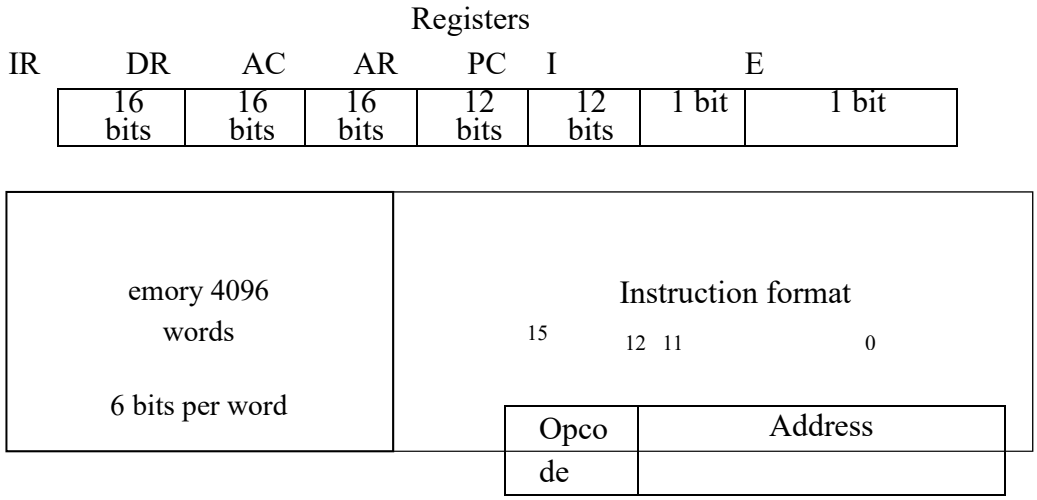
Memory hierarchy (main, cache and auxiliary memory), Input-Output Interface, Modes of Transfer: Programmed I/O, Interrupt initiated I/O, Direct memory access.

Practical **(30 hours)**

List of Practicals:

(Use Simulator – CPU Sim 3.6.9 or any higher version for the implementation)

1. Create a machine based on the following architecture:



Basic Computer Instructions

Memory Reference			Register Reference	
Symbol	Hex		Symbol	Hex
AND	0xxx	Direct Addressing	CLA	7800
ADD	1xxx		CLE	7400
LDA	2xxx		CMA	7200
STA	3xxx		CME	7100
BUN	4xxx		CIR	7080
BSA	5xxx		CIL	7040
ISZ	6xxx		INC	7020
AND_I	8xxx	Indirect Addressing	SPA	7010
ADD_I	9xxx		SNA	7008
LDA_I	Axxx		SZA	7004
STA_I	Bxxx		SZE	7002
BUN_I	Cxxx		HLT	7001
BSA_I	Dxxx		INP	F800
ISZ_I	Exxx		OUT	F400

Refer to Chapter-5 of reference 1 for description of instructions.

Design the register set, memory and the instruction set. Use this machine for the assignments of this section.

2. Create a Fetch routine of the instruction cycle.

3. Write an assembly program to simulate ADD operation on two user-entered numbers.
4. Write an assembly program to simulate SUBTRACT operation on two user-entered numbers.
5. Write an assembly program to simulate the following logical operations on two user-entered numbers.
    i. AND
    ii. OR
    iii. NOT
    iv. XOR
    v. NOR
    vi. NAND
6. Write an assembly program for simulating following memory-reference instructions.
    i. ADD
    ii. LDA
    iii. STA
    iv. BUN
    v. ISZ
7. Write an assembly language program to simulate the machine for following register reference instructions and determine the contents of AC, E, PC, AR and IR registers in decimal after the execution:
    i. CLA
    ii. CMA
    iii. CME
    iv. HLT
8. Write an assembly language program to simulate the machine for following register reference instructions and determine the contents of AC, E, PC, AR and IR registers in decimal after the execution:
    i. INC
    ii. SPA
    iii. SNA
    iv. SZE

9. Write an assembly language program to simulate the machine for following register reference instructions and determine the contents of AC, E, PC, AR and IR registers in decimal after the execution:

    i. CIR

    ii. CIL

10. Write an assembly program that reads in integers and adds them together; until a negative non-zero number is read in. Then it outputs the sum (not including the last number).

11. Write an assembly program that reads in integers and adds them together; until zero is read in. Then it outputs the sum.

Essential Readings

- David A. Patterson and John L. Hennessy. "Computer Organization and Design: The Hardware/Software interface", 5th edition, Elsevier, 2012.
- Mano, M. Computer System Architecture, 3rd edition, Pearson Education, 1993.

Suggestive Readings

- Mano, M. Digital Design, Pearson Education Asia, 1995.
- Null, L., & Lobur, J. The Essentials of Computer Organization and Architecture. 5th edition, (Reprint) Jones and Bartlett Learning, 2018.
- Stallings, W. Computer Organization and Architecture Designing for Performance 8th edition, Prentice Hall of India, 2010.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DISCIPLINE SPECIFIC CORE COURSE – 3: MATHEMATICS FOR COMPUTING

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

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

Learning Objectives

The Learning Objectives of this course are as follows:

- Introduces the students to the fundamental concepts and topics of linear algebra and vector calculus.
- To build the foundation for some of the core courses in later semesters.

Learning Outcomes

This course will enable the students to:

- Perform operations on matrices and sparse matrices.
- Compute the determinant, rank and eigenvalues of a matrix.
- Perform diagonalization.
- Perform operations on vectors, the dot product and cross product.
- Represent vectors geometrically and calculate the gradient, divergence, curl.
- Apply linear algebra and vector calculus to solve problems in sub-disciplines of computer science.

SYLLABUS OF DSC – 3

Theory

Unit – 1 **(6 hours)**

Introduction to Matrix Algebra

Echelon form of a Matrix, Rank of a Matrix, Determinant and Inverse of a matrix, Solution of System of Homogeneous & Non-Homogeneous Equations: Gauss elimination and Solution of System of Homogeneous Equations: Gauss Jordan Method.

Unit – 2 **(21 hours)**

Vector Space and Linear Transformation

Vector Space, Sub-spaces, Linear Combinations, Linear Span, Convex Sets, Linear Independence/Dependence, Basis & Dimension, Linear transformation on finite dimensional vector spaces, Inner Product Space, Schwarz Inequality, Orthonormal Basis, Gram-Schmidt Orthogonalization Process.

Unit – 3 **(9 hours)**

EigenValue and EigenVector

Characteristic Polynomial, Cayley Hamilton Theorem, Eigen Value and Eigen Vector of a matrix, Eigenspaces, Diagonalization, Positive Definite Matrices, Applications to Markov Matrices.

Unit – 4 **(9 hours)**

Vector Calculus

Vector Algebra, Laws of Vector Algebra, Dot Product, Cross Product, Vector and Scalar Fields, Ordinary Derivative of Vectors, Space Curves, Partial Derivatives, Del Operator, Gradient of a Scalar Field, Directional Derivative, Gradient of Matrices, Divergence of a Vector Field, Laplacian Operator, Curl of a Vector Field.

Practical **(30 hours)**

List of Practicals:

1. Create and transform vectors and matrices (the transpose vector (matrix) conjugate transpose of a vector (matrix))
2. Generate the matrix into echelon form and find its rank.
3. Find cofactors, determinant, adjoint and inverse of a matrix.
4. Solve a system of Homogeneous and non-homogeneous equations using Gauss elimination method.
5. Solve a system of Homogeneous equations using the Gauss Jordan method.
6. Generate basis of column space, null space, row space and left null space of a matrix space.
7. Check the linear dependence of vectors. Generate a linear combination of given vectors of Rn/ matrices of the same size and find the transition matrix of given matrix space.
8. Find the orthonormal basis of a given vector space using the Gram-Schmidt orthogonalization process.
9. Check the diagonalizable property of matrices and find the corresponding eigenvalue and verify the Cayley- Hamilton theorem.
10. Application of Linear algebra: Coding and decoding of messages using nonsingular matrices.
    eg code “Linear Algebra is fun” and then decode it.
11. Compute Gradient of a scalar field.
12. Compute Divergence of a vector field.
13. Compute Curl of a vector field.

Essential Reading

- Strang Gilbert. Introduction to Linear Algebra, 5th Edition, Wellesley-Cambridge Press, 2021.
- Kreyszig Erwin. Advanced Engineering Mathematics, 10th Edition, Wiley, 2015.
- Strang Gilbert. Linear Algebra and Learning from Data, 1st Edition, Wellesley-Cambridge Press, 2019.
- Jain R. K., Iyengar S.R. K. Advanced Engineering Mathematics, 5th Edition, Narosa, 2016.

Suggestive Reading

- Deisenroth, Marc Peter, Faisal A. Aldo and Ong Cheng Soon. Mathematics for Machine Learning, 1st Edition, Cambridge University Press, 2020.
- (Lipschutz Seymour and Lipson Marc. Schaum's Outline of Linear Algebra, 6th Edition, McGraw Hill, 2017.

DEPARTMENT OF COMPUTER SCIENCE

BSc. (Hons.) Computer Science -DSC

Category I

DISCIPLINE SPECIFIC CORE COURSE – 4: Object Oriented Programming

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC04 Object Oriented Programming with C++	4	3	0	1	Class XII pass with Mathematics	Nil

Learning Objectives

This course is designed to introduce programming concepts using C++ to students. The course aims to develop structured as well as object-oriented programming skills using C++ programming language. The course also aims to achieve competence amongst its students to develop correct and efficient C++ programs to solve problems spanning multiple domains.

Learning outcomes

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

- Write simple programs using built-in data types of C++.
- Implement arrays and user defined functions in C++.
- Write programs using dynamic memory allocation, handling external files, interrupts and exceptions.
- Solve problems spanning multiple domains using suitable programming constructs in C++.
- Solve problems spanning multiple domains using the concepts of object oriented programming in C++.

SYLLABUS OF DSC-4

UNIT – I (3 Hours)

Introduction to C++: Overview of Procedural and Object-Oriented Programming, Using main() function, Header Files, Compiling and Executing Simple Programs in C++

UNIT – II (12 Hours)

Programming Fundamentals: Data types, Variables, Operators, Expressions, Arrays, Keywords, Decision making constructs, Iteration, Type Casting, Input-output statements, Functions, Command Line Arguments/Parameters

UNIT – III (15 Hours)

Object Oriented Programming: Concepts of Abstraction, Encapsulation. Creating Classes and objects, Modifiers and Access Control, Constructors, Destructors, Implementation of Inheritance and Polymorphism, Template functions and classes

UNIT – IV (9 Hours)

Pointers and References: Static and dynamic memory allocation, Pointer and Reference Variables, Implementing Runtime polymorphism using pointers and references

UNIT – V (6 Hours)

Exception and File Handling: Using try, catch, throw, throws and finally; Nested try, creating user defined exceptions, File I/O Basics, File Operations

Practical component (if any) -30 Hours

1. Write a program to compute the sum of the first n terms of the following series:

   $sum = 1 - \frac{1}{2^2} + \frac{1}{3^3} -$ . . . .

   The number of terms n is to be taken from the user through the command line. If the command line argument is not found then prompt the user to enter the value of n.
2. Write a program to remove the duplicates from an array.
3. Write a program that prints a table indicating the number of occurrences of each alphabet in the text entered as command line arguments.
4. Write a menu driven program to perform string manipulation (without using inbuilt string functions):
   a. Show address of each character in string
   b. Concatenate two strings.
   c. Compare two strings
   d. Calculate length of the string (use pointers)
   e. Convert all lowercase characters to uppercase
   f. Reverse the string
   g. Insert a string in another string at a user specified position
5. Write a program to merge two ordered arrays to get a single ordered array.

6. Write a program to search a given element in a set of N numbers using Binary search

   (i) with recursion (ii) without recursion.

7. Write a program to calculate GCD of two numbers (i) with recursion (ii) without recursion.

8. Create a Matrix class. Write a menu-driven program to perform following Matrix operations (exceptions should be thrown by the functions if matrices passed to them are incompatible and handled by the main() function):

   a. Sum

   b. Product

   c. Transpose

9. Define a class Person having name as a data member. Inherit two classes Student and Employee from Person. Student has additional attributes as course, marks and year and Employee has department and salary. Write display() method in all the three classes to display the corresponding attributes. Provide the necessary methods to show runtime polymorphism.

10. Create a Triangle class. Add exception handling statements to ensure the following conditions: all sides are greater than 0 and sum of any two sides are greater than the third side. The class should also have overloaded functions for calculating the area of a right angled triangle as well as using Heron's formula to calculate the area of any type of triangle.

11. Create a class Student containing fields for Roll No., Name, Class, Year and Total Marks. Write a program to store 5 objects of Student class in a file. Retrieve these records from the file and display them.

12. Copy the contents of one text file to another file, after removing all whitespaces.

Essential/recommended readings

1. Stephen Prata, *C++ Primer Plus*, 6th Edition, Pearson India, 2015.
2. E Balaguruswamy, *Object Oriented Programming with C++*, 8th edition, McGraw-Hill Education, 2020.
3. D.S. Malik, *C++ Programming: From Problem Analysis to Program Design*, 6th edition, Cengage Learning, 2013.

Suggestive readings

(i) Schildt, H. *C++: The Complete Reference*, 4th edition, McGraw Hill, 2003

(ii) Forouzan, A. B., Gilberg, R. F. *Computer Science: A Structured Approach using C++,* 2nd edition, Cengage Learning, 2010

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DISCIPLINE SPECIFIC CORE COURSE – 5: Discrete Mathematical Structures

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC 05 Discrete Mathematical Structures	**4**	**3**	**0**	**1**	**Class XII pass with Mathematics**	**Nil**

Learning Objectives

This course is designed as a foundational course to make students learn about the mathematical constructs that are used in Computer Science such as Boolean algebra, sets, relations, functions, principles of counting, and recurrences. In this course, the knowledge of mathematical notation, ideas and concepts learnt at the pre-college levels is extended to orient the students towards mathematical thinking required in Computer Science.

Learning outcomes

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

- Relate mathematical concepts and terminology to examples in the domain of Computer Science.
- Model real world problems using various mathematical constructs.
- Use different proofing techniques; construct simple mathematical proofs using logical arguments.
- Formulate mathematical claims and construct counterexamples.

SYLLABUS OF DSC- 5
UNIT – I (06 Hours)
Sets, Functions, Sequences and Summations, Relations: Sets: Set Operations, Computer Representation of Sets, Countable and Uncountable Set, Principle of Inclusion and Exclusion, Multisets; Functions: One-to-one and Onto Functions, Inverse Functions and Compositions of

Functions, Graphs of Functions Sequences and Summations: Sequences, Special Integer Sequences, Summations; Relations: Properties of Binary Relations, Equivalence relations and Partitions, Partial Ordering Relations and Lattices.

UNIT – II (09 Hours)

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

UNIT – III (09 Hours)

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

UNIT – IV (06 Hours)

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

UNIT – V (09 Hours)

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

UNIT – VI (06 Hours)

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

Practical component (if any) – 30 Hours

1. Create a class SET. Create member functions to perform the following SET operations:
    1) is member: check whether an element belongs to the set or not and return value as true/false.
    2) powerset: list all the elements of the power set of a set .
    3) subset: Check whether one set is a subset of the other or not.
    4) union and Intersection of two Sets.
    5) complement: Assume Universal Set as per the input elements from the user.
    6) set Difference and Symmetric Difference between two sets.
    7) cartesian Product of Sets.

    Write a menu driven program to perform the above functions on an instance of the SET class.

2. Create a class RELATION, use Matrix notation to represent a relation. Include member functions to check if the relation is Reflexive, Symmetric, Anti-symmetric, Transitive. Using these functions check whether the given relation is: Equivalence or Partial Order relation or None

3. Write a Program that generates all the permutations of a given set of digits, with or without repetition.
4. For any number n, write a program to list all the solutions of the equation $x_1 + x_2 + x_3 + ...+ x_n = C$, where C is a constant (C<=10) and $x_1, x_2, x_3, ..., x_n$ are nonnegative integers, using brute force strategy.
5. Write a Program to evaluate a polynomial function. (For example store $f(x) = 4n^2 + 2n + 9$ in an array and for a given value of n, say n = 5, compute the value of f(n)).
6. Write a Program to check if a given graph is a complete graph. Represent the graph using the Adjacency Matrix representation.
7. Write a Program to check if a given graph is a complete graph. Represent the graph using the Adjacency List representation.
8. Write a Program to accept a directed graph G and compute the in-degree and out-degree of each vertex.

Essential/recommended readings

1. Liu, C. L., Mohapatra, D. P. *Elements of Discrete Mathematics: A Computer Oriented Approach*, 4th edition, Tata McGraw Hill, 2017.
2. Rosen, K. H.. *Discrete Mathematics and Its Applications*, 8th edition, McGraw Hill, 2018.

Suggestive readings

(i) Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein C. *Introduction to Algorithms*, 4th edition, Prentice Hall of India. 2022.
(ii) Trembley, J. P., Manohar, R. *Discrete Mathematical Structures with Application to Computer Science*, Tata McGraw Hill, 1997.
(iii) Albertson, M. O. and Hutchinson, J. P. *Discrete Mathematics with Algorithms*, John Wiley and Sons, 1988.

DISCIPLINE SPECIFIC CORE COURSE – 6: Probability for Computing

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC06 Probability for computing	**4**	**3**	**0**	**1**	Class XII pass with Mathematics	**Nil**

Learning Objectives

This course introduces the students to the fundamental concepts and topics of probability and statistics, whose knowledge is important in other computer science courses. The course aims to build the foundation for some of the core courses in later semesters.

Learning outcomes

After successful completion of this course, the student will be able to:

- Use probability theory to evaluate the probability of real-world events.
- Describe discrete and continuous probability distribution functions and generate random numbers from the given distributions.
- Find the distance between two probability distributions
- Define and quantify the information contained in the data.
- Perform data analysis in a probabilistic framework.
- Visualize and model the given problem using mathematical concepts covered in the course.

SYLLABUS OF DSC-6 UNIT-I (09 Hours)

Basic Probability: Introduction to the notion of probability, Random experiment, Sample space and Events, Probability defined on events, Algebra of events. Conditional probabilities, independent events, Bayes' theorem.

UNIT-II (12 Hours)

Random Variables: Introduction to Random Variables, Probability mass/density functions, Cumulative distribution functions. Discrete Random Variables (Bernoulli, Binomial, Poisson, Multinomial and Geometric). Continuous Random Variables (Uniform, Exponential and Normal). Expectation of a Random Variable, Expectation of Function of a Random Variable and Variance. Markov inequality, Chebyshev's inequality, Central Limit Theorem, Weak and Strong Laws of Large Numbers.

UNIT-III (09 Hours)

Joint Distributions: Jointly distributed Random Variables, Joint distribution functions, Independent Random Variables, Covariance of Random Variables, Correlation Coefficients,

Conditional Expectation.

UNIT-IV (15 Hours)

Markov Chain and Information Theory: Introduction to Stochastic Processes, Chapman–Kolmogorov equations, Classification of states, Limiting and Stationary Probabilities. Random Number Generation, Pseudo Random Numbers, Inverse Transformation Method, Rejection Method, Uncertainty, Information and Entropy, Mutual Information, KL Divergence.

Practical component (if any) – 30 Hours

The goal of this lab is to develop data interpretation skills. Following exercises are designed to enable students to understand data characteristics either by visualization or by interpreting computed measures. All the exercises are to be completed using MS Excel functions and graphs. At the end of each exercise, the student should be able to draw a conclusion and state in a concise manner. Teachers are expected to guide students to obtain real data available through the internet for the following exercises.

1. Plotting and fitting of Binomial distribution and graphical representation of probabilities.
2. Plotting and fitting of Multinomial distribution and graphical representation of probabilities.
3. Plotting and fitting of Poisson distribution and graphical representation of probabilities.
4. Plotting and fitting of Geometric distribution and graphical representation of probabilities.
5. Plotting and fitting of Uniform distribution and graphical representation of probabilities.
6. Plotting and fitting of Exponential distribution and graphical representation of probabilities.
7. Plotting and fitting of Normal distribution and graphical representation of probabilities.
8. Calculation of cumulative distribution functions for Exponential and Normal distribution.
9. Given data from two distributions, find the distance between the distributions.
10. Application problems based on the Binomial distribution.
11. Application problems based on the Poisson distribution.
12. Application problems based on the Normal distribution.
13. Presentation of bivariate data through scatter-plot diagrams and calculations of covariance.
14. Calculation of Karl Pearson's correlation coefficients.
15. To find the correlation coefficient for a bivariate frequency distribution.
16. Generating Random numbers from discrete (Bernoulli, Binomial, Poisson) distributions.

17. Generating Random numbers from continuous (Uniform, Normal) distributions.
18. Find the entropy from the given data set.

Essential/recommended readings

1. Ross **S**heldon M. *Introduction to Probability Models*, 12th Edition, Elsevier, 2019.
2. Trivedi, K. S. *Probability and Statistics with Reliability, Queuing and Computer Science Applications*, 2nd edition, Wiley, 2015.
3. Deisenroth, Marc Peter, Faisal A. Aldo and Ong Cheng Soon, *Mathematics for Machine Learning*, 1st edition, Cambridge University Press, 2020.
4. Ian F. Blake, *An Introduction to Applied Probability*, John Wiley.

Suggestive readings

(i) Johnson James L., *Probability and Statistics for Computer Science*, 6th edition, Wiley, 2004.

(ii) Forsyth David, *Probability and Statistics for Computer Science*, 1st edition, Springer, 2019.

(iii) Freund J.E., *Mathematical Statistics with Applications*, 8th edition, Pearson Education, 2013.

(iv) Devore Jay L., *Probability and Statistics for Engineering and the Sciences*, 9th edition, Cengage Learning, 2020.

UNIVERSITY OF DELHI

CNC-II/093/1(25)/2023-24/64
Dated: 30.05.2023

N O T I F I C A T I O N

Sub: Amendment to Ordinance V

[E.C Resolution No. 60/ (60-1-7/) dated 03.02.2023]

Following addition be made to Appendix-II-A to the Ordinance V (2-A) of the Ordinances of the University;

Add the following:

Syllabi of Semester-III of the following departments under Faculty of Mathematical Sciences based on Under Graduate Curriculum Framework -2022 implemented from the Academic Year 2022-23.

FACULTY OF MATHEMATICAL SCIENCES

DEPARTMENT OF COMPUTER SCIENCE

BSC. (HONS.) COMPUTER SCIENCE

DISCIPLINE SPECIFIC CORE COURSE -7 (DSC-7) : Data Structures

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC07 Data Structures	**4**	**3**	**0**	**1**	**Passed 12th class with Mathematics**	Programming using Python/Object Oriented Programming with C++

Learning Objectives

The course aims at developing the ability to use basic data structures like arrays, stacks, queues, lists, and trees to solve problems. C++ is chosen as the language to implement the implementation of these data structures.

Learning outcomes

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

- Compare two functions for their rates of growth.
- Understand abstract specification of data-structures and their implementation.
- Compute time and space complexity of operations on a data-structure.
- Identify the appropriate data structure(s) for a given application and understand the trade-offs involved in terms of time and space complexity.
- Apply recursive techniques to solve problems.

SYLLABUS OF DSC-7

Unit 1 (9 hours)

Growth of Functions, Recurrence Relations: Functions used in analysis, asymptotic notations, asymptotic analysis, solving recurrences using recursion trees, Master Theorem.

Unit 2 (16 hours)

Arrays, Linked Lists, Stacks, Queues: Arrays: array operations, applications, two-dimensional arrays, dynamic allocation of arrays; Linked Lists: singly linked lists, doubly linked lists, circularly linked lists, Stacks: stack as an ADT, implementing stacks using arrays, implementing stacks using linked lists, applications of stacks; Queues: queue as an ADT, implementing queues using arrays, implementing queues using linked lists,. Time complexity analysis.

Unit 3 (5 hours)

Recursion: Recursive functions, linear recursion, binary recursion.

Unit 4 (6 hours)

Trees, Binary Trees: Trees: definition and properties, tree traversal algorithms, and their time complexity analysis; binary trees: definition and properties, traversal of binary trees, and their time complexity analysis.

Unit 5 (7 hours)

Binary Search Trees, Balanced Search Trees: Binary Search Trees: insert, delete, search operations, time complexity analysis of these operations; Balanced Search Trees: insert, search operations, time complexity analysis of these operations. Time complexity analysis.

Unit 6 (2 hours)

Binary Heap: Binary Heaps: heaps, heap operations.

Essential/recommended readings

1. Goodrich, M.T., Tamassia, R., & Mount, D., *Data Structures and Algorithms Analysis in C++,* 2nd edition, Wiley, 2011.
2. Cormen, T.H., Leiserson, C.E., Rivest, R. L., Stein C. *Introduction to Algorithms*, 4th edition, Prentice Hall of India, 2022.

Additional references

1. Sahni, S. *Data Structures, Algorithms and applications in C++*, 2nd edition, Universities Press, 2011.
2. Langsam Y., Augenstein, M. J., & Tanenbaum, A. M. *Data Structures Using C and C++*, Pearson, 2009.

Practical List (If any): (30 Hours)

Practical exercises such as

1. Write a program to implement singly linked list as an ADT that supports the following operations:
   (i) Insert an element x at the beginning of the singly linked list
   (ii) Insert an element x at i^{th} position in the singly linked list
   (iii) Remove an element from the beginning of the singly linked list
   (iv) Remove an element from i^{th} position in the singly link
   (v) Search for an element x in the singly linked list and return its pointer
   (vi) Concatenate two singly linked lists

2. Write a program to implement doubly linked list as an ADT that supports the following operations:
   (i) Insert an element x at the beginning of the doubly linked list
   (ii) Insert an element x at i^{th} position in the doubly linked list
   (iii) Insert an element x at the end of the doubly linked list
   (iv) Remove an element from the beginning of the doubly linked list
   (v) Remove an element from i^{th} position in the doubly linked list.
   (vi) Remove an element from the end of the doubly linked list
   (vii) Search for an element x in the doubly linked list and return its pointer
   (viii) Concatenate two doubly linked lists

3. Write a program to implement circular linked list as an ADT which supports the following operations:
   (i) Insert an element x at the front of the circularly linked list
   (ii) Insert an element x after an element y in the circularly linked list
   (iii) Insert an element x at the back of the circularly linked list
   (iv) Remove an element from the back of the circularly linked list
   (v) Remove an element from the front of the circularly linked list
   (vi) Remove the element x from the circularly linked list
   (vii) Search for an element x in the circularly linked list and return its pointer

(viii) Concatenate two circularly linked lists

4. Implement a stack as an ADT using Arrays.

5. Implement a stack as an ADT using the Linked List ADT.

6. Write a program to evaluate a prefix/postfix expression using stacks.

7. Implement Queue as an ADT using the circular Arrays.

8. Implement Queue as an ADT using the Circular Linked List ADT.

9. Write a program to implement Binary Search Tree as an ADT which supports the following operations:
   (i) Insert an element x
   (ii) Delete an element x
   (iii) Search for an element x in the BST and change its value to y and then place the node with value y at its appropriate position in the BST
   (iv) Display the elements of the BST in preorder, inorder, and postorder traversal
   (v) Display the elements of the BST in level-by-level traversal
   (vi) Display the height of the BST

10. Write a program to implement a balanced search tree as an ADT.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DISCIPLINE SPECIFIC CORE COURSE – 8 (DSC-8): Operating Systems

Credit distribution, Eligibility and Prerequisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC 08 Operating Systems	**4**	**3**	**0**	**1**	**Passed 12th class with Mathematics**	Programming using Python/Object Oriented Programming with C++, Computer System Architecture

Learning Objectives

The course provides concepts that underlie all operating systems and are not tied to any particular operating system. The emphasis is on explaining the need and structure of an operating system using its common services such as process management (creation, termination etc.), CPU Scheduling, Process Synchronization, Handling Deadlocks, main memory management, virtual memory, secondary memory management. The course also introduces various scheduling algorithms, structures, and techniques used by operating systems to provide these services.

Learning outcomes

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

- Describe the need of an operating system and define multiprogramming and Multithreading concepts.
- Implement the process synchronization service (Critical Section, Semaphores), CPU scheduling service with various algorithms.
- Implement Main memory Management (Paging, Segmentation) algorithms, Handling of Deadlocks
- Identify and appreciate the File systems Services, Disk Scheduling service

SYLLABUS OF DSC-8

Unit 1 (6 hours)

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

Unit 2 (9 hours)

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

Unit 3 (10 hours)

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

Unit 4 (11 hours)

Memory Management: Physical and Logical address space, Swapping, Contiguous memory allocation strategies - fixed and variable partitions, Segmentation, Paging.
Virtual Memory Management: Demand Paging and Page Replacement algorithms: FIFO Page Replacement, Optimal Page replacement, LRU page replacement.

Unit 5 (9 hours)

File System: File Concepts, File Attributes, File Access Methods, Directory Structure: Single-Level, Two-Level, Tree-Structured, and Acyclic-Graph Directories.
Mass Storage Structure: Magnetic Disks, Solid-State Disks, Magnetic Tapes, Disk Scheduling algorithms: FCFS, SSTF, SCAN, C-SCAN, LOOK, and C-LOOk Scheduling.

Essential/recommended readings

1. Silberschatz, A., Galvin, P. B., Gagne G. *Operating System Concepts,* 9th edition, John Wiley Publications, 2016.
2. Tanenbaum, A. S. *Modern Operating Systems*, 3rd edition, Pearson Education, 2007.
3. Stallings, W. *Operating Systems: Internals and Design Principles,* 9th edition, Pearson Education, 2018.

Additional References

1. Dhamdhere, D. M., *Operating Systems: A Concept-based Approach,* 2nd edition, Tata McGraw-Hill Education, 2017.
2. Kernighan, B. W., Rob Pike, R. *The Unix Programming Environment*, Englewood Cliffs, NJ: Prentice-Hall, 1984.

Suggested Practical List (If any): (30 Hours)

Practical exercises such as

1. Execute various Linux commands for:
   i. Information Maintenance: wc, clear, cal, who, date, pwd
   ii. File Management: cat, cp, rm, mv, cmp, comm, diff, find, grep, awk
   iii. Directory Management : cd, mkdir, rmdir, ls
2. Execute various Linux commands for:
   i. Process Control: fork, getpid, ps, kill, sleep
   ii. Communication: Input-output redirection, Pipe
   iii. Protection Management: chmod, chown, chgrp
3. Write a programme (using fork() and/or exec() commands) where parent and child execute:
   i. same program, same code.
   ii. same program, different code.
   iii. Before terminating, the parent waits for the child to finish its task.
4. Write a program to to report behaviour of Linux kernel including kernel version, CPU type and model. (CPU information)

5. Write a program to report behaviour of Linux kernel including information on 19 configured memory, amount of free and used memory. (Memory information)
6. Write a program to copy files using system calls.
7. Use an operating system simulator to simulate operating system tasks.
8. Write a program to implement scheduling algorithms FCFS/ SJF/ SRTF/ non-preemptive scheduling algorithms.
9. Write a program to calculate the sum of n numbers using Pthreads. A list of n numbers is divided into two smaller lists of equal size, and two separate threads are used to sum the sublists.
10. Write a program to implement first-fit, best-fit and worst-fit allocation strategies.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DISCIPLINE SPECIFIC CORE COURSE– 9 (DSC-9): Numerical Optimization

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC09 Numerical Optimization	**4**	**3**	**0**	**1**	**Passed 12th class with Mathematics**	Programming using Python/Object Oriented Programming with C++

Learning Objectives

The course aims to provide students with the experience of mathematically formulating a large variety of optimization/decision problems emerging out of various fields like data science, machine learning, business, and finance. The course focuses on learning techniques to optimize problems in order to obtain the best possible solution.

Learning outcomes

At the end of the course, students will be able to:

- Mathematically formulate the optimization problems using the required number of independent variables.
- Define constraint functions on a problem.
- Check the feasibility and optimality of a solution.
- Apply conjugate gradient method to solve the problem.

SYLLABUS OF DSC-9

Unit 1 (6 hours)

Introduction: Mathematical Formulation using example, Continuous versus Discrete Optimization, Constrained and Unconstrained Optimization, Global and Local Optimization, Stochastic and Deterministic Optimization, Convexity, Optimization Algorithms

Unit 2 (14 hours)

Fundamentals of Unconstrained Optimization: Concept of a Solution - Recognizing a Local Minimum, Nonsmooth Problems, Overview of Algorithms - Two Strategies: Line Search and Trust Region, Search Directions for Line Search Methods, Models for Trust-Region Methods, Scaling. Line Search - Convergence of Line Search Methods, Rate of Convergence - Convergence Rate of Steepest Descent; Newton's Method, Quasi-Newton Methods. Trust Region - The Cauchy Point Algorithm; Global Convergence - Reduction Obtained by the Cauchy Point; Convergence to Stationary Points.

Unit 3 (7 hours)

Conjugate Gradient Methods: Basic Properties of the Conjugate Gradient Method, A Practical Form of the Conjugate Gradient Method, and Rate of Convergence

Unit 4 (8 hours)

Calculating Derivatives: Finite-Difference Derivative Approximations, Approximating the Gradient, Approximating a Sparse Jacobian, Approximating the Hessian, Approximating a Sparse Hessian

Unit 5 (10 hours)

Theory of Constrained Optimization: Local and Global Solutions, Smoothness, Examples - A Single Equality Constraint, A Single Inequality Constraint, Two Inequality Constraints, Tangent Cone and Constraint Qualifications, First-Order Optimality Condition, Second-Order Conditions - Second-Order Conditions and Projected Hessians. Linear and non-linear constrained optimization, augmented Lagrangian Method

Essential/recommended readings

1. J. Nocedal and S.J. Wright, *Numerical Optimization*, 2nd edition, Springer Series in Operations Research, 2006.
2. A, Mehra, S Chandra, Jayadeva, *Numerical Optimization with Applications*, Narosa Publishing House, New Delhi, 2009,

Additional References

1. R. W. Hamming, *Numerical Methods for Scientists and Engineers,* 2nd edition, Dover Publications, 1986.
2. Q. Kong, T. Siauw, A. Bayen, *Python Programming and Numerical Methods: A Guide for Engineers and Scientists,* 1st edition, 2020.

Suggested Practical List (If any) :(30 Hours)

Practical exercises such as

Write a program to implement the following methods:

Constrained and Unconstrained Optimization, Global and Local Optimization, Line Search and Trust Region, Convergence of Line Search Methods, Rate of Convergence - Convergence Rate of Steepest Descent, Newton's Method, Quasi-Newton Methods, The Cauchy Point algorithm, Finite-Difference Derivative Approximations, Convergence to Stationary Points, Conjugate Gradient Method, Rate of Convergence, Approximating a Sparse Jacobian, Approximating the Hessian, Approximating a Sparse Hessian, First-Order Optimality Condition, Second-Order Conditions - Second-Order Conditions, and Projected Hessians. Linear and non-linear constrained optimization Augmented Lagrangian Methods.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

Computer Science Courses for Undergraduate Programme of study with Computer Science discipline Elective

DISCIPLINE SPECIFIC ELECTIVE COURSE: Data Analysis and Visualization

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Data Analysis and Visualization (DAV)	**4**	**3**	**0**	**1**	**Pass in XII class**	Programming using Python/ Class XI-XII Computer Science/ Class XI-XII Informatics Practices

Learning Objectives

This course is designed to introduce the students to real-world data analysis problems, the use of statistics to get a deterministic view of data, and interpreting results in the field of exploratory data science using Python. This course is the first in the "Data Science" pathway and builds the foundation for three subsequent courses in the pathway.

Learning outcomes

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

1. Apply descriptive statistics to obtain a deterministic view of data
2. Perform data handling using Numpy arrays
3. Load, clean, transform, merge, and reshape data using Pandas
4. Visualize data using Pandas and matplot libraries
5. Solve real world data analysis problems

SYLLABUS OF DSE

Unit 1 (10 hours)

Introduction to basic statistics and analysis: Fundamentals of Data Analysis, Statistical foundations for Data Analysis, Types of data, Descriptive Statistics, Correlation and covariance, Linear Regression, Statistical Hypothesis Generation and Testing, Python Libraries: NumPy, Pandas, Matplotlib

Unit 2 (8 hours)

Array manipulation using Numpy: Numpy array: Creating Numpy arrays; various data types of Numpy arrays, indexing and slicing, swapping axes, transposing arrays, data processing using Numpy arrays.

Unit 3 (12 hours)

Data Manipulation using Pandas: Data Structures in Pandas: Series, DataFrame, Index objects, Loading data into Pandas data frame, Working with DataFrames: Arithmetics, Statistics, Binning, Indexing, Reindexing, Filtering, Handling missing data, Hierarchical indexing, Data wrangling: Data cleaning, transforming, merging and reshaping

Unit 4 (8 hours)

Plotting and Visualization: Using Matplotlib to plot data: figures, subplots, markings, color and line styles, labels and legends, Plotting functions in Pandas: Line, bar, Scatter plots, histograms, stacked bars, Heatmap

Unit 5 (7 hours)

Data Aggregation and Group operations: Group by mechanics, Data aggregation, General split-apply-combine, Pivot tables and cross tabulation

Essential/recommended readings

1. McKinney W. *Python for Data Analysis: Data Wrangling with Pandas, NumPy and IPython*, 2nd edition, O'Reilly Media, 2018.
2. Molin S. *Hands-On Data Analysis with Pandas*, Packt Publishing, 2019.
3. Gupta S.C., Kapoor V.K. *Fundamentals of Mathematical Statistics*,12th edition, Sultan Chand & Sons, 2020.

Additional References

1. Chen D. Y. *Pandas for Everyone: Python Data Analysis*, First edition, Pearson Education, 2018.
2. Miller J.D. *Statistics for Data Science,* Packt Publishing Limited, 2017.

Suggested Practical List (If any): (30 Hours)

Practical exercises such as

Use a dataset of your choice from Open Data Portal (https:// data.gov.in/, UCI repository) or load from scikit, seaborn library for the following exercises to practice the concepts learnt.

1. Load a Pandas dataframe with a selected dataset. Identify and count the missing values in a dataframe. Clean the data after removing noise as follows
   a) Drop duplicate rows.
   b) Detect the outliers and remove the rows having outliers
   c) Identify the most correlated positively correlated attributes and negatively correlated attributes

2. Import iris data using sklearn library or (Download IRIS data from: https://archive.ics.uci.edu/ml/datasets/iris or import it from sklearn.datasets)
    i. Compute mean, mode, median, standard deviation, confidence interval and standard error for each feature
    ii. Compute correlation coefficients between each pair of features and plot heatmap
    iii. Find covariance between length of sepal and petal
    iv. Build contingency table for class feature

3. Load Titanic data from sklearn library , plot the following with proper legend and axis labels:

    a. Plot bar chart to show the frequency of survivors and non-survivors for male and female passengers separately
    b. Draw a scatter plot for any two selected features
    c. Compare density distribution for features age and passenger fare
    d. Use a pair plot to show pairwise bivariate distribution

4. Using Titanic dataset, do the following
    a. Find total number of passengers with age less than 30
    b. Find total fare paid by passengers of first class
    c. Compare number of survivors of each passenger class

5. Download any dataset and do the following
    a. Count number of categorical and numeric features
    b. Remove one correlated attribute (if any)
    c. Display five-number summary of each attribute and show it visually

Project: Students are encouraged to work on a good dataset in consultation with their faculty and apply the concepts learned in the course.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DISCIPLINE SPECIFIC ELECTIVE COURSE: Microprocessors

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Microprocessors	**4**	**3**	**0**	**1**	**Pass in XII class**	Computer System Architecture

Learning Objectives

This course introduces the internal architecture, programming models of Intel Microprocessors (8086 - Pentium) and assembly language programming. Students will also learn interfacing of memory and I/O devices with microprocessors.

Learning outcomes

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

- Describe the internal architecture of Intel microprocessors.
- Define and implement interfaces between the microprocessor and the devices.
- Write assembly language programs.

SYLLABUS OF DSE

Unit 1 (5 hours)

Microprocessor Architecture: Internal Architecture, Programming Model, Addressing Modes, Data Movement Instructions

Unit 2 (7 hours)

Microprocessor programming: Register Organization, instruction formats, Program control instructions, assembly language.

Unit 3 (10 hours)

Interfacing: Bus timings, Memory address decoding, cache memory and cache controllers, I/O interface, keyboard, timer, Interrupt controller, DMA controller, video controllers, communication interfaces.

Unit 4 (7 hours)

Data transfer schemes: Synchronous data transfer, asynchronous data transfer, interrupt driven data transfer, DMA mode data transfer.

Unit 5 (8 hours)

Microprocessor controllers: I/O controllers, interrupt controller, DMA controller, USART controller.

Unit 6 (8 hours)

Advanced microprocessor architecture: CISC architecture, RISC architecture, superscalar architecture, multicore architecture.

Essential/recommended readings

1. Brey, B.B. *The Intel Microprocessors: Architecture, Programming and Interfacing,* 8th edition, Pearson education, 2009.

2. Triebel, W.A., & Singh, A. *The 8088 and 8086 Microprocessors Programming, Interfacing, Software, Hardware and Applications*, 4th edition, Pearson education, 2002.

Additional References

1. Ramesh S Gaonkar *Microprocessor architecture, programming, and applications with the 8085,* 6th edition, Penram International Publishing, 2013.

Suggested Practical List (If any): (30 Hours)

Practical exercises such as

ASSEMBLY LANGUAGE PROGRAMMING

1. Write a program to print 'Hello World'.
2. Write a program to print two strings on two different lines.
3. Write a program to take a single digit number from the user and print that number on the console.
4. Write a program to compare two single digit numbers and check if they are equal or not.
5. Write a program for 8-bit addition of two single digit numbers. Show the result after ASCII adjust.
6. Write a program for 16-bit addition of two double digit numbers. Show the result after ASCII adjust.
7. Write a program for 16-bit BCD addition.
8. Write a program for 32-bit BCD addition and subtraction.
9. Write a program for 32-bit Binary addition,subtraction,multiplication and division.
10. Write a program for Binary to ASCII conversion.
11. Write a program for ASCII to Binary conversion.
12. Write a program to take input in an array and print it on the console.
13. Write a program to sort an array using bubble sort.
14. Write a program to perform linear search in an array.
15. Write a program to perform binary search in an array.
16. Write a program to add and subtract two arrays.
17. write programs to interface a microprocessor with external devices such as a keyboard and elevator.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

(Computer Science Courses for Undergraduate Programme of study with Computer Science discipline as one of the three Core Disciplines)

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSE 01a PYTHON Programming for Data Handling	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objectives

The course introduces students to the concept of data handling using files and GUI designing. This would equip the students with knowledge to work on real world data from various applications and GUI development for effective data handling.

Learning outcomes

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

- Learn constructs of Python language
- Perform data handling with files using Python.
- Design and implement GUI applications using Tkinter.

SYLLABUS OF DSE 01a

Unit 1 (15 Hours)

Introduction to Python Programming, Basic Constructs, and Python Built-in Data Structures: Introduction to Python programming language, Basic syntax, variables, and data

types in Python, Functions and modular programming; Conditional statements (if, elif, else); Looping structures (for and while loops); Mutable and Immutable Data Structures, Strings-Indexing, slicing, traversal, operations; Lists-indexing, slicing, traversal, operations; tuples, dictionaries, and sets and their operations in Python

Unit 2 (5 Hours)

File Handling: Opening, reading, writing, and closing files; File modes and file object methods; Reading and writing text and binary files; Working with CSV files

Unit 3 (15 Hours)

Designing GUI Applications with Tkinter (15): What is Tkinter? Creating a Tkinter window, Layout managers, Tkinter widgets -Entry, Spinbox, Combobox, Checkbutton, Text, Button, LabelFrame; Implementing the application - LabelInput class, building of form, adding LabelFrame and other widgets, retrieving data from form, resetting form, building our application class.

Unit 4 (10 Hours)

Combining Python file handling and Tkinter: Creating a simple Tkinter application, Reading and writing to csv files in a Tkinter application

Essential/recommended readings

1. Taneja S., Kumar, N. Python Programming- A modular approach, 1st Edition, Pearson Education India, 2018,

2. Moore, Alan D. Python GUI Programming with Tkinter: Develop responsive and powerful GUI applications with Tkinter. Packt Publishing Ltd, 2021.

Additional References:

1. Guttag, J.V. Introduction to computation and programming using Python, 2nd edition, MIT

Online references/material:

1. https://docs.python.org/3/library/csv.html

Suggested Practical List (If any): (30 Hours)

Installing and setting up Python and relevant libraries; Python development environments (e.g., Anaconda, Jupyter Notebook)

1. Write a Python program to calculate the factorial of a number.
2. Write a Python program to generate prime numbers between 1 to n, where n is provided as input by the user.
3. Write a Python program to find the sum and average of numbers in a given list.
4. Given two sets, set1 and set2, write a Python program to find their union, intersection and difference.
5. Given a list of numbers, write a Python program to count the number of times an element occurs in a list and create a dictionary with *element:count* as *key:value* pairs.
6. Write a Python program to swap the first two and last two characters in a given string.
7. Write a Python program to create a text file having names of ten Indian cities.
8. Write a Python program to create a text file having atleast five lines about your college using `writelines()` function.
9. Write a Python program which reads the data from three input files having Employee Names and merges them into one output file.
10. Write a Python program to count the number of vowels in a file and write the *vowel : count* in a dictionary.
11. Write a Python program to create a CSV file having student data: RollNo, Enrollment No, Name, Course, Semester.
12. Write a Python program library to read the CSV file created in the above program and filter out records of II semester students.
13. Write a Python program using tkinter library to create a GUI to enter registration details for an event.
14. Write a Python program using tkinter library to create a calculator to perform addition, subtraction, multiplication and division of two numbers entered by the user.
15. Write a Python program using tkinter library to create an age calculator to calculate age when DOB is entered.
16. Write a Python program using tkinter library to read and write student data to and from a CSV file (refer question 11).

Note: **Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.**

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Android Programming using Java	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objective

The course enables the students to understand Android architecture and its key features, making them competent to develop Android applications using Java.

Learning outcomes

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

- logically organize Java classes and interfaces using packages.
- understand the design of the Android operating system.
- design user interfaces using various dialog boxes, menus, etc.
- design Android applications with interaction among various activities/applications.

SYLLABUS OF DSE 01b

Unit 1 (15 hours)
Review of Object Oriented Programming and Java Fundamentals: Structure of Java programs, classes and objects, data types, type casting, looping constructs, inheritance.

Unit 2 (2 hours)
Interfaces: Interface basics, defining, implementing and extending interfaces.

Unit 3 (4 hours)
Packages: Basics of packages, creating and accessing packages.

Unit 4 (7 hours)
GUI Programming: AWT classes, event handling.

Unit 5 (5 hours)

Introduction to Android Programming: Introduction to Android Operating System, Android SDK, AVD, components of an Android Application, parcels, and bundles.

Unit 6 (6 hours)

User Interface Architecture: Android Architecture, Contexts in Android, Intents and Intent Filters, Activity Life Cycle, Activity Stack, Fragments, and Fragments Life Cycle.

Unit 7 (6 hours)

User Interface Design: Android Layouts, Views, Spinner, Menu, Toggle Buttons, Radio Buttons, Check Boxes, Alert Box, and Toasts.

Essential/recommended readings

1. Schildt H. Java: The Complete Reference. 12th edition. McGraw-Hill Education, 2021
2. Griffiths D. & Griffiths D. Head First Android Development. O'Reilly, 2017
3. Meier R. Professional AndroidTM 4 Application Development. John Wiley & Sons, Inc., 2012

Additional Resources:

1. Horstmann, C. S. Core Java - Vol. I – Fundamentals. 12th edition. Pearson Education, 2021
2. Murphy M. L. The Busy Coder's Guide to Android Development. CommonsWare, 2018
3. Phillips B., Stewart C., Hardy B. & Marsicano K. Android Programming: The Big Nerd Ranch Guide. Big Nerd Ranch, LLC, 2015
4. Sheusi J. C. Android Application Development for Java Programmers. Cengage Learning, 2013

Suggested Practical List (If any): (30 Hours)

1. Write a function to find whether a number is prime or not. Use this function to determine the nth prime number. Read n from the user.

2. Design a class Complex having a real part (x) and an imaginary part (y). Provide methods to perform the following on complex numbers:

    a. Add two complex numbers.

b. Multiply two complex numbers.

c. toString() method to display complex numbers in the form: x + i y

3. Create a class TwoDim which contains private members as x and y coordinates in package P1. Define the default constructor, a parameterized constructor and override toString() method to display the co-ordinates. Now reuse this class and in package P2 create another class ThreeDim, adding a new dimension as z as its private member. Define the constructors for the subclass and override toString() method in the subclass also. Write appropriate methods to show dynamic method dispatch. The main() function should be in a package P.

4. Write a program to create an Applet. Create a frame as a child of an applet. Implement mouseClicked( ), mouseEntered( ) and mouseExited( ) events for the applet. Frame is visible when mouse enters applet window and hidden when mouse exits from the applet window.

5. Write a program to display a string in a frame window with pink color as background.

6. Write a program to create an Applet that has two buttons named “Red” and “Blue”. When a button is pressed, the background color of the applet is set to the color named by the button’s label.

7. Create a “Hello World” application. That will display “Hello World” in the middle of the screen in the emulator. Also display “Hello World” in the middle of the screen in the Android Phone.

8. Create an Android application with a login module. (Check username and password).

9. Create a Spinner with strings taken from resource folder (res >> value folder) and on changing the spinner value, Image will change.

10. Create a Menu with 5 options and a selected option should appear in the text box.

11. Create an application with three option buttons, on selecting a button colour of the screen will change.

12. Create an Application to display various Activity and Fragment Life Cycle Methods.

13. Create an application with 2 fragments, one to set the background and other to set the fore-color of the text.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

COMMON POOL OF GENERIC ELECTIVES (GE) COURSES

GENERIC ELECTIVES : Database Management Systems

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course	Department offering the course
		Lecture	Tutorial	Practical/ Practice			
Database Management Systems	**4**	**3**	**0**	**1**	**Pass in class XII**	NIL	**Computer Science**

Learning Objectives

The course introduces the students to the fundamentals of database management systems and their applications. Emphasis is given to the popular relational database system. Students will learn about the importance of database structure and its design using entity relationship diagrams and a formal approach using normalization. Basic concepts of file indexing and transaction processing will be taught. The course would give students hands-on practice with structured query language to create, manipulate, and implement a relational database.

Learning outcomes

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

- Use relational database management software to create and manipulate the database.
- Create conceptual data models using entity relationship diagrams for modeling real-life situations and map it to corresponding relational database schema.
- Use the concept of functional dependencies to remove redundancy and update anomalies.
- Apply normalization theory to get a normalized database scheme to get anomalies free databases.
- Write queries in relational algebra.
- Implement relational databases and formulate queries for data retrieval and data update problems using SQL.
- Learn the importance of index structures and concurrent execution of transactions in database systems.

SYLLABUS

Unit 1 (5 hours)

Introduction to Database: Database, characteristics of database approach, data models, database management system, three-schema architecture, components of DBMS, data independence, and file system approach vs. database system approach

Unit 2 (8 hours)

Entity Relationship Modeling: Conceptual data modeling - motivation, entities, entity types, attributes, relationships, relationship types, constraints on relationship, Entity Relationship diagram as conceptual data model.

Unit 3 (11 hours)

Relational Data Model: Data anomalies, Relational Data Model - Characteristics of a relation, schema-instance distinction, types of keys, relational integrity constraints. Relational algebra operators like selection, projection, cartesian product, join and write simple queries using them.

Unit 4 (10 hours)

Structured Query Language (SQL): DDL to create database and tables, table constraints, DML, Querying in SQL to retrieve data from the database, aggregation functions group by and having clauses, generate and query views.

Unit 5 (11 hours)

Database Design: Mapping an Entity Relationship diagram to corresponding relational database scheme, functional dependencies and Normal forms, 1NF, 2NF, and 3NF decompositions and desirable properties of them.

Essential/recommended readings

1. Elmasri, R., Navathe, B. S., *Fundamentals of Database Systems*, 7th Edition, Pearson Education, 2016.
2. Murach J., *Murach's MySQL,* 3th Edition, Pearson, 2019.

Additional References

1. Connolly T. M., Begg C. E. *Database Systems: A Practical Approach to Design, Implementation, and Management*, 6th edition, Pearson, 2019.
2. Ramakrishnan R., Gehrke J. *Database Management Systems*, 3rd Edition, McGraw-Hill, 2014.
3. Silberschatz A., Korth H.F., Sudarshan S. *Database System Concepts*, 7th Edition, McGraw Hill, 2019.

Suggested Practical List (if any): (30 hours)

Practical exercises based on a given schema.

Create and use the following student-course database schema for a college to answer the given queries using the standalone SQL editor.

STUDENT	**Roll No**	**Student Name**	**Course ID**	**DOB**
	Char(6)	Varchar(20)	Varchar(10)	Date

COURSE	**CID**	**Course Name**	**Course Type**	**Teacher-in-charge**	**Total Seats**	**Duration**
	Char(6)	Varchar (20)	Char (8)	Varchar (15)	Unsigned int	Unsigned int

ADMISSION	**Roll No**	**CID**	**Date of Admission**
	Char(6)	Char(6)	**Date**

Here, Rollno (ADMISSION) and SID (ADMISSION) are foreign keys. Note that course type may have two values viz. Fulltime and Parttime and a student may enroll in any number of courses

1. Retrieve names of students enrolled in any course.
2. Retrieve names of students enrolled in at least one part time course.
3. Retrieve students' names starting with letter 'A'.
4. Retrieve students' details studying in courses 'computer science' or 'chemistry'.
5. Retrieve students' names whose roll no either starts with 'X' or 'Z' and ends with '9'
6. Find course details with more than N students enrolled where N is to be input by the user.
7. Update student table for modifying a student name.
8. Find course names in which more than five students have enrolled
9. Find the name of youngest student enrolled in course 'BSc(P)CS'
10. Find the name of most popular society (on the basis of enrolled students)
11. Find the name of two popular part time courses (on the basis of enrolled students)
12. Find the student names who are admitted to full time courses only.
13. Find course names in which more than 30 students took admission
14. Find names of all students who took admission to any course and course names in which at least one student has enrolled
15. Find course names such that its teacher-in-charge has a name with 'Gupta' in it and the course is full time.
16. Find the course names in which the number of enrolled students is only 10% of its total seats.
17. Display the vacant seats for each course
18. Increment Total Seats of each course by 10%
19. Add enrollment fees paid ('yes'/'No') field in the enrollment table.
20. Update the date of admission for all the courses by 1 year.
21. Create a view to keep track of course names with the total number of students enrolled in it.

22. Count the number of courses with more than 5 students enrolled for each type of course.
23. Add column Mobile number in student table with default value '9999999999'
24. Find the total number of students whose age is > 18 years.
25. Find names of students who are born in 2001 and are admitted to at least one part time course.
26. Count all courses having 'science' in the name and starting with the word 'BSc'.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

GENERIC ELECTIVES : Java Programming

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course	Department offering the course
		Lecture	Tutorial	Practical/ Practice			
GE: Java Programming	**4**	**3**	**0**	**1**	**Pass in class XII**	NIL	**Computer Science**

Learning Objectives

This course is designed to develop understanding of object-oriented programming concepts like Classes, Objects, Inheritance and Polymorphism using Java. The course provides understanding of multithreading and exception handling in Java. It also introduces how to create Java applications with graphical user interface (GUI).

Learning outcomes

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

- Understand the object-oriented concepts – Classes, Objects, Inheritance, Polymorphism– for problem solving.
- Create and handle multithreading.
- Handle program exceptions.
- Handle input/output through files.
- Create Java applications with a graphical user interface (GUI).

SYLLABUS OF GE

Unit 1 (6 hours)

Introductory Concepts: program, identifiers, variables, constants, primitive data types, expressions, Naming Conventions, Type casting, operators, control statements, structured data types, arrays, functions.

Unit 2 (13 hours)

Object Oriented Concepts: Abstraction, encapsulation, objects, classes, methods, constructors, inheritance, polymorphism, static and dynamic binding, Anonymous block, Static Data members, overloading and overriding, Usage of super and this keyword, Abstract classes, Interfaces and Packages, Access modifiers, Object class

Unit 3 (11 hours)

Multithreading: Creating Threads, Thread Priority, Blocked States, Extending Thread Class, Runnable Interface, Starting Threads, Thread Synchronization, Sync Code Block, Overriding Synced Methods, Thread Communication, wait, notify and notify all.

Unit 4 (8 hours)

Introduction to Exception handling: Exception and Error, Throw, try and catch Blocks, Exception handlers, java.lang Exceptions, Built-InExceptions.

Unit 5 (7 hours)

Introduction to File Handling: Byte Stream, Character Stream, File I/O Basics, File Operations, Serialization.

Essential/recommended readings

1. Cay S. Horstmann, *Core Java - Vol. I – Fundamentals*, 10th edition, Pearson, 2017.
2. James Gosling, Bill Joy, Guy L. Steele Jr, Gilad Bracha, Alex Buckley, *The Java Language Specification*, *Java SE* 7th edition, Addison-Wesley, 2011

Additional References

1. Herbert Schildt, *Java: The Complete Reference*, 10th edition, McGraw-Hill Education, 2018.

2 Richard Johnson, *An Introduction to Java Programming and Object-Oriented Application Development*, Thomson Learning, 2006.

3. Kathy Sierra and Bert Bates, *Head First Java*, 3rd edition, O'Reilly, 2022.

Suggested Practical List (If any): (30 Hours)

Practical exercises such as

1. Create a java program to implement stack and queue concepts.
2. Write a program to take input from command line arguments.
3. Write a java program to show static and dynamic polymorphism.
4. Write a java program to show multiple inheritance using interfaces.
5. Write a program in java to show the chaining of execution of construction.
6. Write a java program to show multithreaded producer and consumer applications.
7. write a program in java to synchronize the multithreaded application
8. Create a customized exception and also make use of all the exception keywords.
9. Write a program to show different ways to get input from user
10. Design a form using AWT components and the Frame container.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

SEMESTER-4

DEPARTMENT OF COMPUTER SCIENCE
[UG Programme for Bachelor in Computer Science (Honours)]

DISCIPLINE SPECIFIC CORE COURSE - 10 (DSC-10) : Design and Analysis of Algorithms

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC 10 Design and Analysis of Algorithms	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 07 Data Structures with C++

Learning Objectives

The course is designed to develop understanding of different algorithm design techniques and use them for problem solving. The course shall also enable the students to verify correctness of algorithms and analyze their time complexity.

Learning outcomes

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

- Compute and compare the asymptotic time complexity of algorithms.
- Prove correctness of algorithms.
- Use appropriate algorithm design technique(s) for solving a given problem.
- Distinguish between tractable and intractable problems.

Unit 1 (10 hours)

Searching, Sorting, Selection: Linear Search, Binary Search, Insertion Sort, Selection Sort, Bubble Sort, Heapsort, Linear Time Sorting, Selection Problem, running time analysis and correctness.

Unit 2 (5 hours)

Graphs: Review of graph traversals**,** graph connectivity, testing bipartiteness, Directed Acyclic Graphs and Topological Ordering.

Unit 3 (10 hours)

Divide and Conquer: Introduction to divide and conquer technique, Merge Sort, Quick Sort, Maximum-subarray problem, Strassen's algorithm for matrix multiplication.

Unit 4 (5 hours)

Greedy algorithms: Introduction to the Greedy algorithm design approach, application to minimum spanning trees, fractional knapsack problem, etc. with correctness, and analysis of time complexity.

Unit 5 (5 hours)

Dynamic Programming: Introduction to the Dynamic Programming approach, application to subset sum, integer knapsack problem etc., correctness, and analysis of time complexity.

Unit 6 (5 hours)

Intractability: Concept of polynomial time computation, polynomial time reductions, decision vs optimization problems, Introduction to NP, NP-hard and NP-Complete classes.

Unit 7 (5 hours)

Advanced Analysis of Algorithms: Amortized Analysis.

Essential/recommended readings

1. Cormen, T.H., Leiserson, C.E., Rivest, R. L., Stein C. *Introduction to Algorithms*, 4th edition, Prentice Hall of India, 2022.
2. Kleinberg, J., Tardos, E. *Algorithm Design,* 1st edition, Pearson, 2013.

Additional references

1. Basse, S., Gelder, A. V., *Computer Algorithms: Introduction to Design and Analysis,* 3rd edition, Pearson, 1999.

Practical List (If any): (30 Hours)

1. i. Write a program to sort the elements of an array using Insertion Sort (The program should report the number of comparisons).
   ii. Write a program to sort the elements of an array using Merge Sort (The program should report the number of comparisons).
2. Write a program to sort the elements of an array using Heap Sort (The program should report the number of comparisons).
3. Write a program to multiply two matrices using the Strassen's algorithm for matrix multiplication.

4. Write a program to sort the elements of an array using Radix Sort.
5. Write a program to sort the elements of an array using Bucket Sort.
6. Display the data stored in a given graph using the Breadth-First Search algorithm.
7. Display the data stored in a given graph using the Depth-First Search algorithm.
8. Write a program to determine a minimum spanning tree of a graph using the Prim's algorithm.
9. Write a program to implement Dijkstra's algorithm to find the shortest paths from a given source node to all other nodes in a graph.
10. Write a program to solve the weighted interval scheduling problem.
11. Write a program to solve the 0-1 knapsack problem.

For the algorithms at S.No 1 and 2, test run the algorithm on 100 different input sizes varying from 30 to 1000. For each size find the number of comparisons averaged on 10 different input instances; plot a graph for the average number of comparisons against each input size. Compare it with a graph of nlogn.

DISCIPLINE SPECIFIC CORE COURSE – 11 (DSC11): Database Management Systems

Credit distribution, Eligibility and Prerequisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC 11 Database Management Systems	**4**	**3**	**0**	**1**	Pass in Class XII	DSC01 Programming using Python / A course in Python at plus 2 level, DSC08

Learning Objectives

The course introduces the students to the fundamentals of database management system and its architecture. Emphasis is given on the popular relational database system including data models and data manipulation. Students will learn about the importance of database structure and its designing using conceptual approach using Entity Relationship Model and formal approach using Normalization. The importance of file indexing and controlled execution of transactions will be taught. The course would give students hands-on practice of structured query language in a relational database management system and glimpse of basic database administration commands.

Learning outcomes

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

- Use database management system software to create and manipulate the database.
- Create conceptual data models using entity relationship diagrams for modeling real-life situations and designing the database schema.
- Use the concept of functional dependencies to remove redundancy and update anomalies.
- Apply normalization theory to get a normalized database scheme.
- Write queries using relational algebra, a procedural language.

SYLLABUS OF DSC11

Unit 1 (5 hours)

Introduction to Database: Purpose of database system, Characteristics of database approach, data models, database management system, database system architecture, three-schema architecture, components of DBMS, data independence, and file system approach vs database system approach.

Unit 2 (7 hours)

Entity Relationship Modeling: Conceptual data modeling - motivation, entities, entity types, attributes, relationships, relationship types, constraints on relationship, Entity Relationship diagram notation.

Unit 3 (7 hours)

Relational Data Model: Update anomalies, Relational Data Model - Concept of relations, schema-instance distinction, keys, relational integrity constraints, referential integrity and foreign keys, relational algebra operators and queries.

Unit 4 (10 hours)

Structured Query Language (SQL): Querying in SQL, DDL to create database and tables, table constraints, update database-update behaviors, DML, aggregation functions group by

and having clauses, retrieve data from the database, generate and query views. Access and manipulate databases using ODBC. Basic Database administration SQL commands.

Unit 5 (8 hours)

Database Design: Mapping an Entity Relationship model to relational database, functional dependencies and Normal forms, 1NF, 2NF, 3NF and BCNF decompositions and desirable properties of them.

Unit 6 (8 hours)

File indexing and Transaction Processing: Data Storage and Indexes- Need of file indexes, file organizations, index structures, single- and multi-level indexing, concurrent execution of transactions, ACID properties, need of data recovery and log file.

Essential/recommended readings

1. Elmasri, R., Navathe, B. S. *Fundamentals of Database Systems*, 7th Edition, Pearson Education, 2015.
2. Krogh, J. W. *MySQL Connector/Python Revealed: SQL and NoSQL Data Storage Using MySQL for Python Programmers*, Apress, 2018.
3. Murach J. *Murach's MySQL,* 3rd edition, Pearson, 2019.

Additional References

1. Ramakrishnan, R., Gehrke J. *Database Management Systems*, 3rd Edition, McGraw-Hill, 2014.
2. Silberschatz, A., Korth, H. F., Sudarshan S. *Database System Concepts*, 7th Edition, McGraw Hill, 2019.
3. Connolly, T. M., Begg, C. E. *Database Systems: A Practical Approach to Design, Implementation, and Management*, 6th edition, Pearson, 2019.

Suggested Practical List (If any): (30 Hours)

Practical exercises such as

It has three components.

I. Create and use the following student-society database schema for a college to answer the given (sample) queries using the standalone SQL editor.

STUDENT	**Roll No**	**StudentName**	**Course**	**DOB**
	Char(6)	Varchar(20)	Varchar(10)	Date

SOCIETY	SID	SocName	MentorName	TotalSeats
	Char(6)	Varchar(20)	Varchar(15)	Unsigned int

ENROLLMENT	Roll No	SID	DateOfEnrollment
	Char(6)	Char(6)	Date

Here Rollno (ENROLLMENT) and SID (ENROLLMENT) are foreign keys.

1. Retrieve names of students enrolled in any society.
2. Retrieve all society names.
3. Retrieve students' names starting with letter ‘A’.
4. Retrieve students' details studying in courses ‘computer science’ or ‘chemistry’.
5. Retrieve students’ names whose roll no either starts with ‘X’ or ‘Z’ and ends with ‘9’
6. Find society details with more than N TotalSeats where N is to be input by the user
7. Update society table for mentor name of a specific society
8. Find society names in which more than five students have enrolled
9. Find the name of youngest student enrolled in society ‘NSS’
10. Find the name of most popular society (on the basis of enrolled students)
11. Find the name of two least popular societies (on the basis of enrolled students)
12. Find the student names who are not enrolled in any society
13. Find the student names enrolled in at least two societies
14. Find society names in which maximum students are enrolled
15. Find names of all students who have enrolled in any society and society names in which at least one student has enrolled
16. Find names of students who are enrolled in any of the three societies ‘Debating’, ‘Dancing’ and ‘Sashakt’.
17. Find society names such that its mentor has a name with ‘Gupta’ in it.
18. Find the society names in which the number of enrolled students is only 10% of its capacity.
19. Display the vacant seats for each society.
20. Increment Total Seats of each society by 10%
21. Add enrollment fees paid (‘yes’/’No’) field in the enrollment table.
22. Update date of enrollment of society id ‘s1’ to ‘2018-01-15’, ‘s2’ to current date and ‘s3’ to ‘2018-01-02’.
23. Create a view to keep track of society names with the total number of students enrolled in it.
24. Find student names enrolled in all the societies.
25. Count number of societies with more than 5 student enrolled in it
26. Add column Mobile number in student table with default value ‘9999999999’
27. Find the total number of students whose age is > 20 years.
28. Find names of students who are born in 2001 and are enrolled in at least one society.
29. Count all societies whose name starts with ‘S’ and ends with ‘t’ and at least 5 students are enrolled in the society.
30. Display the following information:

Society name Mentor name Total Capacity Total Enrolled Unfilled Seats

II. Do the following database administration commands:
create user, create role, grant privileges to a role, revoke privileges from a role, create index

II. Execute queries given in part I through a high-level language using ODBC connection.

DISCIPLINE SPECIFIC CORE COURSE– 12 (DSC-12): Computer Networks

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits				Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC12 Computer Networks	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 04 Object Oriented Programming with C++/ GE 1a Programming using C++ / GE1b Programming with Python/ DSC 01 Programming using Python/ GE 3b: Java Programming

Learning Objectives
The course objectives of this paper are to:

- Understand the concepts behind computer networks and data communication.
- Learn the different types of networks, network topologies and their characteristics.
- Learn the working of protocols used at various layers.
- Understand the utility of different networking devices.

Learning outcomes

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

- differentiate between various types of computer networks and their topologies.
- understand the difference between the OSI and TCP/IP protocol suit.

- distinguish between different types of network devices and their functions.
- design/implement data link and network layer protocols in a simulated networking environment.

SYLLABUS OF DSC06

Unit 1 (8 hours)
Introduction: Types of computer networks, Internet, Intranet, network topologies (bus, star, ring, mesh, tree, hybrid topologies), network classifications. layered architecture approach, OSI Reference Model, TCP/IP Reference Model. Transmission Modes: simplex, half duplex and full duplex.

Unit 2 (9 hours)
Physical Layer: Analog signal, digital signal, the maximum data rate of a channel, transmission media (guided transmission media, wireless transmission, satellite communication), multiplexing (frequency division multiplexing, time-division multiplexing, wavelength division multiplexing). Guided Media (Wired) (Twisted pair, Coaxial Cable, Fiber Optics. Unguided Media (Radio Waves, Infrared, Micro-wave, Satellite).

Unit 3 (10 hours)
Data Link and MAC Layer: Data link layer services, error detection and correction techniques, error recovery protocols (stop and wait, go back n, selective repeat), multiple access protocols with collision detection, MAC addressing, Ethernet, data link layer switching, point-to-point protocol.

Unit 4 (8 hours)
Network layer: Networks and Internetworks, virtual circuits and datagrams, addressing, subnetting, Dijkstra Routing algorithm, Distance vector routing, Network Layer protocol- (ARP, IPV4, ICMP).

Unit 5 (10 hours)
Transport and Application Layer: Process to process Delivery- (client-server paradigm, connectionless versus connection-oriented service); User Datagram Protocols, TCP/IP protocol, Flow Control. FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol), Telnet (Remote login protocol), WWW (World Wide Web), HTTP (HyperText Transfer Protocol), URL (Uniform Resource Locator).

Essential/recommended readings

1. Tanenbaum, A.S. & Wethrall, D.J.. *Computer Networks*, 5th edition, Pearson Education, 2012.

2. Forouzan, B. A.. *Data Communication and Networking*, 4th edition, McGraw-Hill Education, 2017.

Additional References

1. Comer, D. E.. *Computer Networks and Internet,* 6th edition, Pearson education, 2015.

2. Stallings, W., *Data and Computer Communications,* 10th edition, Pearson education India, 2017.

Suggested Practical List :

Practical exercises such as

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

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

Computer Science Courses for Undergraduate Programme of study with Computer Science discipline Elective

DISCIPLINE SPECIFIC ELECTIVE COURSE: Data Mining - I

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Data Mining - I	**4**	**3**	**0**	**1**	Pass in Class XII	DSC01 Programming using Python / GE1b Programming with Python

Learning Objectives

This course aims to introduce data mining techniques and their application on real-life datasets. The students will learn to pre-process the dataset and make it ready for application of data mining techniques. The course will focus on three main techniques of data mining i.e. Classification, Clustering and Association Rule Mining. Different algorithms for these techniques will be discussed along with appropriate evaluation metrics to judge the performance of the results delivered.

Learning outcomes

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

1. Pre-process the data for subsequent data mining tasks
2. Apply a suitable classification algorithm to train the classifier and evaluate its performance.
3. Apply appropriate clustering algorithm to cluster the data and evaluate clustering quality
4. Use association rule mining algorithms and generate frequent item-sets and association rules

SYLLABUS OF DSE

Unit 1 (5 hours)

Introduction to Data Mining: Motivation and challenges for data mining, Types of data mining tasks, Applications of data mining, Data measurements, Data quality, Supervised vs. unsupervised techniques

Unit 2 (10 hours)

Data Pre-processing: Data aggregation, sampling, dimensionality reduction, feature subset selection, feature creation, variable transformation.

Unit 3 (10 hours)

Cluster Analysis: Basic concepts of clustering, measure of similarity, types of clusters and clustering methods, K-means algorithm, measures for cluster validation, determine optimal number of clusters

Unit 4 (10 hours)

Association Rule mining: Transaction data-set, frequent itemset, support measure, rule generation, confidence of association rule, Apriori algorithm, Apriori principle

Unit 5 (10 hours)

Classification: Naive Bayes classifier, Nearest Neighbour classifier, decision tree, overfitting, confusion matrix, evaluation metrics and model evaluation.

Essential/recommended readings

1. Tan P.N., Steinbach M, Karpatne A. and Kumar V. *Introduction to Data Mining,* 2nd edition, Pearson, 2021.
2. Han J., Kamber M. and Pei J. *Data Mining: Concepts and Techniques*, 3rd edition, 2011, Morgan Kaufmann Publishers.
3. Zaki M. J. and Meira J. Jr. *Data Mining and Machine Learning: Fundamental Concepts and Algorithms*, 2nd edition, Cambridge University Press, 2020.

Additional References

1. Aggarwal C. C. *Data Mining: The Textbook*, Springer, 2015.
2. Dunham M. *Data Mining: Introductory and Advanced Topics,* 1st edition, Pearson Education India, 2006.

Suggested Practical List (If any): (30 Hours)

Practical exercises such as

All topics covered in theory will be implemented using Python. The operations may be performed on the datasets loaded through scikit, seaborn libraries or can be downloaded from from Open Data Portal (https:// data.gov.in/, UCI repository http://archive.ics.uci.edu/ml/).

Recommended Datasets for :

Classification: Abalone, Artificial Characters, Breast Cancer Wisconsin (Diagnostic)

Clustering: Grammatical Facial Expressions, HTRU2, Perfume data

Association Rule Mining: MovieLens, Titanics

Additional Suggested Practicals List

1. Apply data cleaning techniques on any dataset (e,g, wine dataset). Techniques may include handling missing values, outliers, inconsistent values. A set of validation rules can be prepared based on the dataset and validations can be performed.
2. Apply data pre-processing techniques such as standardization / normalization, transformation, aggregation, discretization/binarization, sampling etc. on any dataset
3. Run Apriori algorithm to find frequent itemsets and association rules on 2 real datasets and use appropriate evaluation measures to compute correctness of obtained patterns
    a) Use minimum support as 50% and minimum confidence as 75%
    b) Use minimum support as 60% and minimum confidence as 60 %

4. Use Naive bayes, K-nearest, and Decision tree classification algorithms and build classifiers on any two datasets. Divide the data set into training and test set. Compare the accuracy of the different classifiers under the following situations:
    I. a) Training set = 75% Test set = 25% b) Training set = 66.6% (2/3rd of total), Test set = 33.3%
    II. Training set is chosen by i) hold out method ii) Random subsampling iii) Cross-Validation. Compare the accuracy of the classifiers obtained.

    Data is scaled to standard format.
5. Use Simple K-means algorithm for clustering on any dataset. Compare the performance of clusters by changing the parameters involved in the algorithm. Plot MSE computed after each iteration using a line plot for any set of parameters.

Project: Students should be promoted to take up one project on any UCI/kaggle/data.gov.in or a dataset verified by the teacher. Preprocessing steps and at least one data mining technique should be shown on the selected dataset. This will allow the students to have a practical knowledge of how to apply the various skills learnt in the subject for a single problem/project.

DISCIPLINE SPECIFIC ELECTIVE COURSE: Combinatorial Optimization

Credit distribution, Eligibility and Pre-requisites of the Course

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

Learning Objectives

This course is designed to introduce the fundamentals of combinatorial optimization to the students in terms of both theory and applications, so as to equip them to explore the more advanced areas of convex and non-convex optimizations.

Learning outcomes

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

- Model problems using linear and integer programs
- Apply polyhedral analysis to develop algorithms for optimization problems
- Use the concept of duality for design of algorithms

SYLLABUS OF DSE

Unit 1 (9 hours)

Introduction: Introduction to Combinatorial Optimization Problems, Linear and Integer Programs- LP Formulation, understanding integer programs, computational complexities of IP vs LP, using LP to find optimal or approximate integral solutions, concept of integrality gap.

Unit 2 (14 hours)

Theory of Linear Programming and Algorithmic Perspective to Simplex Method: standard vs. equational form, basic feasible solutions, convexity and convex polyhedra, correspondence between vertices and basic feasible solutions, geometry of Simplex algorithm, exception handling (unboundedness, degeneracy, infeasibility), Simplex algorithm, avoiding cycles.

Unit 3 (12 hours)

Primal-Dual Algorithms: interpretation of dual, optimality conditions for primal and dual, weak and strong duality, complementary slackness, primal-dual algorithm for the shortest path problem.

Unit 4 (10 hours)

Network Flows: linear programming formulations for network flows and bipartite matching, totally unimodular matrices.

Essential/recommended readings

2. Papadimitriou, C.H. & Steiglitz, K. *Combinatorial Optimization: Algorithms and complexity***,** New edition, Dover Publications inc., 2000.

Additional References

(i) Bazaraa, M.S., Jarvis, J.J., & and Sherali, H.D. *Linear Programming and Network Flows*, 4th edition, Wiley, 2010.
(ii) Korte, B., & Vygen, J. *Combinatorial Optimization,* 6th edition, Springer, 2018.

Tutorials

Tutorials based on Theory

DISCIPLINE SPECIFIC ELECTIVE COURSE: Network Security

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Network Security		**3**	**1**	**0**	Pass in Class XII	DSC 04 Object Oriented Programming with C++/ GE 1a Programming using C++ / GE1b Programming with Python/ DSC 01 Programming using Python/ GE 3b: Java Programming

Learning Objectives

This course will provide students with an understanding of the fundamental concepts, principles, and techniques of network security. Students will learn how to assess, design, and implement secure networks using various tools and technologies.

Learning outcomes

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

- Describe the importance of network security and the principles of the CIA triad (confidentiality, integrity, and availability), types of security threats and attacks
- Describe the basics of cryptography, including symmetric and asymmetric encryption, hash functions, digital signatures, and public key infrastructure (PKI).
- Apply authentication and access control techniques, including password-based, token-based, and biometric authentication, as well as authorization models and single sign-on (SSO).
- Design and implement secure networks using network segmentation, security zones, and VPNs for remote access.

- Implement and manage firewalls, intrusion detection systems (IDS), and intrusion prevention systems (IPS) to protect network resources, secure wireless networks,
- Implement endpoint security and malware protection measures, including antivirus, patch management, and host-based firewalls.

SYLLABUS OF DSE

Unit 1 (10 hours)

Introduction to Network Security and Network Fundamentals: Importance of network security, Confidentiality, integrity, and availability (CIA) triad, Types of security threats and attacks. OSI and TCP/IP models. IP addressing and subnetting. Networking devices (hubs, switches, routers, firewalls). Network protocols and services (HTTP, HTTPS, FTP, SSH, etc.).

Unit 2 (12 hours)

Cryptography Basics, Authentication and Access Control, Secure Network Design: Symmetric and asymmetric encryption, Hash functions and digital signatures, Public key infrastructure (PKI), Common cryptographic algorithms (AES, RSA, SHA, etc.). Authentication techniques (passwords, tokens, biometrics), Authorization and access control models (RBAC, ABAC, MAC, DAC), Single sign-on (SSO) and multi-factor authentication (MFA). Defense-in-depth strategy, Network segmentation and isolation, Security zones and DMZ, VPNs and secure remote access.

Unit 3 (12 hours)

Firewalls and Intrusion Detection/Prevention Systems, Wireless Network Security: Types of firewalls (packet filtering, stateful inspection, application layer), IDS and IPS concepts and deployment, Signature-based and anomaly-based detection, Honeypots and honeynets. Wireless standards and technologies (802.11, Bluetooth, RFID), Wireless security protocols (WEP, WPA, WPA2, WPA3), Rogue access points and wireless attacks, Securing wireless networks.

Unit 4 (8 hours)

Endpoint Security and Malware Protection, Security Monitoring and Incident Response: Antivirus and antimalware solutions, Patch management and software updates, Host-based firewalls and intrusion detection, Mobile device management (MDM). Security Information and Event Management (SIEM) systems, Log management and analysis, Incident response process and procedures, Forensic analysis and evidence handling.

Unit 5 (3 hours)

Network Security Best Practices and Compliance: Security policies and procedures, Risk assessment and management, Security awareness training, Regulatory compliance (HIPAA, GDPR, PCI-DSS, etc.).

Essential/recommended readings

1. Behrouz Forouzan, Cryptography and network security. 3rd edition (2015), McGraw Hill Education.

2. Stallings, W. (2021). Cryptography and Network Security: Principles and Practice (8th Edition). Pearson.
3. Harris, S. (2018). All-in-One CISSP Exam Guide (8th Edition). McGraw-Hill Education.
4. Atul Kahate, Cryptography and Network Security, McGraw-Hill; Fourth edition (8 May 2019); McGraw Hill Education (India).

Additional References

i. Conklin, W. A., White, G., Williams, D., Davis, R., & Cothren, C. (2021). Principles of Computer Security: CompTIA Security+ and Beyond (6th Edition). McGraw-Hill Education.
ii. Chapple, M., & Seidl, D. (2020). Network Security For Dummies. Wiley.
iii. Gibson, D. (2021). CompTIA Security+ Get Certified Get Ahead: SY0-601 Study Guide. YCDA Publishing.

Online Additional Reference Materials:

1. NIST Special Publications: https://csrc.nist.gov/publications/sp
   a. SP 800-53: Security and Privacy Controls for Federal Information Systems and Organizations
   b. SP 800-82: Guide to Industrial Control Systems (ICS) Security
   c. SP 800-115: Technical Guide to Information Security Testing and Assessment
2. ISO/IEC 27000 series: Information Security Management Systems (ISMS)
   a. ISO/IEC 27001: Information Security Management
   b. ISO/IEC 27002: Code of Practice for Information Security Controls
   c. ISO/IEC 27005: Information Security Risk Management
3. Center for Internet Security (CIS) Critical Security Controls: https://www.cisecurity.org/controls/
   a. A prioritized set of actions to improve network security.
4. OWASP Top Ten Project: https://owasp.org/www-project-top-ten/
   a. A list of the most critical web application security risks.
5. SANS Institute Reading Room: https://www.sans.org/reading-room/
   a. A collection of whitepapers and articles on various network security topics.
6. Vendor documentation and best practices guides (Cisco, Juniper, Palo Alto Networks, etc.)

DISCIPLINE SPECIFIC ELECTIVE COURSE: Introduction to Web Programming

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Introduction to web programming	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objectives

The course aims at introducing the basic concepts and techniques of client side web programming. The student shall be able to develop simple websites using HTML, CSS and Javascript.

Learning outcomes

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

- Build websites using the elements of HTML.
- Build dynamic websites using the client side programming techniques with CSS, Javascript and jQuery.
- Learn to validate client-side data.

SYLLABUS OF DSE

Unit 1 (5 hours)

Introduction: Introduction to Internet and web design. Basic concepts of web architecture.

Unit 2 (12 hours)

HTML: Introduction to hypertext mark-up language (html), creating web pages, lists, hyperlinks, tables, web forms, inserting images, frames.

Unit 3 (8 hours)

Cascading style sheet (CSS): Concept of CSS, creating style sheet, Importing style sheets, CSS properties, CSS styling (background, text format, controlling fonts), CSS rules, Style Types, CSS Selectors, CSS cascade, working with block elements and objects, working with lists and tables, CSS id and class, box model (introduction, border properties, padding

properties, margin properties).

Unit 4 (10 hours)

Javascript: Document object model, data types and variables, functions, methods and events, controlling program flow, JavaScript object model, built-in objects and operators, validations.

Unit 5 (10 hours)

jQuery and JSON: Introduction to jQuery, syntax, selectors, events. JSON file format for storing and transporting data.

Essential/recommended readings

1. Nixon, R. *Learning PHP, MySQL & JavaScript with jQuery, CSS and HTML5*, O'Rielly, 2018.
2. Powell, T.A. *HTML & CSS: The Complete Reference, 5th edition*, Tata McGrawHill, 2010.
3. Duckett, J. *JavaScript and JQuery: Interactive Front-End Web Development*, Wiley, 2014.

Additional References

1. Minnick, J. *Web Design with HTML5 and CSS3*, 8th edition, Cengage Learning, 2015.
2. Boehm, A., & Ruvalcaba, Z. *Munarch's HTML5 and CCS,* 4th edition, Mike Murach & Associates, 2018.
3. J. A. Ramalho *Learn Advanced HTML 4.0 with DHTML*, BPB Publications, 2007.
4. Ivan Bayross *Web Enabled Commercial Application Development Using Html, Dhtml, Javascript, Perl CGI*, BPB Publications, 2009.

Suggested Practical List (If any): (30 Hours)

Practical exercises such as

HTML

1. Create an HTML document with following formatting – Bold, Italics, Underline, Colors, Headings, Title, Font and Font Width, Background, Paragraph, Line Brakes, Horizontal Line, Blinking text as well as marquee text.
2. Create an HTML document with Ordered and Unordered lists, Inserting Images, Internal and External linking
3. Create an HTML displaying this semester's time table.
4. Create a website with horizontal and vertical frames. Top horizontal frame showing your college's name and logo. Bottom horizontal frame split into two vertical frames. The left frame with hyperlinks to pages related to faculty, courses, student activities, etc. The right frame showing corresponding pages based on the link clicked on the left frame.
5. Create a student registration form using HTML which has the following controls:

I. Text Box
II. Dropdown box
III. Option/radio buttons
IV. Check boxes
V. Reset and Submit button

CSS

Create a webpage for your department with drop down navigation menu for faculty, courses, activities, etc.. Implement the webpage using styles, rules, selectors, ID, class.

Javascript

1. Create event driven programs for the following:
    a. Enter a number and on click of a button print its multiplication table.
    b. Print the largest of three numbers entered by the user.
    c. Find the factorial of a number entered by the user.
    d. Enter a list of positive numbers using the prompt terminated by a zero. Find the sum and average of these numbers.
2. Create a student registration form using text, radio button, check box, drop down box, text field and all other required HTML elements. Customise the CSS and javascript to input and validate all data. Create functions to perform validation of each element, example:
    a. Roll number is a 7-digit numeric value
    b. Name should be an alphabetical value(String)
    c. Non-empty and valid fields like DOB

jQuery and JSON

1. Change text color and contents using button click events using jQuery
2. Select elements using ID, class, elements name, attribute name
3. Run code on click events in jQuery
4. Handle HTML form, store the data in JSON object, pass them to another page and display it there using jQuery/Javascript

COMMON POOL OF GENERIC ELECTIVES (GE) COURSES

GENERIC ELECTIVES (GE-4a): Data Structures using C++

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
GE4a Data Structures using C++	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objectives

The course aims at developing the ability to use basic data structures like arrays, stacks, queues, lists, trees to solve problems. C++ is chosen as the language to understand implementation of these data structures.

Learning outcomes

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

- Compare two functions for their rates of growth.
- Understand abstract specification of data-structures and their implementation.
- Compute time and space complexity of operations on a data-structure.
- Identify the appropriate data structure(s) for a given application and understand the trade-offs involved in terms of time and space complexity.
- Apply recursive techniques to solve problems.

SYLLABUS OF GE-4a

Unit 1 (7 hours)

Growth of Functions, Recurrence Relations: Functions used in analysis, asymptotic notations, asymptotic analysis, solving recurrences using recursion tree, Master Theorem.

Unit 2 (14 hours)

Arrays, Linked Lists, Stacks, Queues, Deques: Arrays: array operations, applications, sorting, two-dimensional arrays, dynamic allocation of arrays; Linked Lists: singly linked lists, doubly linked lists, circularly linked lists, Stacks: stack as an ADT, implementing stacks using arrays, implementing stacks using linked lists, applications of stacks; Queues: queue as an ADT, implementing queues using arrays, implementing queues using linked lists, double-ended queue as an ADT. Time complexity analysis of operations on all data structures.

Unit 3 (4 hours)

Sorting: Insertion Sort, Count Sort and their complexity analysis.

Unit 4 (4 hours)

Recursion: Recursive functions, linear recursion, binary recursion.

Unit 5 (10 hours)

Trees, Binary Trees: Trees: definition and properties, binary trees: definition and properties, traversal of binary trees and their time complexity analysis. Binary Search Trees: insert, delete (by copying), search operations, time complexity analysis of these operation.

Unit 6 (6 hours)

Binary Heap, Priority Queue: Binary Heaps: motivation and introduction, application of heaps - Priority Queues.

Essential/recommended readings

1. Goodrich, M.T, Tamassia, R., & Mount, D., *Data Structures and Algorithms Analysis in C++,* 2nd edition, Wiley, 2011.
2. Cormen, T.H., Leiserson, C.E., Rivest, R. L., Stein C., Introduction to Algorithms. 4th edition. Prentice Hall of India. 2022.
3. Drozdek, A., *Data Structures and Algorithms in C++*, 4th edition, Cengage Learning, 2012.

Additional References

1. Sahni, S. *Data Structures, Algorithms and applications in C++*, 2nd edition. Universities Press, 2011.
2. Langsam Y., Augenstein, M. J., & Tanenbaum, A. M. *Data Structures Using C and C++*, 2nd edition, Pearson, 2009.

Suggested Practical List (If any): (30 Hours)

Practical exercises such as

1. Perform matrix addition and multiplication.
2. Implement following recursive functions:
    a. Factorialof a number
    b. N^{th} fibonacci number
    c. Power function: x^y
3. Implement singly linked lists.
3. Implement doubly linked lists.

4. Implement circular linked lists.
5. Implement stack data structure and its operations using arrays.
6. Implement stack data structure and its operations using linked lists.
7. Convert Prefix expression to Infix and Postfix expressions, and evaluate.
8. Implement queue data structure and its operations using arrays.
9. Implement queue data structure and its operations using linked lists.
10. Implement Binary Trees and its traversals.

GENERIC ELECTIVES (GE-4b): Introduction to Web Programming

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
GE4b: Introduction to Programming	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objectives

The course aims at introducing the basic concepts and techniques of client side web programming. The student shall be able to develop simple websites using HTML, CSS and Javascript.

Learning outcomes

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

- Build websites using the elements of HTML.
- Build dynamic websites using the client side programming techniques with CSS, Javascript and jQuery.
- Learn to validate client-side data.

SYLLABUS OF GE4b

Unit 1 (5 hours)

Introduction: Introduction to Internet and web design. Basic concepts of web architecture.

Unit 2 (12 hours)

HTML: Introduction to hypertext mark-up language (html), creating web pages, lists,

hyperlinks, tables, web forms, inserting images, frames.

Unit 3 (8 hours)

Cascading style sheet (CSS): Concept of CSS, creating style sheet, Importing style sheets, CSS properties, CSS styling (background, text format, controlling fonts), CSS rules, Style Types, CSS Selectors, CSS cascade, working with block elements and objects, working with lists and tables, CSS id and class, box model (introduction, border properties, padding properties, margin properties).

Unit 4 (10 hours)

Javascript: Document object model, data types and variables, functions, methods and events, controlling program flow, JavaScript object model, built-in objects and operators, validations.

Unit 5 (10 hours)

jQuery and JSON: Introduction to jQuery, syntax, selectors, events. JSON file format for storing and transporting data.

Essential/recommended readings

1. Nixon, R. *Learning PHP, MySQL & JavaScript with jQuery, CSS and HTML5*, O'Rielly, 2018.
2. Powell, T.A. *HTML & CSS: The Complete Reference, 5th edition*, Tata McGrawHill, 2010.
3. Duckett, J. *JavaScript and JQuery: Interactive Front-End Web Development*, Wiley, 2014.

Additional References

1. Minnick, J. *Web Design with HTML5 and CSS3*, 8th edition, Cengage Learning, 2015.
2. Boehm, A., & Ruvalcaba, Z. *Munarch's HTML5 and CCS,* 4th edition, Mike Murach & Associates, 2018.
3. J. A. Ramalho *Learn Advanced HTML 4.0 with DHTML*, BPB Publications, 2007.
4. Ivan Bayross *Web Enabled Commercial Application Development Using Html, Dhtml, Javascript, Perl CGI*, BPB Publications, 2009.

Suggested Practical List (If any): (30 Hours)

Practical exercises such as

HTML

1. Create an HTML document with following formatting – Bold, Italics, Underline, Colors, Headings, Title, Font and Font Width, Background, Paragraph, Line Brakes, Horizontal Line, Blinking text as well as marquee text.
2. Create an HTML document with Ordered and Unordered lists, Inserting Images, Internal and External linking
3. Create an HTML displaying this semester's time table.
4. Create a website with horizontal and vertical frames. Top horizontal frame showing

your college's name and logo. Bottom horizontal frame split into two vertical frames. The left frame with hyperlinks to pages related to faculty, courses, student activities, etc. The right frame showing corresponding pages based on the link clicked on the left frame.

5. Create a student registration form using HTML which has the following controls:
   I. Text Box
   II. Dropdown box
   III. Option/radio buttons
   IV. Check boxes
   V. Reset and Submit button

CSS

Create a webpage for your department with drop down navigation menu for faculty, courses, activities, etc.. Implement the webpage using styles, rules, selectors, ID, class.

Javascript

1. Create event driven programs for the following:
   e. Enter a number and on click of a button print its multiplication table.
   f. Print the largest of three numbers entered by the user.
   g. Find the factorial of a number entered by the user.
   h. Enter a list of positive numbers using the prompt terminated by a zero. Find the sum and average of these numbers.
2. Create a student registration form using text, radio button, check box, drop down box, text field and all other required HTML elements. Customise the CSS and javascript to input and validate all data. Create functions to perform validation of each element, example:
   d. Roll number is a 7-digit numeric value
   e. Name should be an alphabetical value(String)
   f. Non-empty and valid fields like DOB

jQuery and JSON

1. Change text color and contents using button click events using jQuery
2. Select elements using ID, class, elements name, attribute name
3. Run code on click events in jQuery

SEMESTER-5

DEPARTMENT OF COMPUTER SCIENCE
[UG Programme for Bachelor in Computer Science (Honours)]

DISCIPLINE SPECIFIC CORE COURSE - 13 (DSC-13) : Algorithms and Advanced Data Structures

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
DSC 13 Algorithms and Advanced Data Structures	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 07 Data Structures with C++, DSC 10 Design and Analysis of Algorithms

Learning Objectives

This course is designed to build upon the fundamentals in data structures and algorithm design and gain exposure to more data structures and algorithms for new problems.

Learning outcomes

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

- Comprehend and use data structures for lists.
- Use hash tables for dictionaries.
- Comprehend and use data structures and algorithms for string matching.
- Apply disk based data structures.
- Implement and analyze advanced data structures and algorithms for graphs.
- Describe the purpose of randomization in data structures and algorithms.

Unit 1 (4 hours)

List and Iterator ADTs: Vectors, Lists, Sequences

Unit 2 (6 hours)

Hash Tables, Dictionaries: Hash Functions, Collision resolution schemes.

Unit 3 (8 hours)

Strings: String Matching: KMP algorithm; Tries: Standard Tries, Compressed Tries, Suffix Tries, Search Engines

Unit 4 (8 hours)

More on Trees: 2-4 Trees, B Trees

Unit 5 (8 hours)

More on Graphs: Bellman Ford Algorithm, Union Find Data Structures - application Kruskal's algorithm

Unit 6 (6 hours)

Randomization: Randomized Quicksort, Randomized Select, Skip lists

Unit 7 (5 hours)

Network Flows: Ford Fulkerson algorithm for max flow problem.

Essential/recommended readings

1. Goodrich, M.T, Tamassia, R., & Mount, D. *Data Structures and Algorithms Analysis in C++*, 2nd edition, Wiley, 2011.
2. Cormen, T.H., Leiserson, C.E., Rivest, R. L., Stein C. *Introduction to Algorithms*, 4th edition, Prentice Hall of India, 2022.
3. Kleinberg, J., Tardos, E. *Algorithm Design*, 1st edition, Pearson, 2013.
4. Drozdek, A. *Data Structures and Algorithms in C++*, 4th edition, Cengage Learning. 2012.

Practical List : (30 Hours)

Practical exercises such as

1. Write a program to sort the elements of an array using Randomized Quick sort (the program should report the number of comparisons).
2. Write a program to find the ith smallest element of an array using Randomized Select.
3. Write a program to determine the minimum spanning tree of a graph using Kruskal's algorithm.
4. Write a program to implement the Bellman Ford algorithm to find the shortest paths from a given source node to all other nodes in a graph.

5. Write a program to implement a B-Tree.
6. Write a program to implement the Tree Data structure, which supports the following operations:
   I. Insert
   II. Search
7. Write a program to search a pattern in a given text using the KMP algorithm.
8. Write a program to implement a Suffix tree.

DISCIPLINE SPECIFIC CORE COURSE – 14 (DSC-14): Theory of Computation

Credit distribution, Eligibility and Prerequisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
DSC 14 Theory of Computation	**4**	**3**	**0**	**1**	Pass in Class XII	DSC04 Object Oriented Programming with C++ / GE1a Programming using C++ /A course in C/C++ at plus 2 level

Learning Objectives

This course introduces formal models of computation, namely, finite automaton, pushdown automaton, and Turing machine; and their relationships with formal languages. make students aware of the notion of computation using abstract computing devices. Students will also learn about the limitations of computing machines as this course addresses the issue of which problems can be solved by computational means (decidability vs undecidability

Learning outcomes

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

- design a finite automaton, pushdown automaton or a Turing machine for a problem at hand.

- apply pumping lemma to prove that a language is non-regular/non-context-free.
- describe limitations of a computing machines and
- recognize what can be solved and what cannot be solved using these machines.

SYLLABUS OF DSC 14

Unit 1 (7 hours)

Introduction: Alphabets, string, language, basic operations on language, concatenation, union, Kleene star.

Unit 2 (15 hours)

Finite Automata and Regular: Regular expressions, Deterministic Finite Automata (DFA), Non-deterministic Finite Automata (NFA), relationship between NFA and DFA, Transition Graphs (TG), properties of regular languages, the relationship between regular languages and finite automata, pumping lemma, Kleene's theorem.

Unit 3 (15 hours)

Context-Free Languages (CFL): Context-Free Grammars (CFG), deterministic and non-deterministic Pushdown Automata (PDA), relationship between CFG and PDA, parse trees, leftmost derivation, Ambiguities in grammars, pumping lemma for CFL, properties of CFL, Chomsky Normal Form.

Unit 4 (8 hours)

Turing Machines and Models of Computations: Turing machine as a model of computation, configuration of Turing machine, Recursive and recursively enumerable languages, Church Turing Thesis, Universal Turing Machine, decidability, Halting problem.

Essential/recommended readings

1. Harry R. Lewis and Christos H. Papadimitriou, *Elements of the Theory of Computation*, 2nd Edition, Prentice Hall of India (PHI), 2002
2. Daniel I.A. Cohen, *Introduction to Computer Theory*, 2nd Edition, Wiley India Pvt. Ltd., 2011.

Additional References

1. J.E. Hopcroft, R. Motwani, and J.D. Ullman, *Introduction to Automata Theory, Languages and Computation*, 3rd edition, Addison Wesley, 2006.
2. Peter Linz, *An Introduction to Formal Languages and Automata*, 6th edition, Jones & Bartlett Learning, 2017.
3. Michael Sipser, *Introduction to the Theory of Computation*, Cengage, 2014

DISCIPLINE SPECIFIC CORE COURSE– 15 (DSC-15): Software Engineering

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
DSC 15 Software Engineering		**3**	**0**	**1**	Pass in Class XII	DSC01 Programming using Python/ DSC04 Object Oriented Programming with C++/A course in C/C++ or Python at plus 2 level

Learning Objectives

This course will acquaint the student with different approaches and techniques used to develop good quality software. The course includes learning of various software development process frameworks, requirement analysis, design modeling, qualitative and quantitative software metrics, risk management, and testing techniques.

Learning outcomes

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

- describe the software development models.
- analyse and model customer requirements and build design models.
- estimate and prepare schedule for software projects.
- analyse the impact of risks involved in software development.
- design and build test cases, and perform software testing.

SYLLABUS OF DSC 15

Unit 1 (9 hours)

Introduction: Software Engineering - A Layered Approach; Software Process – Process Framework, Umbrella Activities; Process Models – Waterfall Model, Incremental Model, and Evolutionary process Model (Prototyping, Spiral Model); Introduction to Agile, Agile Model – Scrum.

Unit 2 (6 hours)

Software Requirements Analysis and Specification: Use Case Approach, Software Requirement Specification Document, Flow-oriented Model, Data Flow Model

Unit 3 (8 hours)

Design Modeling: Translating the Requirements model into the Design Model, The Design Process, Design Concepts - Abstraction, Modularity and Functional Independence; Structure Charts.

Unit 4 (7 hours)

Software Metrics and Project Estimation: Function based Metrics, Software Measurement, Metrics for Software Quality; Software Project Estimation (FP based estimations); Project Scheduling (Timeline charts, tracking the schedule).

Unit 5 (5 hours)

Quality Control and Risk Management: Quality Control and Quality Assurance, Software Process Assessment and Improvement; Software Risks, Risk Identification, Risk Projection, Risk Mitigation, Monitoring and Management.

Unit 6 (10 hours)

Software Testing: Strategic Approach to Software Testing, Unit Testing, Integration Testing, Validation Testing, System Testing; Black-Box and White Box Testing, Basis Path Testing.

Essential/recommended readings

1. Pressman, R.S. *Software Engineering: A Practitioner's Approach*, 9th edition, McGraw-Hill, 2020.
2. Aggarwal, K.K., Singh, Y. *Software Engineering*, 3rd edition, New Age International Publishers, 2007.
3. Jalote, P. An Integrated Approach to Software Engineering, 3rd Edition, Narosa Publishing House, 2005.

Additional References

1. Sommerville, I. *Software Engineering*, 9th edition, Addison Wesley, 2011.
2. *The Definitive Guide to Scrum: The Rules of the Game*, Ken Schwaber, Jeff Sutherland, July 2016.

Suggested Practical List :(30 Hours)

Practical exercises such as

The students document, design and code a module of a Software Project using an appropriate Software Process model. The Software Project should include the use of software engineering tools and include.

1. Problem Statement, Process Model
2. Requirement Analysis: Create Data Flow, Data Dictionary, Use Cases, Sequence Diagram, Software Requirement Specification Document
3. Project Management: Timeline Chart, Compute FP, Effort, Cost, Risk Table.
4. Design Engineering: Architectural Design, Pseudocode of a small module.
5. Coding: Develop at least a single module using any programming Language
6. Testing: Compute Basic path set for at least one module from a project, Generate test cases.

Some of the sample projects are given below:

1. Criminal Record Management: Implement a criminal record management system for jailers, police officers and CBI officers
2. DTC Route Information: Online information about the bus routes and their frequency and fares.
3. Car Pooling: To maintain a web-based intranet application that enables the corporate employees within an organization to avail the facility of carpooling effectively.
4. Patient Appointment and Prescription Management System
5. Organized Retail Shopping Management Software
6. Online Hotel Reservation Service System
7. Examination and Result computation System
8. Automatic Internal Assessment System
9. Parking Allocation System

Computer Science Courses for Undergraduate Programme of study with Computer Science discipline Elective

DISCIPLINE SPECIFIC ELECTIVE COURSE: Data Mining-II

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
Data Mining-II	**4**	**3**	**0**	**1**	Pass in Class XII	DSC01 Programming using Python, / GE1b Programming with Python / A1 Programming Fundamentals using Python ,Data Mining-I

Learning Objectives

The course introduces the students to the important supervised and unsupervised learning techniques. Students will learn about the importance of ensemble methods, cluster analysis, anomaly detection and their applicability in mining patterns in real applications. At the end students will be exposed to two advanced topics: text mining and time-series mining. Students will use the learned topics in solving real applications using Open-source software.

Learning outcomes

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

- Differentiate between partition-based, density-based and hierarchical clustering.
- Build ensemble models to improve predictive performance of the classifier.
- Identify anomalies and outliers using supervised and unsupervised techniques.
- Analyze time-series data and extract patterns from the stamped data.
- Mine textual data and do topic modelling.

SYLLABUS OF DSE

Unit 1 (10 hours)

Clustering: Partitioning Methods, Hierarchical Methods, Density-Based Methods, Comparison of different methods

Unit 2 (8 hours)

Ensemble Methods: Need of ensemble, Random Forests, Bagging and Boosting

Unit 3 (10 hours)

Anomaly Detection: Outliers and Outlier Analysis, Outlier Detection Methods, Statistical Approaches, Proximity-based and density-based outlier detection, Clustering-based approaches

Unit 4 (7 hours)

Mining Text Data: Document Preparation and Similarity, Clustering Methods for Text, Topic Modeling

Unit 5 (10 hours)

Stream Mining: Time series basics, Date Ranges, Frequencies, and Shifting, Resampling and moving windows functions, Decay function, Clustering stamped data**:** STREAM and CluStream

Essential/recommended readings

1. Tan P.N., Steinbach M, Karpatne A. and Kumar V. *Introduction to Data Mining,* 2nd edition, Pearson, 2019.
2. Zaki M. J. and Meira J. Jr. *Data Mining and Machine Learning: Fundamental Concepts and Algorithms*, 2nd edition, Cambridge University Press, 2020.
3. Aggarwal C. C. *Data Mining: The Textbook*, Springer, 2015.

Additional References

1. Han J. Kamber M. and Pei J. *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, 2011.
2. Dunham M. *Data Mining: Introductory and Advanced Topics*, Pearson, 2006.

Suggested Practical List : (30 Hours)

Practical exercises such as

Use a dataset of your choice from Open Data Portal (https:// data.gov.in/, UCI repository or a dataset verified by the teacher) or load from scikit, seaborn library for the following exercises to practice the concepts learnt.

1. Apply Partitioning Methods, Hierarchical Methods, Density-Based Methods for clustering on a data set and compare the performance of the obtained results using different metrics.
2. Create an ensemble using Random Forest and show the impact of bagging and boosting on the performance.

3. Apply different outlier-detection methods on a noisy dataset and compare their effectiveness in terms of outliers reported.
4. Compute similarity between two documents after required document preparation.
5. Considering a time-stamped data (sales data/weather data), compare the aggregate values visually using different moving windows function.
6. Write a program to find the latent topics in a document using any topic modeling method and display top 5 terms that contribute to each topic along with their strength. Also, visualize the distribution of terms contributing to the topics.

Project: Students are encouraged to work on a good dataset in consultation with their faculty and apply the concepts learned in the course.

DISCIPLINE SPECIFIC ELECTIVE COURSE: Data Privacy

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
Data Privacy	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objectives

This course aims to provide students with an ability to identify privacy related aspects of data uses(including attacks on data privacy), evaluate proposed mechanisms for privacy protection and relate to ethical issues related to data privacy.

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

- Understand the basic principles of data privacy and the implications of data breaches.
- Identify and evaluate different methods of protecting sensitive data.
- Explain the role of privacy regulations in safeguarding personal information.
- Implement basic cryptographic techniques to secure data.
- Apply data anonymization techniques to protect personal information.
- Analyze the ethical considerations in data privacy

SYLLABUS OF DSE

Unit 1 (10 hours)

Introduction to Data Privacy and Privacy Regulations: Notion of data privacy, Historical context of data privacy, Types of sensitive data, Privacy laws and regulations.

Unit 2 (15 hours)

Data Privacy Attacks, Cryptography and Data Protection: Type of Attacks/ Data Breaches on Data Privacy, Impact of Data Breaches / Attacks, Introduction to cryptography, Symmetric and asymmetric encryption, Hashing and digital signatures.

Unit 3 (10 hours)

Data Collection, Use and Reuse: Harms Associated with Data collections, use and reuse, Introduction to data anonymization, Data Anonymization Techniques for anonymizing data, Challenges in anonymizing data

Unit 4 (10 hours)

Ethical considerations in Data Privacy: Privacy and Surveillance, Ethics of Data Collection and Use, Bias and discrimination in data analysis

Essential/recommended readings

1. Ronald Leenes, Rosamunde van Brakel, and Serge Gutwirth: *Data Protection and Privacy: The Age of Intelligent Machines*, Hart Publishing, 2017.
2. Naavi: *Personal Data Protection Act of India (PDPA 2020)*: Be Aware, Be Ready and Be Compliant, Notion Press, 2020.
3. Ravinder Kumar Gaurav Goyal, *The Right to Privacy in India: Concept and Evolution*, Publisher: Lightning Source, 2016.

Additional References

1. https://onlinecourses.nptel.ac.in/noc22_cs37/preview
2. https://www.coursera.org/learn/northeastern-data-privacy/home/info

Suggested Practical List : (30 Hours)

Practical exercises such as

1. Data Privacy Audit: Students can conduct a data privacy audit of a company or organization to identify potential vulnerabilities and risks in their data privacy practices.

2. Privacy Impact Assessment: Students can conduct a privacy impact assessment (PIA) of a new technology or system to identify potential privacy risks and develop strategies to mitigate them.
3. Regulation Compliance: Students can explore the requirements of the Data Protection Regulations and develop a plan for ensuring compliance with the regulation.
4. Cryptography: Students can learn about different cryptographic techniques and tools, such as encryption, hashing, and digital signatures, and implement them in practice.
5. Anonymization Techniques: Students can learn about data anonymization techniques, such as k-anonymity, differential privacy, and data masking, and apply them to a real-world dataset.
6. Privacy Policy Analysis: Students can analyze the privacy policies of different companies and identify gaps or areas for improvement.
7. Privacy-Enhancing Technologies: Students can explore privacy-enhancing technologies (PETs), such as virtual private networks (VPNs), Tor, and secure messaging apps, and evaluate their effectiveness in protecting privacy.
8. Privacy Breach Response Plan: Students can develop a privacy breach response plan for a company or organization, including steps to take in the event of a data breach and strategies for communicating with affected parties.
9. Ethical Considerations: Students can explore ethical considerations in data privacy, such as the balance between privacy and security, the impact of data collection and analysis on marginalized communities, and the role of data ethics in technology development.
10. Case Studies: Students can analyze case studies of privacy breaches or successful privacy protection strategies, and identify key lessons and takeaways.

DISCIPLINE SPECIFIC ELECTIVE COURSE: Unix Network Programming

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	
		Lecture	Tutorial	Practical/ Practice		
Unix Network Programming	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 04 Object Oriented Programming with C++/ GE 1a Programming using C++ / GE1b Programming with Python/ DSC 01 Programming using Python/ GE 3b: Java Programming

Learning Objectives

This course introduces the concepts of Internet protocols, ports used during communication, Client/Server concepts and various transport protocols used in computer network applications and services. The objective is to equip the students with technical knowledge of it comprises of the study of the sockets used with TCP and UDP.

Learning outcomes

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

- Describe and analyze the various Internet Transport layer protocols used in TCP AND UDP.
- Comprehend the concepts and structures of both TCP based connection-oriented and UDP based connectionless client server applications.
- Write various real-life client-server applications using socket programming.
- Modify, maintain and extend the present internet client-server applications and write any new type of internet applications to suit the current needs of Internet users.

SYLLABUS OF DSE

Unit 1 (6 hours)

Introduction Basics of Client Server applications, Example of day time client server, concurrent servers, protocols, sockets, port numbers.

Unit 2 (17 hours)

Connection-oriented Socket Applications: Elementary TCP sockets – Socket, connect, bind, listen, accept, fork and exec function, close function, Socket Address Structures, Byte Ordering and Manipulation Functions, TCP Client and Server for Echo, Signal Handling in case of crashing and rebooting of server, Shutdown process function. Socket Options: Getsockopt and stockpot functions, Socket states, Generic socket option.

Unit 3 (15 hours)

Connectionless Socket Applications: TCP-oriented basic concurrent client server applications, UDP oriented Echo client and server application, Handling of errors like lost datagram, Lack of flow control with UDP, determining outgoing interface with UDP.

Unit 4 (7 hours)

Elementary name and Address conversions: Domain Name System, socket functions like gethostbyname, gethostbyname2, gethostbyaddr function, uname function, gethostname function, getservbyname and getservbyport functions.

Essential/recommended readings

1. W. Richard Stevens, Bill Fenner, Andrew M. Rudoff, *Unix Network Programming*, The sockets Networking API, Vol. 1, 3rd Edition, PHI.
2. B. A. Forouzan: *Data Communications and Networking*, THM Publishing Company Ltd
3. R. Stevens, *Unix Network Programming*, PHI 2nd Edition

Suggested Practical List : (30 Hours)

Practical exercises such as

1. Implement TCP Echo client and TCP Echo server (Iterative).
2. Implement TCP Echo client and TCP Echo server (Concurrent).
3. Implement TCP daytime client and TCP daytime server (Iterative).
4. Implement TCP daytime client and TCP daytime server (concurrent).
5. Implement UDP Echo Client and UDP Echo Server.
6. Implement UDP daytime Client and UDP daytime server.
7. Implement TCP client and server (concurrent) where client gets input from the user and sends it to server. Server displays it on the screen. Server then gets another input from the user and sends it to client. Client displays it on the screen. The process continues till server or client sends “bye” to the other party.
8. Implement TCP client and server (concurrent) where client requests server to transfer a file. Assume file is smaller than 1K size. If the file is present on the server, it is sent to the client otherwise an error message is sent to client. Client copies the file on the hard disk and disconnects.
9. Implement UDP client and UDP server where server displays the IP address and port number of the client sending the datagram. Client sends a datagram (size 64 bytes) three times to the same server. Server sends the message back to client. Client reports the time elapsed in sending and receiving of the message. Use connected UDP sockets.
10. Write a program to
    a) display name of the host
    b) all IP addresses of the host.
    c) Check whether FTP and HTTP services are running on the system.
    d) Display the name of the service running on port number specified by user.

COMMON POOL OF GENERIC ELECTIVES (GE) COURSES

(For all the Generic Elective courses offered by your department, please put it in the format provided below)

GENERIC ELECTIVES (GE-5a): Operating Systems

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
GE-5a Operating Systems	**4**	**3**	**0**	**1**	Pass in Class XII	Knowledge of Programming in C/C++/Java/Python at class XII level or above

Learning Objectives

The course introduces Operating System and its importance in computer systems. The focus is to explain the common services provided by an operating system like process management, memory (primary, secondary & virtual) management, I/O management, file management. The course talks about the various functional components of the operating and their design.

Learning outcomes

On successful completion of the course, students will be able to gain knowledge of different concepts of the operating System and its components. They would learn about shell scripts and would be able to use the system in an efficient manner.

SYLLABUS OF GE-5a

Unit 1 (5 hours)

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

Unit 2 (6 hours)

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

Unit 3 (10 hours)

Process Management: Process Concept, States. Process Control Block, Context Switch, Process scheduling, Schedulers, Overview of threads and Scheduling Algorithms: First Come First Served, Shortest-Job-First, Priority & Round-Robin.

Unit 4 (10 hours)

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

Unit 5 (8 hours)

File and Input / Output Device Management: File Concepts, File Attributes, File Access Methods, Directory Structure: Single-Level, Two-Level, Tree-Structured, and Acyclic-Graph Directories, Magnetic Disks, Solid-State Disks, Magnetic Tapes.

Unit 6 (6 hours)

Shell Scripting: Shell variables, parameter passing conditional statements, iterative statements, writing and executing shell scripts, utility programs (cut, paste, grep, echo, pipe, filter etc.)

Essential/recommended readings

1. Galvin, S. P. B., Gagne, G., *Operating System Concepts*, 9th edition, John Wiley Publications, 2016.
2. G. Nutt, Operating Systems, Pearson, 2009

Additional References

1. Dhamdhere, D. M., *Operating Systems: A Concept-based Approach,* 2nd edition, Tata McGraw-Hill Education, 2017.
2. Kernighan, B. W., Pike, R., *The Unix Programming Environment*, Englewood Cliffs, NJ: Prentice-Hall, 1984.
3. Stallings, W., *Operating Systems: Internals and Design Principles*, 9th edition, Pearson Education, 2018.
4. Tanenbaum, A. S., *Modern Operating Systems.* 3rd edition. Pearson Education, 2007.

Suggested Practical List : (30 Hours)

Practical exercises such as

1. Usage of following commands: ls, pwd, cat, who, rm, mkdir, rmdir,cd.

2. Usage of following commands: cal, cat(append), cat(concatenate), mv, cp, man, date.
3. Usage of following commands: chmod, grep, bc.
4. Write a shell script to display date in the mm/dd/yy format.
5. Write a shell script to display the multiplication table any number.
6. Write a shell script to find the factorial of a given number.
7. Program to show the pyramid of special character "*".
8. Write a shell script to find the sum of digits of a given number.
9. Write a shell script to perform the tasks of basic calculator.
10. Write a shell script to find the power of a given number.
11. Write a shell script to check whether the number is Armstrong or not.
12. Write a shell script to find the GCD (greatest common divisor) of two numbers.
13. Write a shell script to check if the number entered at the command line is prime or not.
14. Write a shell script to display on the screen sorted output of "who" command along with the total number of users.
15. Write a shell script to accept a login name. If not a valid login name display message – "Entered login name is invalid".
16. Write a shell script to compare two files and if found equal asks the user to delete the duplicate file.
17. Write a shell script to merge the contents of three files, sort the contents and then display them page by page.
18. Write a shell script to check whether the file have all the permissions or not.
19. Write a shell script to modify "cal" command to display calendars of the specified months.
20. Write a shell script to modify "cal" command to display calendars of the specified range of months.

GENERIC ELECTIVES (GE-5b): Advanced Web Programming

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
GE5b: Advanced Web Programming	**4**	**3**	**0**	**1**	Pass in Class XII	Knowledge of HTML, CSS

Learning Objectives

The course aims to familiarize the students with the concepts and techniques of server side web programming. This will enable the students to create dynamically generated web pages using HTML, PHP, MySql and JQuery.

Learning outcomes

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

- develop interactive and dynamic websites.
- write programs to communicate with the server using GET and POST methods
- learn to connect and manipulate databases using MySql
- validate server-side/backend data

SYLLABUS OF GE-5b

Unit 1 (7 hours)

Introduction to PHP: Basic syntax, defining variables and constants, data types including arrays, operators and expressions, decision making statements, constructs for iterations.

Unit 2 (5 hours)

String Handling: Creating a string and accessing its content, searching and replacing content of a string, and other built-in functions.

Unit 3 (12 hours)

Handling HTML Form with PHP: Creating a form, submitting data to the server at the backend using GET and POST methods, GET vs POST methods, PHP global functions.

Unit 4 (15 hours)

Database: Connectivity with MySQL: Connectivity with database, database creation, creating tables, create, retrieve, update, and delete (CRUD) operations

Unit 5 (6 hours)

jQuery and JSON: Introduction to jQuery syntax (selectors, events, AJAX, JSON).

Essential/recommended readings

1. Nixon, R. Learning PHP, MySQL & JavaScript with jQuery, CSS and HTML5, O'Reilly, 2018.
2. Holzner S. PHP: The Complete Reference, McGraw Hill, 2017
3. Murach J, Murach's PHP and MySQL, 2nd edition, Mike Murach & Associates, 2014.
4. Duckett, J. JavaScript and JQuery: Interactive Front-End Web Development, Wiley, 2014.

Additional References

1. https://www.w3schools.com/php/default.asp

2. https://www.tutorialspoint.com/php/index.htm

Suggested Practical List : (30 Hours)

Practical exercises such as

1. Write a PHP script to reverse the digits of a number.
2. Create a web page containing two text boxes and a button named "Evaluate". When the user enters numbers in the text boxes and clicks on the "Evaluate" button, a function should evaluate the sum of the numbers and display the result.
3. Write a PHP script to perform following string operations using in-built functions and built an interactive web page having buttons for each of the following operation:
    a. Find the length of a string
    b. Find a substring from a string
    c. Replace text within a string
    d. Remove whitespace and other predefined characters from both sides of a string.
    e. Check if a value is a string
    f. Convert the first character of each word in a string into uppercase.
4. Design a Login form and validate that form using PHP code. Display error message box when data is not valid otherwise redirect to the next page and display "Welcome username!".
5. Design a student registration form, using appropriate input fields consisting of following:
    a. First Name
    b. Last Name
    c. Gender
    d. Roll Number
    e. Phone Number
    f. Course

Submit and retrieve the form data using $_POST, $_GET variable.

6. Write PHP Code to make connection to MySql database, create database and tables and perform insertion, deletion, and retrieval of the data (Using SQL operations like .JOIN, ORDER BY, GROUP BY) Display the messages such as "The record is added in the database!" when data is inserted into the database, "A record is deleted from the database" when data is deleted from the database. Use appropriate button names such as Add Data, Delete Data, and Display Data.

jQuery and JSON

1. Change text color and contents using button click events using jQuery
2. Select elements using ID, class, elements name, attribute name
3. Run code on click events in jQuery

4. Handle HTML form, store the data in JSON object, pass them to another page and display it there using jQuery/Javascript

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
GE5c: Java Based Web App Development	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 04 Object Oriented Programming with C++/ GE 1a Programming using C++ / GE1b Programming with Python/ DSC 01 Programming using Python/ GE 3b: Java Programming / A programming course at class XII level

Learning Objectives

The course aims to familiarize the students with the concepts and techniques of web app development based on Java. The students will learn about database connectivity, use of HTTP protocol, client side programming, and use of servlets and JSP for server side programming.

Learning outcomes

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

- develop an understand of client-server architecture, HTTP protocol, and web application components.
- connect an application to database and perform basic database operations.
- create servlets and JSP for web applications
- deploy web applications

SYLLABUS OF GE-5c

Unit 1 (8 hours)

Review of Programming Language: Programming Constructs, Data types, Operators, Concepts of Class, Interface, Inheritance, Exception Handling, Util package, Multithreading, event handling.

Unit 2 (10 hours)

Java Database Connections: Database connectivity, Connection, statement, result set object, Metadata, Connection pooling, CRUD operations, Prepared and callable statements

Unit 3 (15 hours)

Introduction to servlets: Concepts of Streams, events and listener, recap of HTML, CSS, XML, Servlet package and interface, life cycle of servlet, deployment descriptor, Filters, HHTP and Generic servlet, request dispatcher, Request Response classes, Dynamic page designing using servlet.

Unit 4 (12 hours)

Introduction to JSP: JSP Life cycle, tags in JSP, custom tags, Expression Language, Introduction to Struts Framework, Implicit objects, database access using JSP

Essential/recommended readings

1. Herbert Schildt, *Java : The Complete Reference***,** 12th edition, McGraw-Hill Education, 2021.
2. Hans Bergsten, *Java Server Pages*, 3rd edition, O′Reilly, 2003.
3. Jim Keogh, *The Complete Reference J2EE*, 1st edition, McGraw-Hill Education, 2017.

Suggested Practical List : (30 Hours)

Practical exercises such as

1. Setting up the development environment: Install Java Development Kit (JDK), Eclipse IDE, and Apache Tomcat web server. Create a new web project in Eclipse.
2. Writing and deploying a "Hello World" servlet: Create a simple servlet that prints "Hello World" on the web page. Deploy the servlet on Tomcat and test it in a web browser.
3. Handling HTTP requests and responses: Write a servlet that reads input from HTTP requests and sends output as HTTP responses.
4. Creating a JSP page: Create a JSP page that displays dynamic.
5. Write a servlet that handles form submissions and saves the data to a database using JDBC.
6. Write a servlet that implements user authentication and authorization using a database.
7. Creating a web application using MVC architecture: Create a web application using Model-View-Controller (MVC) architecture. Use servlets as controllers, JSP pages as views.
8. Deploying a web application to a server: Configure and deploy a web application to a server using Apache Tomcat Manager or other deployment tools.

(Computer Science Courses for Undergraduate Programme of study with Computer Science discipline as one of the three Core Disciplines)

DISCIPLINE SPECIFIC CORE COURSE (DSC-5): Database Management Systems

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
DSC 5: Database Management Systems	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objectives

The course introduces the students to the fundamentals of database management system and its applications. Emphasis is given on the popular relational database system. Students will learn about the importance of database structure and its designing using Entity Relationship diagram and formal approach using normalization. Basic concepts of file indexing and transaction processing will be taught. The course would give students hands-on practice of structured query language to create, manipulate and implement a relational database.

Learning outcomes

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

- Use relational database management software to create and manipulate the database.
- Create conceptual data models using entity relationship diagrams for modeling real-life situations and map it to corresponding relational database schema.
- Use the concept of functional dependencies to remove redundancy and update anomalies.
- Apply normalization theory to get a normalized database scheme to get anomalies free database.
- Write queries in relational algebra.
- Implement relational databases and formulate queries for data retrieval and data update problems using SQL.
- Learn the importance of index structures and concurrent execution of transactions in database systems.

Semester-VI

DEPARTMENT OF COMPUTER SCIENCE

[UG Programme for Bachelor in Computer Science (Honours)]

DISCIPLINE SPECIFIC CORE COURSE – 16 (DSC-16) : Artificial Intelligence

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
DSC-16 Artificial Intelligence	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 04 Object Oriented Programming with C++/ GE 1a Programming using C++ / GE1b Programming with Python/ DSC 01 Programming using Python/ GE 3b: Java Programming

Learning Objectives

The course objectives of this course are to:

- To introduce basic concepts and techniques of Artificial Intelligence (AI).
- To apply informed search techniques for different applications.
- To learn various knowledge representation techniques and writing Prolog programs.
- To learn about the latest techniques for developing AI systems.

Learning outcomes

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

- identify problems that are amenable to solutions by specific AI methods.
- state the utility of different types of AI agents.
- apply different informed search techniques for solving real world problems.
- use knowledge representation techniques for AI systems.

SYLLABUS OF DSC-16

Unit 1 6 Hours

Introduction: Introduction to artificial intelligence, background and applications, Turing test, Weak AI, Strong AI, Narrow AI, Artificial General Intelligence, Super AI, rational agent

approaches to AI, introduction to intelligent agents, their structure, behavior and task environment.

Unit 2 12 Hours

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

Unit 3 16 Hours

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

Unit 4 8 Hours

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

Unit 5 3 Hours

AI The Present and the Future: Symbolic AI, Data-driven AI and Machine Learning, Introduction to Machine Learning and Deep Learning based AI, some applications of symbolic and data driven AI, Interpretable and Explainable AI, Ethics of AI: benefits and risks of AI.

Essential/recommended readings

1. Russell, Stuart, J. and Norvig, Peter, *Artificial Intelligence - A Modern Approach*, Pearson, 4th edition, 2020..
2. Bratko, Ivan, *Prolog Programming for Artificial Intelligence*, Addison-Wesley, Pearson Education, 4th edition, 2012.
3. Patterson, DAN,W, *Introduction to A.I. and Expert Systems* – PHI, 2007.
4. Clocksin, W., F. and Mellish, *Programming in PROLOG*, 5th edition, Springer, 2003.

Additional references

1. Kaushik, Saroj, *Artificial Intelligence*, Cengage Learning India, 2011.
2. Rich, Elaine and Knight, Kelvin, *Artificial Intelligence*, 3rd edition, Tata McGraw Hill, 2010

Practical List :

Practical exercises such as

1. Write a program in Prolog to implement TowerOfHanoi(N) where N represents the number of disks.
2. Write a program to implement the Hill climbing search algorithm in Prolog.
3. Write a program to implement the Best first search algorithm in Prolog.
4. Write a program to implement A* search algorithm in Prolog.
5. Write a program to implement the min-max search algorithm in Prolog.

6. Write a program to solve the Water-Jug Problem in Prolog.
7. Implement sudoku problem (minimum 9×9 size) using constraint satisfaction in Prolog.
8. Write a Prolog program to implement the family tree and demonstrate the family relationship.
9. Write a Prolog program to implement knowledge representation using frames with appropriate examples.
10. Write a Prolog program to implement conc(L1, L2, L3) where L2 is the list to be appended with L1 to get the resulted list L3.
11. Write a Prolog program to implement reverse(L, R) where List L is original and List R is reversed list.
12. Write a Prolog program to generate a parse tree of a given sentence in English language assuming the grammar required for parsing.
13. Write a Prolog program to recognize context free grammar $a^n b^n$.

DISCIPLINE SPECIFIC CORE COURSE – 17 (DSC-17): Machine Learning

Credit distribution, Eligibility and Prerequisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
DSC-17 Machine Learning	**4**	**3**	**0**	**1**	Pass in Class XII	DSC01 Programming using Python / A course in Python at plus 2 level

Learning Objectives

The course aims at introducing the basic concepts and techniques of machine learning so that a student can apply machine learning techniques to a problem at hand.

Learning outcomes

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

- Differentiate between supervised and unsupervised learning tasks.
- State the need of preprocessing, feature scaling and feature selection.
- Formulate classification, regression and clustering problems as optimization problems
- Implement various machine learning algorithms learnt in the course.

SYLLABUS OF DSC-17

Unit 1 (5 Hours)

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

Unit 2 (8 Hours)

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

Unit 3 (12 Hours)

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

Unit 4 (12 Hours)

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

Unit 5 (8 Hours)

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

Essential/recommended readings

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

Additional References

1. Flach, P., *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*, Cambridge University Press, 2015.
2. Christopher & Bishop, M., *Pattern Recognition and Machine Learning*, New York: Springer-Verlag, 2016.
3. Sebastian Raschka, *Python Machine Learning,* Packt Publishing Ltd, 2019

Suggested Practical List:

Practical exercises such as

Use Python for practical labs for Machine Learning. Utilize publicly available datasets from online repositories like https://data.gov.in/ and https://archive.ics.uci.edu/ml/datasets.php

For evaluation of the regression/classification models, perform experiments as follows:

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

Report the efficacy of the machine learning models as follows:

- MSE and R^2 score for regression models
- Accuracy, TP, TN, FP, TN, error, Recall, Specificity, F1-score, AUC for classification models

For relevant datasets make prediction models for the following

1. Naïve Bayes Classifier
2. Simple Linear Regression multiple linear regression
3. Polynomial Regression
4. Lasso and Ridge Regression
5. Logistic regression
6. Artificial Neural Network
7. *k*-NN classifier
8. Decision tree classification
9. SVM classification
10. K-Means Clustering
11. Hierarchical Clustering

DISCIPLINE SPECIFIC CORE COURSE– 18 (DSC-18): Introduction to Parallel Programming

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
DSC-18 Introduction to Parallel Programming	**4**	**3**	**0**	**1**	Pass in Class XII	DSC-02 Computer Systems Architecture DSC04 /A course in C/C++ at plus 2 level, DSC-07 Data Structures with C++, DSC-08 Operating Systems

Learning Objectives

The course introduces the students to the basic concepts and techniques of parallel programming. It enables them to design and implement parallel algorithms. The course would give the students hands-on practice to write parallel programs using shared and distributed memory models using OpenMP and Message Passing Interface (MPI).

Learning outcomes

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

- State the need of Parallel algorithms
- Describe architectures for parallel and distributed systems.
- Develop elementary parallel algorithms for shared memory models.
- Develop elementary parallel algorithms for distributed memory models.

SYLLABUS OF DSC-18

Unit 1 (6 Hours)

Introduction to Parallel Computing: Trends in microprocessor architectures, memory system performance, dichotomy of parallel computing platforms, physical organization of parallel platforms, communication costs in parallel machines, SIMD versus MIMD architectures, shared versus distributed memory, PRAM shared-memory model, distributed-memory model.

Unit 2 (15 Hours)

OpenMP programming for shared memory systems: Thread Basics, Controlling Thread and Synchronization Attributes, Multi-thread and multi-tasking, Context Switching, Basic OpenMP thread functions, Shared Memory Consistency Models and the Sequential Consistency Model, Race Conditions, Scoping variables, work-sharing constructs, critical sections, atomic operations, locks, OpenMP tasks, Introduction to tasks, Task queues and task execution, Accessing variables in tasks, Completion of tasks and scoping variables in tasks, Recursive task spawning and pitfalls

Unit 3 (15 Hours)
MPI programming for distributed memory systems: MPI basic communication routines (Introduction to MPI and basic calls, MPI calls to send and receive data, MPI call for broadcasting data, MPI Non-blocking calls, MPI Collectives (MPI Collectives and MPI broadcast, MPI Gathering and scattering collectives, MPI reduction and Alltoall collectives, MPI collectives design), Types of interconnects (Characterization of interconnects, Linear arrays, 2D mesh and torus, cliques)

Unit 4 (9 Hours)
Applications: Matrix-matrix multiply, Odd-Even sorting, distributed histogram, Breadth First search, Dijkstra's algorithm

Essential/recommended readings

1. Grama, A., Gupta, A., Karypis, G., Kumar, V., *Introduction to Parallel Computing*, 2nd edition, Addison-Wesley, 2003.
2. Quinn, M. *Parallel Programming in C with MPI and OpenMP*, 1st Edition, McGraw-Hill, 2017.
3. Revdikar, L., Mittal, A., Sharma, A., Gupta, S., *A Naïve Breadth First Search Approach Incorporating Parallel Processing Technique For Optimal Network Traversal*, International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 5, May 2016.

Additional References

No additional references mentioned

Suggested Practical List :

Practical exercises such as

1. Implement Matrix-Matrix Multiplication in parallel using OpenMP
2. Implement distributed histogram Sorting in parallel using OpenMP
3. Implement Breadth First Search in parallel using OpenMP
4. Implement Dijkstra's Algorithm in parallel using OpenMP

DISCIPLINE SPECIFIC ELECTIVE COURSES

DISCIPLINE SPECIFIC ELECTIVE COURSE: Social Network Analytics

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
Social Network Analytics	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 01 Programming using Python, DSC03 Mathematics for Computing

Learning Objectives

The course introduces basic graph theory and draws distinction between graph as an abstract structure and real-life situation modelled as network. This course aims to expose the students to the strengths and capabilities of network analysis and their applications through the use of open source software.

Learning outcomes

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

- Model real life situation as networks
- Identify and apply quantitative network measures to characterize social networks at the local and global level
- Generate synthetic networks that satisfy properties of real world networks
- Discover, analyse and evaluate the intrinsic community structure of networks
- Model an information diffusion process for predictive analysis of networks

SYLLABUS OF DSE

Unit 1 (7 Hours)

Introduction to Social Network Analysis: Graph theory, random walk, degree distribution, mapping of real world situation into networks and applications of social network analysis, types of networks

Unit 2 (10 Hours)

Network Measures: Centrality measures, Page Rank, Hubs and Authority, Assortativity, Transitivity and Reciprocity, Similarity and Structural Equivalence

Unit 3 (10 Hours)

Network Models: Properties of Real-World Networks, Random Network Model, Small World Network Model, Preferential Attachment Model

Unit 4 (10 Hours)

Community Structure in Networks: Types of Communities, Community Detection algorithms and evaluation of communities obtained

Unit 5 (8 Hours)

Information Diffusion in Social Media: Information Cascades, Diffusion of Innovations, Basic Epidemic Models

Essential/recommended readings

1. Chakraborty T. *Social Network Analysis***,** 1st edition, Wiley India Pvt. Ltd., 2021.
2. Zafarani R., Abbasi M. A., Liu H. *Social Media Mining: An Introduction***,** 1st edition, Cambridge University Press, 2014.
3. Barabási A. L. , Pósfai M. *Network Science*, 1st edition, Cambridge University Press, 2016.

Additional References

1. Easley, Kleinberg J. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World***,** 1st edition, Cambridge University Press, 2012.

Suggested Practical List :

Practical exercises such as

Python Packages like igraph, NetworkX, NDlib etc. may be used for programming

1. Plot a weighted directed network such that node size and edge width is proportional to their degree and edge weight respectively
2. Compute and plot degree distribution of a real-world network. Also compute its local and global properties.
3. Generate three networks of 1000 nodes each using Random Network Model, Small World Network Model, Preferential Attachment Model and compare their characteristics.
4. Compute different centrality measures to identify top-N nodes and compare their ranks with those obtained by PageRank method.
5. Apply community detection algorithms on a small real-world network (e.g. Karate club) and compare modularity using bar plot. Also plot the communities revealed with different colors.
6. Simulate diffusion trends for different epidemic models and present results using appropriate visuals.

DISCIPLINE SPECIFIC ELECTIVE COURSE: Computer Graphics

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
Computer Graphics	**4**	**3**	**1**	**0**	Pass in Class XII	DSC 03 (Mathematics for Computing - I), DSC 04 Object Oriented Programming with C++/ GE 1a Programming using C++ / GE1b Programming with Python/ DSC 01 Programming using Python/ GE 3b Java Programming

Learning Objectives

This course introduces fundamental concepts of Computer Graphics with focus on modeling, rendering and interaction aspects of computer graphics. The course emphasizes the basic principles needed to design, use and understand computer graphics system.

Learning outcomes

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

- Describe Standard raster and vector scan devices as well as Graphical Input and output devices
- Implement algorithms for drawing basic primitives such as line, circle and ellipse.
- Implement algorithms for line clipping, polygon clipping and polygon filling.
- Implement a 3D object representation scheme, carryout 2D and 3D transformation, 3D projections
- Implement visible surface determination algorithms, Illumination models and surface rendering methods
- Implement a simple computer animation algorithm

SYLLABUS OF DSE

Unit 1 (8 Hours)

Introduction: Introduction to Graphics systems, Basic elements of Computer graphics, Applications of computer graphics. Architecture of Raster and Random scan display devices, input/output devices.

Unit 2 (8 Hours)
Drawing and clipping primitives: Raster scan line, circle and ellipse drawing algorithms, Polygon filling, line clipping and polygon clipping algorithms

Unit 3 (12 Hours)
Transformation and Viewing: 2D and 3D Geometric Transformations, 2D and 3D Viewing transformations (Projections- Parallel and Perspective), Vanishing points.

Unit 4 (9 Hours)
Geometric Modeling: Polygon Mesh Representation, Cubic Polynomial curves (Hermite and Bezier).

Unit 5 (8 Hours)
Visible Surface determination and Surface Rendering: Z-buffer algorithm, List-priority algorithm and area subdivision algorithm for visible surface determination. Illumination and shading models, RGB Color model and Basics of Computer Animation.

Essential/recommended readings

1. Hearn, D & Baker, M.P. *Computer Graphics*, 2nd edition, Prentice Hall of India, 2009.
2. Foley, J. D., Dam, A.V, Feiner, S. K., & Hughes, J. F. *Computer Graphics: Principles and Practice in C*, 2nd edition, Pearson education, 2002.
3. Rogers, D. F. *Mathematical Elements for Computer Graphics*, 2nd edition, McGraw Hill Education, 2017.

Additional References

1. Bhattacharya, S. *Computer Graphics*, Oxford University Press, 2018.
2. Marschner, S., & Shirley, P. *Fundamentals of Computer Graphics,* 4th edition CRC Press, 2017.

Suggested Practical List :

Practical exercises such as

1. Write a program to implement Bresenham's line drawing algorithm.
2. Write a program to implement a midpoint circle drawing algorithm.
3. Write a program to clip a line using Cohen and Sutherland line clipping algorithm.
4. Write a program to clip a polygon using Sutherland Hodgeman algorithm.
5. Write a program to fill a polygon using the Scan line fill algorithm.
6. Write a program to apply various 2D transformations on a 2D object (use homogeneous Coordinates).
7. Write a program to apply various 3D transformations on a 3D object and then apply parallel and perspective projection on it.
8. Write a program to draw Hermite /Bezier curve.

DISCIPLINE SPECIFIC ELECTIVE COURSE: Deep Learning

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
Deep Learning	**4**	**3**	**0**	**1**	Pass in Class XII	DSC03 Mathematics for Computing - I, DSC17 Machine Learning

Learning Objectives

The objective of this course is to introduce students to deep learning algorithms and their applications in order to solve real problems.

Learning outcomes

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

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

SYLLABUS OF DSE

Unit 1 (8 Hours)

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

Unit 2 (8 Hours)

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

Unit 3 (8 Hours)

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

Unit 4 (8 Hours)
Recurrent neural networks (RNNs): Sequence modeling using RNNs, Backpropagation through time, LongShort Term Memory (LSTM), Bidirectional RNN, Bidirectional LSTM
Unit 5 (8 Hours)
Unsupervised deep learning: Autoencoders, Generative Adversarial Networks.

Unit 6 (5 Hours)
Applications: Computer vision, Speech recognition and NLP.

Essential/recommended readings

1. Ian Goodfellow, Yodhua Bengio and Aaron Courville, *Deep Learning,* MIT Press Book**,** 2016.
2. Francois Chollet, *Deep Learning with python***,** *2nd edition*, Meaning Publications Co, 2021.

Additional References

1. Bunduma, N., *Fundamentals of Deep Learning*, 1st edition, O'reilly Books, 2017.
2. Heaton, J., *Deep Learning and Neural Networks*, 1st edition, Heaton Research Inc., 2015.

Suggested Practical List :

Practical exercises such as

The following practicals are to be conducted using Python.

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

DISCIPLINE SPECIFIC ELECTIVE COURSE: Ethical Hacking

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
Ethical Hacking	**4**	**3**	**0**	**1**	Pass in Class XII	Any Programming Language at plus 2 level or above

Learning Objectives

The objective of this course is to enable students to be part of such a team that can conduct the security assessment of an organization through the process of ethical hacking. This course will introduce the students, the idea of security assessment of systems and networks under investigation and how to perform them under the legal and ethical framework. Further, this course will outline the importance of various stages of ethical hacking, including but not limited to tasks such as penetration testing, and usage of various tools at each stage.

Learning outcomes

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

- Understand and acknowledge the relevance of legal, ethical, and professional challenges faced by an ethical hacker.
- Apply fundamental principles of system, application, and network security to ethically attack / penetrate the system to uncover the security flaws.
- Perform evaluation of security systems through a systematic ethical hacking process and recommend countermeasures to improve security.
- Understand and use various tools and techniques used in various stages of the ethical hacking process.

SYLLABUS OF DSE

Unit 1 (4 Hours)

Introduction: Overview of information security threats and attack vectors, vulnerability assessment and penetration testing concepts, information security controls, security laws and standards. OWASP.

Unit 2 (6 hours)

Footprinting and Reconnaissance: Introduction to network reconnaissance tools such ipconfig, ifconfig, domain tools, nmap, Wireshark, etc.

Unit 3 (6 hours)

Scanning and Enumeration: Network penetration testing, Password cracking techniques and countermeasures, NetBIOS tools

Unit 4 (6 hours)

Gaining and Maintaining Access: Network level attacks and countermeasures, Metasploit framework, Burp Suite

Unit 5 (6 hours)

Exploitation and Covering Tracks: Privilege escalation, social Engineering**,** identity theft, countermeasures, Covering tracks using attrib command and creating Alternate Data Stream (ADS) in Windows, Erasing evidence from Windows logs, Strategies for maintaining access.

Unit 6 (6 hours)

Advanced stages: Denial of service, Session hijacking, hacking web servers, hacking web applications, sql injection etc.

Unit 7 (6 hours)

NIST Cybersecurity framework and ISO standards: NIST cybersecurity framework, Cyber Kill chain, ISO/IEC 27001 and related standards.

Unit 8 (5 Hours)

Cyber Defense and Reporting: Preparing vulnerability assessment reports, presenting post testing findings, preparing recommendations

Essential/recommended readings

1. Patrick Engbretson, The Basics of Hacking and Penetration Testing, 2nd Edition, Syngress, 2013.
2. Georgia Weidman, Penetration TEsting: A Hands-On Introduction to Hacking, 1st Edition, No Starch Press, 2014.

Additional References

1. Peter Kim, The Hacker Playbook 3: Practical Guide to Penetration Testing, Zaccheus Entertainment, 2018.
2. Jon Erickson, Hacking: The Art of Exploitation, No Starch Press, 2008.
3. Online Resources:
    a. https://www.sans.org/cyberaces/
    b. https://skillsforall.com/
    c. https://www.hackingloops.com/ethical-hacking/

Suggested Practical List

Practical exercises such as

Perform the following activities.

(NOTE: Exercise extra caution while performing these exercises and codes)

1. Perform various Virtual Machine based exercises on https://vulnhub.com/
2. Perform exercises from https://www.hacker101.com/
3. Follow the lessons and activities from https://www.hackingloops.com/ethical-hacking/
4. Activities on Google site for hacking https://google-gruyere.appspot.com/
5. ACtivities on OWASP WebGoat https://github.com/WebGoat/WebGoat

DISCIPLINE SPECIFIC ELECTIVE COURSE: Cloud Computing

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
Cloud Computing	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objectives

The objective of an undergraduate cloud computing course is to provide students with a comprehensive understanding of cloud computing technologies, services, and applications.

Learning outcomes

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

- Knowledge of the fundamental concepts and principles of cloud computing, including virtualization, scalability, reliability, and security.
- Ability to design, develop, and deploy cloud-based applications using popular cloud platforms and services.
- Familiarity with cloud computing architectures, including Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).
- Understanding of the economic, legal, and ethical implications of cloud computing, including issues related to data privacy, ownership, and security.
- Ability to evaluate and select cloud-based solutions based on their technical, economic, and business requirements.
- Understanding of the broader societal and environmental impacts of cloud-based services and applications.

SYLLABUS OF DSE

Unit 1

Overview of Computing Paradigm: Recent trends in Computing : Grid Computing, Cluster Computing, Distributed Computing, Utility Computing, Cloud Computing,

Unit 2

Introduction to Cloud Computing: Introduction to Cloud Computing, History of Cloud Computing, Cloud service providers, Benefits and limitations of Cloud Computing,

Unit 3

Cloud Computing Architecture: Comparison with traditional computing architecture (client/server), Services provided at various levels, Service Models- Infrastructure as a Service(IaaS), Platform as a Service(PaaS), Software as a Service(SaaS), How Cloud Computing Works, Deployment Models- Public cloud, Private cloud, Hybrid cloud, Community cloud, Case study of NIST architecture.

Unit 4

Case Studies: Case study of Service model using Google Cloud Platform (GCP), Amazon Web Services (AWS), Microsoft Azure, Eucalyptus.

Unit 5

Cloud Computing Management: Service Level Agreements(SLAs), Billing & Accounting, Comparing Scaling Hardware: Traditional vs. Cloud, Economics of scaling.

Unit 6

Cloud Computing Security: Infrastructure Security- Network level security, Host level security, Application level security, Data security and Storage- Data privacy and security Issues, Jurisdictional issues raised by Data location, Authentication in cloud computing.

Essential/recommended readings

1. Thomas Erl, Ricardo Puttini and Zaigham Mahmood, Cloud Computing: Concepts, Technology and Architecture, Publisher: PHI, 2013.
2. Rajkumar Buyya, James Broberg, and Andrzej Goscinski, Cloud Computing: Principles and Paradigms, Wiley, 2013.
3. Boris Scholl, Trent Swanson, and Peter Jausovec, Cloud Native: Using Containers, Functions, and Data to Build Next-Generation Applications, Publisher : Shroff/O'Reilly, 2019.

Additional References

1. *Cloud Computing Bible,* Barrie Sosinsky*, Wiley-India,* 2010
2. *Cloud Computing: Principles and Paradigms,* Editors: Rajkumar Buyya, James Broberg, Andrzej M. Goscinski*, Wile,* 2011
3. *Cloud Computing: Principles, Systems and Applications, Editors:* Nikos Antonopoulos, Lee Gillam*, Springer,* 2012
4. *Cloud Security: A Comprehensive Guide to Secure Cloud Computing,* Ronald L. Krutz, Russell Dean Vines*, Wiley-India,* 2010

Suggested Practical List:

Practical exercises such as

1. Create virtual machines that access different programs on same platform.
2. Create virtual machines that access different programs on different platforms

3. Working on tools used in cloud computing online:
    a. Storage
    b. Sharing of data
    c. manage your calendar, to-do lists,
    d. a document editing tool
4. Exploring Google cloud
5. Exploring Microsoft cloud
6. Exploring Amazon cloud

COMMON POOL OF GENERIC ELECTIVES (GE) COURSES

GENERIC ELECTIVES (GE-6a): Computer Networks

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
GE6a Computer Networks	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 04 Object Oriented Programming with C++/ GE 1a Programming using C++ / GE1b Programming with Python/ DSC 01 Programming using Python/ GE 3b Java Programming

Learning Objectives

The course objectives of this paper are to understand the concepts behind computer networks and data communication. learn the different types of networks, network topologies and their characteristics. learn the working of protocols used at various layers. understand the utility of different networking devices.

Learning outcomes

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

- differentiate between various types of computer networks and their topologies.
- understand the difference between the OSI and TCP/IP protocol suit.
- distinguish between different types of network devices and their functions.
- design/implement data link and network layer protocols in a simulated networking environment.

SYLLABUS OF GE-6a

Unit 1

Introduction: Types of computer networks, Internet, Intranet, network topologies (bus, star, ring, mesh, tree, hybrid topologies), network classifications. layered architecture approach, OSI Reference Model, TCP/IP Reference Model. Transmission Modes: simplex, half duplex and full duplex.

Unit 2

Physical Layer: Analog signal, digital signal, the maximum data rate of a channel, transmission media (guided transmission media, wireless transmission, satellite

communication), multiplexing (frequency division multiplexing, time-division multiplexing, wavelength division multiplexing). Guided Media (Wired) (Twisted pair, Coaxial Cable, Fiber Optics. Unguided Media (Radio Waves, Infrared, Micro-wave, Satellite).

Unit 3

Data Link and MAC Layer: Data link layer services, error detection and correction techniques, error recovery protocols (stop and wait, go back n, selective repeat), multiple access protocols with collision detection, MAC addressing, Ethernet, data link layer switching, point-to-point protocol.

Unit 4

Network layer: Networks and Internetworks, virtual circuits and datagrams, addressing, subnetting, Dijkstra Routing algorithm, Distance vector routing, Introduction to Network Layer protocol- (ARP, IPV4, ICMP).

Unit 5

Introduction to Transport and Application Layer: Introduction to Process to process Delivery- (client-server paradigm, connectionless versus connection-oriented service); User Datagram Protocols, TCP/IP protocol, Flow Control. FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol), Telnet (Remote login protocol), WWW (World Wide Web), HTTP (HyperText Transfer Protocol), URL (Uniform Resource Locator).

Essential/recommended readings

1. Tanenbaum, A.S. & Wetherall, D.J., *Computer Networks*, *5th edition*, Pearson Education, 2012.

2. Forouzan, B. A., *Data Communication and Networking*, *4th edition*, McGraw-Hill Education, , 2017.

Additional References

1. Comer, D. E., *Computer Networks and Internet,* 6th edition*,* Pearson Publication, 2015.

2. (ii) Stallings, W., *Data and Computer Communications*, 10th edition, Pearson education India, 2017.

Suggested Practical List :

Practical exercises such as

Introduce students to CISCO Packet Tracer and do the following:

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

GENERIC ELECTIVES (GE-6b): Internet Technologies: Web App Design and Development

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
GE6b: Internet Technologies: Web App Design and Development	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objectives

The course aims at:

- Develop understanding of Web Development Architecture.
- Using React components in Web applications
- Introduce REST APIs Design
- Understanding of Angular Architecture, data-binding and dependency injection
- Understand form validations and application of templates

Learning outcomes

On successful completion of the course students will be able to

- Develop interfaces for single page applications
- Develop a complete client side solutions using angular js
- Develop a RESTful web services.
- Apply form validations

SYLLABUS OF GE-6b

Unit 1

Introduction to React: Definition of React, React library, React Developer tools, Introduction to ES6, Declaring variables, Arrow Functions, Objects and Arrays, modules, Introduction to AJAX, Functions in AJAX Pure React: Page setup, virtual DOM, React Element, React DOM, Constructing Elements with Data, React Components, DOM Rendering, First React Application using Create React App, React with JSX, React Element as JSX Props, State and Component Tree: Property Validation, Validating Props with createClass, Default Props, ES6 Classes and stateless functional components, React state management, State within the component tree, state vs props, Forms in React

Unit 2

Rest APIs: JSON: Introduction, Syntax, Data Types, Objects, Schema. REST API:

Introduction, WRML, REST API Design, Identifier Design with URIs, Interaction Design with HTTP, Representation Design, Caching, Security.

Unit 3

Angular.js.: Introduction to Angular: Angular architecture; introduction to components, component interaction and styles; templates, interpolation and directives; forms, user input, form validations; data binding and pipes; retrieving data using HTTP; Angular modules

Essential/recommended readings

1. D. Brad, B. Dayley and C. Dayley, *Node. js, MongoDB and Angularjs Web Development: The definitive guide to using the MEAN stack to build web applications*, 2nd edition, Addison-Wesley, 2018.
2. D. Herron, *Node.js Web Development*, 5th edition, Packt Publishing, 2020.
3. A. Banks and E. Porcello, *Learning React: Functional Web Development with React and Redux*, 1st edition, O'Reilly, 2017.
4. M. Masse, *REST API – Design Rulebook*, 1st edition, O'Reilly, 2011.

Additional References

No additional references mentioned.

Suggested Practical List :

Practical exercises such as

1. Angular.js:
a. Build a simple Angular.js application that displays a list of items.
b. Create a form in Angular.js to add new items to the list.
c. Implement filtering and sorting functionality in Angular.js to manipulate the displayed list.
d. Integrate Angular.js with a RESTful API to fetch data and display it in the application.
e. Implement authentication and authorization using Angular.js routing and services.

2. React:
a. Create a basic React component that displays "Hello, World!" on the screen.
b. Build a React application that fetches data from a REST API and renders it in a list.
c. Implement form handling in React to create, update, and delete items from the list.
d. Create a search functionality using React to filter the displayed list based on user input.
e. Implement routing in React to navigate between different pages within the application.

3. REST API:
a. Build a simple REST API using a framework like Node.js and Express.
b. Create endpoints to perform CRUD operations (Create, Read, Update, Delete) on a specific resource (e.g., users, products).
c. Implement authentication and authorization mechanisms using JSON Web Tokens (JWT) to secure the API.
d. Develop endpoints that handle file uploads and downloads.

GENERIC ELECTIVES (GE-6c): Artificial Intelligence

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
GE6c: Artificial Intelligence	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 04 Object Oriented Programming with C++/ GE 1a Programming using C++ / GE1b Programming with Python/ DSC 01 Programming using Python/ GE 3b: Java Programming

Learning Objectives

The course objectives of this course are to:

- To introduce basic concepts and techniques of Artificial Intelligence (AI).
- To apply informed search techniques for different applications.
- To learn various knowledge representation techniques and writing Prolog programs.
- To learn about the latest techniques for developing AI systems.

Learning outcomes

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

- identify problems that are amenable to solutions by specific AI methods.
- state the utility of different types of AI agents.
- apply different informed search techniques for solving real world problems.
- use knowledge representation techniques for AI systems.

SYLLABUS OF GE-6c

Unit 1 6 Hours

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

Unit 2 12 Hours

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

Unit 3 16 Hours

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

Unit 4 8 Hours

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

Unit 5 3 Hours

AI The Present and the Future: Symbolic AI, Data-driven AI and Machine Learning, Introduction to Machine Learning and Deep Learning based AI, some applications of symbolic and data driven AI, Interpretable and Explainable AI, Ethics of AI: benefits and risks of AI.

Essential/recommended readings

1. Russell, Stuart, J. and Norvig, Peter, *Artificial Intelligence - A Modern Approach*, Pearson, 4th edition, 2020..
2. Bratko, Ivan, *Prolog Programming for Artificial Intelligence*, Addison-Wesley, Pearson Education, 4th edition, 2012.
3. Patterson, DAN,W, *Introduction to A.I. and Expert Systems* – PHI, 2007.
4. Clocksin, W., F. and Mellish, *Programming in PROLOG*, 5th edition, Springer, 2003.

Additional references

1. Kaushik, Saroj, *Artificial Intelligence*, Cengage Learning India, 2011.
2. Rich, Elaine and Knight, Kelvin, *Artificial Intelligence*, 3rd edition, Tata McGraw Hill, 2010

Practical List :

Practical exercises such as

14. Write a program in Prolog to implement TowerOfHanoi(N) where N represents the number of disks.
15. Write a program to implement the Hill climbing search algorithm in Prolog.
16. Write a program to implement the Best first search algorithm in Prolog.
17. Write a program to implement A* search algorithm in Prolog.
18. Write a program to implement the min-max search algorithm in Prolog.
19. Write a program to solve the Water-Jug Problem in Prolog.
20. Implement sudoku problem (minimum 9×9 size) using constraint satisfaction in Prolog.
21. Write a Prolog program to implement the family tree and demonstrate the family relationship.
22. Write a Prolog program to implement knowledge representation using frames with appropriate examples.
23. Write a Prolog program to implement conc(L1, L2, L3) where L2 is the list to be appended with L1 to get the resulted list L3.
24. Write a Prolog program to implement reverse(L, R) where List L is original and List R is reversed list.
25. Write a Prolog program to generate a parse tree of a given sentence in English language assuming the grammar required for parsing.
26. Write a Prolog program to recognize context free grammar $a^n b^n$.

(Computer Science Courses for Undergraduate Programme of study with Computer Science discipline as one of the three Core Disciplines)

DISCIPLINE SPECIFIC CORE COURSE (DSC06): Computer Networks

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
DSC06: Computer Networks	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 04 Object Oriented Programming with C++/ GE 1a Programming using C++ / GE1b Programming with Python/ DSC 01 Programming using Python/ GE 3b: Java Programming

Learning Objectives
The course objectives of this paper are to:

- Understand the concepts behind computer networks and data communication.
- Learn the different types of networks, network topologies and their characteristics.
- Learn the working of protocols used at various layers.
- Understand the utility of different networking devices.

Learning outcomes

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

- differentiate between various types of computer networks and their topologies.
- understand the difference between the OSI and TCP/IP protocol suit.
- distinguish between different types of network devices and their functions.
- design/implement data link and network layer protocols in a simulated networking environment.

SYLLABUS OF DSC06

Unit 1 (8 hours)
Introduction: Types of computer networks, Internet, Intranet, network topologies (bus, star, ring, mesh, tree, hybrid topologies), network classifications. layered architecture approach,

OSI Reference Model, TCP/IP Reference Model. Transmission Modes: simplex, half duplex and full duplex.

Unit 2 (9 hours)
Physical Layer: Analog signal, digital signal, the maximum data rate of a channel, transmission media (guided transmission media, wireless transmission, satellite communication), multiplexing (frequency division multiplexing, time-division multiplexing, wavelength division multiplexing). Guided Media (Wired) (Twisted pair, Coaxial Cable, Fiber Optics. Unguided Media (Radio Waves, Infrared, Micro-wave, Satellite).

Unit 3 (10 hours)
Data Link and MAC Layer: Data link layer services, error detection and correction techniques, error recovery protocols (stop and wait, go back n, selective repeat), multiple access protocols with collision detection, MAC addressing, Ethernet, data link layer switching, point-to-point protocol.

Unit 4 (8 hours)
Network layer: Networks and Internetworks, virtual circuits and datagrams, addressing, subnetting, Dijkstra Routing algorithm, Distance vector routing, Network Layer protocol- (ARP, IPV4, ICMP).

Unit 5 (10 hours)
Transport and Application Layer: Process to process Delivery- (client-server paradigm, connectionless versus connection-oriented service); User Datagram Protocols, TCP/IP protocol, Flow Control. FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol), Telnet (Remote login protocol), WWW (World Wide Web), HTTP (HyperText Transfer Protocol), URL (Uniform Resource Locator).

Essential/recommended readings

1. Tanenbaum, A.S. & Wethrall, D.J.. *Computer Networks*, 5th edition, Pearson Education, 2012.

2. Forouzan, B. A.. *Data Communication and Networking*, 4th edition, McGraw-Hill Education, 2017.

Additional References

1. Comer, D. E.. *Computer Networks and Internet,* 6th edition*,* Pearson education, 2015.

2. Stallings, W., *Data and Computer Communications,* 10th edition, Pearson education India, 2017.

Suggested Practical List :

Practical exercises such as

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

1. To Study basic network command and Network configuration commands.
2. To study and perform PC to PC communication.

3. To create Star topology using Hub and Switch.
4. To create Bus, Ring, Tree, Hybrid, Mesh topologies.
5. Perform an initial Switch configuration.
6. Perform an initial Router configuration.
7. To implement Client Server Network.
8. To implement connection between devices using router.
9. To perform remote desktop sharing within LAN connection.