Annexure-LXXXIII
Item No. 4-1-4
EC dated 18.08.2022

UNIVERSITY OF DELHI

Bachelor in Computer Science (Honours)

(SEMESTER-I)
based on

Undergraduate Curriculum Framework 2022 (UGCF)

(Effective from Academic Year 2022-23)

University of Delhi

Semester –I

DSCs:-

Course Title	Nature of the Course	Total Credits	Components			Eligibility Criteria/ Prerequisite	Contents of the course and references may be seen at
			L	T	P		
Programming using Python	DSC-1	4	3	0	1	Class XII Pass	Annexure – I
Computer System Architecture	DSC-2	4	3	0	1	Class XII Pass	Annexure -II
Mathematics for computing	DSC-3	4	3	0	1	Class XII Pass	Annexure -III

GEs:-

Course Title	Nature of the Course	Total Credits	Components			Eligibility Criteria/ Prerequisite	Contents of the course and references may be seen at
			L	T	P		
Programming using C++	GE1a	4	3	0	1	Class XII Pass	Annexure – IV
Programming with Python	GE1b	4	3	0	1	Class XII Pass	Annexure -V

Department of Computer Science

University of Delhi

Curriculum

Bachelor in Computer Science (Honors)

under UGCF 2022

Approved in
PG Committee meeting held on May 4, 2022
Faculty of Mathematical Sciences meeting held on May 25, 2022
Standing Committee on Academic Matters

Table of Contents

Table of Core Courses

Semester	DSC-No.	Title	L	T*	P*	Total credits	Prerequisites
I	DSC 01	Programming using Python	3	0	1	4	Pass in Class XII
I	DSC 02	Computer System Architecture	3	0	1	4	Pass in Class XII
I	DSC 03	Mathematics for computing	3	0	1	4	Pass in Class XII
II	DSC 04	Object Oriented Programming with C++	3	0	1	4	Pass in Class XII
II	DSC 05	Discrete Mathematical Structures	3	0	1	4	Pass in Class XII
II	DSC 06	Probability for	3	0	1	4	Pass in Class XII

		Computing					
III	**DSC 07**	**Data Structures**	**3**	**0**	**1**	**4**	DSC 04/ a course in C/C++ at plus 2 level/**
III	**DSC 08**	**Operating Systems**	**3**	**0**	**1**	**4**	DSC 04/a course in C/C++ at plus 2 level/**
III	**DSC 09**	**Numerical Optimization**	**3**	**0**	**1**	**4**	Pass in Class XII
IV	**DSC10**	**Design and Analysis of Algorithms**	**3**	**0**	**1**	**4**	DSC07
IV	**DSC 11**	**Database Management System**	**3**	**0**	**1**	**4**	DSC01/ a course in Python at plus 2 level/** , DSC08
IV	**DSC 12**	**Computer Networks**	**3**	**0**	**1**	**4**	DSC04/a course in C/C++ at plus 2 level/** , DSC 07, DSC 08
V	**DSC 13**	**Algorithm**	**3**	**0**	**1**	**4**	DSC07,

		s and Advanced Data Structures					DSC10
V	**DSC 14**	**Theory of Computation**	**3**	**0**	**1**	**4**	DSC04/a course in C/C++ at plus 2 level/** DSC05
V	**DSC 15**	**Software Engineering**	**3**	**0**	**1**	**4**	DSC01,/ DSC04,/ a course in C/C++/Python at plus 2 level/**
VI	**DSC 16**	**Artificial Intelligence**	**3**	**0**	**1**	**4**	DSC 01 DSC03 DSC 06 DSC 07 DSC 09 DSC10 DSC14
VI	**DSC 17**	**Machine Learning**	**3**	**0**	**1**	**4**	DSC 01 DSC03 DSC 06 DSC 07 DSC 09 DSC10
VI	**DSC 18**	**Introduction to Parallel Programming**	**3**	**0**	**1**	**4**	DCS 02, DSC04, DSC 07, DSC 08
VII	**DSC 19**	**Compiler Design**	**3**	**0**	**1**	**4**	DSC-14:

VIII	**DSC 20**	**Information Security**	**3**	**0**	**1**	**4**	DSC 01 DSC04, DSC 07 DSC 08 DSC 11 DSC 12

Note: Batch size for Practicals will be (8-10) and Tutorials will be (12-15).

Syllabi of Core Courses

This section gives the detailed syllabus of the core courses. Each course describes the course objective, learning outcomes, the units and the reading material. The reading material has 2 -3 components: main resource(/s), additional text material, and online resources. Main resources are kept to a minimum possible and no more than 3. Additional resources and the online material may be used to enhance the knowledge of the subject.

DSC 01: Programming using Python

Course Objective

This course is designed as the first course that introduces programming concepts using Python to Computer Science students. The course focuses on the development of Python programming to solve problems of different domains. It also introduces the concept of object- oriented programming.

Course Learning Outcomes

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

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

Syllabus

Unit 1 Introduction to Programming: Problem solving strategies; Structure of a Python program; Syntax and semantics; Executing simple programs in Python.

Unit 2 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 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 Object Oriented Programming: Introduction to classes, objects and methods; Standard libraries.

Unit 5 File and exception handling: File handling through libraries; Errors and exception handling.

References

1. Taneja, S., Kumar, N. *Python Programming- A modular Approach*, 1st edition, Pearson Education India, 2018.
2. Balaguruswamy E. *Introduction to Computing and Problem Solving using Python,* 2nd edition, McGraw Hill Education, 2018.

Additional References

(i) Brown, Martin C. *Python: The Complete Reference*, 2nd edition, McGraw Hill Education, 2018.

(ii) Guttag, J.V. *Introduction to computation and programming using Python,* 2nd edition, MIT Press, 2016.

Suggested Practical List

1. WAP to find the roots of a quadratic equation

2. WAP to accept a number ‘n’ and
    a. Check if ’n’ is prime
    b. Generate all prime numbers till ‘n’
    c. 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

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

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

4. 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)
5. 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.
6. WAP to swap the first n characters of two strings.
7. 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.
8. 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

9. WAP to read a file and
    a. Print the total number of characters, words and lines in the file.
    b. Calculate the frequency of each character in the file. Use a variable of dictionary type to maintain the count.
    c. Print the words in reverse order.
    d. Copy even lines of the file to a file named 'File1' and odd lines to another file named 'File2'.
10. 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.
11. Write a function that prints a dictionary where the keys are numbers between 1 and 5 and the values are cubes of the keys.
12. 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
13. 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.

DSC 02: Computer System Architecture

Course Objective

This course introduces the students to the fundamental concepts of digital computer organization, design and architecture. It aims to 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.

Course Learning Outcomes

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

1. Design Combinational Circuits using basic building blocks. Simplify these circuits

using Boolean algebra and Karnaugh maps. Differentiate between combinational circuits and sequential circuits.

2. Represent data in binary form, convert numeric data between different number systems and perform arithmetic operations in binary.
3. Determine various stages of instruction cycle, pipelining and describe interrupts and their handling.
4. Explain how CPU communicates with memory and I/O devices and distinguish between different types of processors.
5. Simulate the design of a basic computer using a software tool.

Syllabus

Unit 1 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 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 Data Representation and Basic Computer Arithmetic: Number System, r and (r-1)'s Complements, data representation and arithmetic operations.

Unit 4 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 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 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.

References

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

Additional References

(i) Mano, M. *Digital Design*, Pearson Education Asia, 1995.

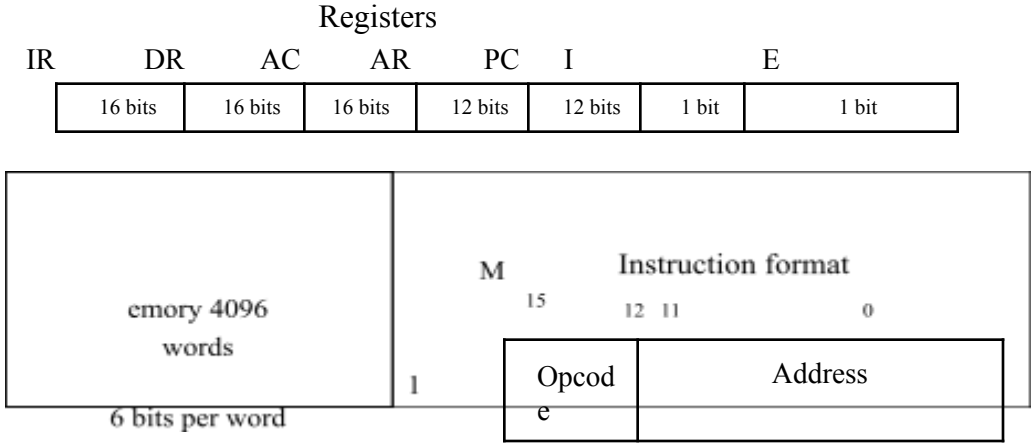
(ii) Null, L., & Lobur, J. *The Essentials of Computer Organization and Architecture*. 5th edition, (Reprint) Jones and Bartlett Learning, 2018.

(iii) Stallings, W. *Computer Organization and Architecture Designing for Performance* 8th edition, Prentice Hall of India, 2010.

Suggested Practical List

(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 Addressi	CLA	7800
ADD	1xxx		CLE	7400
LDA	2xxx		CMA	7200
STA	3xxx		CME	7100

BUN	4xxx	ng	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.

DSC 03: Mathematics for computing

Course Objective

This course introduces the students to the fundamental concepts and topics of linear algebra and vector calculus, 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.

Course Learning Outcomes

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

1. Perform operations on matrices and sparse matrices
2. Compute the determinant, rank and eigenvalues of a matrix
3. Perform diagonalization

4. Perform operations on vectors, the dot product and cross product
5. Represent vectors geometrically and calculate the gradient, divergence, curl
6. Apply linear algebra and vector calculus to solve problems in sub-disciplines of computer science.

Syllabus

Unit 1 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 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 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 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.

References

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

2016.

Additional References

(i) Deisenroth, Marc Peter, Faisal A. Aldo and Ong ChengSoon. *Mathematics for Machine Learning*, 1st Edition, Cambridge University Press, 2020.

(ii) Lipschutz Seymour and Lipson Marc. *Schaum's Outline of Linear Algebra*, 6th Edition, McGraw Hill, 2017.

Suggested Practical List

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.

Department of Computer Science
University of Delhi

General Electives (GEs)

Offered by
Department of Computer Science
Under UGCF 2022

Approved in
UG Committee meeting held on May17, 2022
Faculty of Mathematical Sciences meeting held on May 25, 2022
Standing Committee on Academic Matters

1. Table of additional GEs

Table of additional GEs

Semester	GE -No.	Title	L	T*	P*	Total credits	Prerequisites
I	**GE1a**	Programming using C++	**3**	**0**	**1**	**4**	Pass in Class XII
	GE1b	Programming with Python	**3**	**0**	**1**	**4**	Pass in Class XII
II	**GE2a**	Data Analysis and Visualization using Python	**3**	**0**	**1**	**4**	A course in Python
	GE2b	Data Analysis and Visualization using Spreadsheet	**3**	**0**	**1**	**4**	Pass in Class XII
III	**GE3a**	Database Management Systems					Pass in Class XII
	GE3b	Java Programming					Pass in Class XII
IV	**GE4a**	Data Structures using C++	**3**	**0**	**1**	**4**	A course in C++
	GE4b	Introduction to Web Programming	**3**	**0**	**1**	**4**	Pass in Class XII
V	**GE5a**	Operating Systems	**3**	**0**	**1**	**4**	Knowledge of Programming and Data Structures

	GE5b	Advanced Web Programming	**3**	**0**	**1**	**4**	Knowledge of HTML, CSS
	GE5c	Java Based Web App Development	**3**	**0**	**1**	**4**	A course in C++/Java
VI	**GE6a**	Computer Networks	**3**	**0**	**1**	**4**	Knowledge of C/C++/Java/Python
	GE6b	Internet Technologies: Web App Design and Development	**3**	**0**	**1**	**4**	A course in C++/Java
	GE6c	Artificial Intelligence	**3**	**0**	**1**	**4**	Knowledge of any programming language, basics of Algorithms and Data structures
VII	**GE7a**	Information Security	**3**	**0**	**1**	**4**	Knowledge of Programming, Computer System Architecture, Database Management Systems, Data Structures, Algorithms, Operating systems, Computer Networks.
	GE7b	Design and Analysis of Algorithms					Knowledge of Programming and Data Structures
	GE7c	Internet Technologies : Mobile App Design and Development	**3**	**0**	**1**	**4**	A course in C++/Java
VIII	**GE8a**	Machine Learning	**3**	**1**	**0**	**4**	A course in probability, statistics, linear algebra and multivariate analysis
	GE8b	Digital Marketing	**3**	**0**	**1**	**4**	Python and HTML

		and Social Media Analytics (under preparation)					
	GE8c	Introduction to Parallel Programming	**3**	**1**	**0**	**4**	Knowledge of Computer System Architecture, C++, Knowledge of Basic Data Structures and Algorithms, Operating Systems

Note:

1. Wherever C++/Java/Python is a prerequisite, C++/Java/Python of plus 2 level is acceptable.
2. Batch size for Practicals will be (8-10) and Tutorials will be (12-15).

It was recommended to add more electives in Semester VII and VIII in due course of time.

GE1a: Programming using C++

Course Objective

This course is designed to develop structured as well as object-oriented programming skills using C++ programming language. The course provides a complete understanding of the object-oriented programming features, namely Encapsulation, Abstraction, Inheritance and Polymorphism along with an in-depth knowledge of C++ constructs.

Course Learning Outcomes

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

1. Explain significance of object oriented paradigm.
2. Solve programming problems using C++.
3. Create classes and reuse them.

Syllabus

Unit 1 Introduction to C++: Overview of Procedural and Object-Oriented Programming, Using main() function, Header Files, Compiling and Executing Simple Programs in C++.

Unit 2 Programming Fundamentals: Data types, Variables, Operators, Expressions,

Arrays, Keywords, Decision making constructs, Iteration, Type Casting, Input-output statements, Functions, Command Line Arguments/Parameters

Unit 3 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

References

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.

Additional References

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

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

Suggested Practical list

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

   S = 1 - 2 + 3 - 4 + … n

   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 display the following pattern:

   1

   22

   333

   4444

   55555

   The number of rows n, is to be taken from the user.

3. Write a program to compute the factors of a given number.

4. Write a menu driven program to perform the following operations on an array:
    a. Find the minimum, maximum and average of the array elements
    b. Search an element in the array using linear and binary search
5. Write a menu driven program to perform the following operations on a string:
    a. Calculate length of the string
    b. Check whether the first character of every word in the string is in uppercase or not
    c. Reverse the string
6. Create a class Triangle. Include overloaded functions for calculating the area of a triangle.
7. Create a template class TwoDim which contains x and y coordinates. Define default constructor, parameterized constructor and void print() function to print the co-ordinates. Now reuse this class in ThreeDim adding a new dimension as z. Define the constructors and void print() in the subclass. Implement main() to show runtime polymorphism.

GE1b: Programming with Python

Course Objective

This course is designed as the first course that introduces programming concepts using Python to students. The course focuses on the development of Python programming to solve problems of different domains. It also introduces the concept of object- oriented programming.

Course Learning Outcomes

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

1. Understand the basics of programming language
2. Develop, document, and debug modular Python programs.
3. Apply suitable programming constructs and built-in data structures to solve a

problem.

4. Use and apply various data objects in Python.
5. Handle files

Syllabus

Unit 1 Introduction to Programming: Problem solving strategies; Structure of a Python program; Syntax and semantics; Executing simple programs in Python.

Unit 2 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 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 File and exception handling: File handling through libraries; Errors and exception handling.

References

1. Taneja, S., Kumar, N., *Python Programming- A modular Approach*, Pearson Education India, 2018.
2. Balaguruswamy E., *Introduction to Computing and Problem Solving using Python,* 2nd edition, McGraw Hill Education, 2018.

Additional References

(i) Brown, Martin C., *Python: The Complete Reference*, 2nd edition, McGraw Hill Education, 2018.

(ii) Guttag, J.V. *Introduction to computation and programming using Python,* 2nd edition, MIT Press, 2016.

Suggested Practical List

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

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

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

4. 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)
5. 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.
6. WAP to swap the first n characters of two strings.

7. 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.
8. 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
9. WAP to read a file and
    a. Print the total number of characters, words and lines in the file.
    b. Calculate the frequency of each character in the file. Use a variable of dictionary type to maintain the count.
    c. Print the words in reverse order.
    d. Copy even lines of the file to a file named ‘File1’ and odd lines to another file named ‘File2’.
10. Write a function that prints a dictionary where the keys are numbers between 1 and 5 and the values are cubes of the keys.
11. 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
12. 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.

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.

Common Pool of Generic Electives (GE) Courses
Offered by Department of Computer Sciences
Category-IV

GENERIC ELECTIVES (GE-2a): 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
		Lecture	Tutorial	Practical/ Practice		
GE2a Data Analysis and Visualization using Python	**4**	**3**	**0**	**1**	**Class XII pass with Mathematics**	knowledge of Python

Learning Objectives

This course is designed to introduce the students to real-world data analysis problems, their analysis and interpretation of results in the field of exploratory data science using Python.

Learning outcomes

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

- Apply descriptive statistics to obtain a deterministic view of data
- Apply basic and advanced level statistical function on data
- Perform data handling using Numpy arrays
- Do data cleaning and transformation before extracting useful information
- Visualize data for ease of understanding the revealed information

SYLLABUS OF GE-2a

UNIT – I & II (09 Hours)

Introduction to basic statistics and analysis: Fundamentals of Data Analysis, Statistical foundations for Data Analysis, Types of data, Descriptive Statistics, Python Libraries: NumPy, Pandas, Matplotlib

Array manipulation using NumPy: NumPy array: Creating NumPy arrays, various data types of NumPy arrays

UNIT – I & II (09 Hours)

Introduction to basic statistics and analysis: contd..

Correlation and covariance, Linear Regression, Statistical Hypothesis Generation and Testing

Unit 2 Array manipulation using Numpy: contd..

Indexing and slicing, swapping axes, transposing arrays, data processing using Numpy arrays

UNIT – III (15 Hours)

Data Manipulation using Pandas: Data Structures in Pandas: Series, Data Frame, Index objects, loading data into Panda's data frame, Working with Data Frames: Arithmetics, Statistics, Binning, Indexing, Reindexing, Filtering, Handling missing data, Hierarchical indexing, Data wrangling: Data cleaning, transforming, merging and reshaping

UNIT – IV (12 Hours)

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

Practical component (if any) – 30 Hours

Use data set 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 data frame with a selected dataset. Identify and count the missing values in a data frame. 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)
   a. Compute mean, mode, median, standard deviation, confidence interval and standard error for each feature
   b. Compute correlation coefficients between each pair of features and plot heatmap
   c. Find covariance between length of sepal and petal
   d. 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

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

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*, Sultan Chand & Sons, 2020.

Suggestive readings

(i) Chen D. Y, *Pandas for Everyone: Python Data Analysis*, Pearson, 2018.

(ii) Miller J.D. *Statistics for Data Science,* Packt Publishing, 2017.

GENERIC ELECTIVES (GE-2b): Data Analysis and Visualization using Spreadsheet

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		
GE2b **Data Analysis and Visualization using Spreadsheet**	**4**	**3**	**0**	**1**	**Class XII pass with Mathematics**	**Nil**

Learning Objectives

This course is designed to inculcate statistical thinking about data to the students who have studied Mathematics up to Class 10^{th} ONLY. It gently introduces the students to basic statistics, and builds gradually to cover advanced functionalities for data analysis in spreadsheets. The objective is to enhance the knowledge of statistics and enable students to make sense of data by analyzing and visualizing it using spreadsheets, interpreting the results and gaining insights.

Learning outcomes

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

- Analyze and visualize data using spreadsheets
- Apply basic and advanced level statistical functions in spreadsheets
- Gain practical, hands-on experience of data analysis using spreadsheet

SYLLABUS OF GE-2b

UNIT – I (09 Hours)

Introduction to Basic Statistics

Fundamentals of Data Analysis, Statistical foundations for Data Analysis, Types of data, Descriptive Statistics, Correlation and covariance, Linear Regression.

UNIT – II (15 Hours)

Data Handling

Spreadsheet concepts, managing worksheets, formatting cells, entering data, Handling operators in formula, Cell referencing and naming of cells and cell ranges, Sorting, Multilayer sorting, Data validation, Find and Replace, Paste special, Filter and advanced filter, Formatting as table, Pivot tables, Formulae vs functions, Cell formulae vs Array formulae.

Mathematical functions, Statistical functions, Logical functions, Date and Time functions, Lookup and reference: Hlookup, and Vlookup, Index and Match functions, Text functions.

What–if-analysis: Goal-seek, Data tables, Scenario manager.

UNIT – III (12 Hours)

Data Analysis

Explore a data model: its content, and its structure, using the Power Pivot add-in. Learning DAX formula language. Create calculated fields and calculated measure for each cell, filter context for calculation, and explore several advanced DAX functions.

Cube formulas to retrieve data from data model.

UNIT – IV (09 Hours)

Data Visualization

Different types of charts including Pivot charts: Column, Line, Pie, Bar, Scatter charts. Fine tuning of charts: Chart Elements, Chart Styles, Chart Filters, Box Plot.

Practical component (if any) – 30 Hours

1. In a meeting of a marketing department of an organization it has been decided that price of selling an item is fixed at Rs. 40. It was resolved to increases the selling of more items and getting the profit of Rs. 50000/-. Use Goal Seek of find out how many items you will have to sell to meet your profit figure.
2. Create worksheet related to crop production of various crops in Indian stats in last five years (wheat, rice, pulses, soya-bean, and cane-sugar etc).
    i) Make a bar chart
    ii) Make a pie chart
    iii) Make a box plot
3. Study and perform the various DAX functions to analyse the data.
4. Create workbook related to sales of Business Company having various product in last four quarters for 10 sales persons.
    i) Make a line graph to show the growth/decline in the sales
    ii) Show the graph of each sales person sales
    iii) Find the two sales persons done in last 2 quarters
    iv) Find the sales persons consistent in last four quarters
    v) Find the most popular product of the company and the current popular product of company.
5. Create a Pivot-table showing the Customer Names who placed orders with GSS during 2019-2022. For each customer, also show the total number of orders, Total Sales, and Total Profit. Add a Slicer or a Filter that can be used to show the information specifically for each Customer Segment. Use information from the Pivot- table to answer the following questions (Hint: Filter and sort the data in the Pivot- table to locate the answer):
    i) Which small business customer had the highest sales?
    ii) Which corporate customer placed the greatest number of orders in 2019-2022? How many orders were placed by the corporate customer?
    iii) Which consumer customer was the most profitable one?
    iv) What is the sales figure of the least profitable home office customer?
6. Consider the following worksheet: (enter 5 records)

FULL NAME	GRADE 1/2/3	BASIC SALARY	HRA	PF	GROSS	NET	VA	VA>HRA

HRA is calculated as follows:

Grade	HRA (% of basic)
1	40%
2	35%
3	30%

PF is 8% for all grades

VA is 15000, 10000, 7000 for Grades 1, 2 and 3.

Gross=Basic + HRA+VA

Net=Gross - PF

i) Find max, min and average salary of employees in respective Grade.
ii) Count no. of people where VA>HRA
iii) Find out most frequently occurring grade.
iv) Extract records where employee name starts with "A" has HRA>10000
v) Print Grade wise report of all employees with subtotals of net salary and also grand totals.
vi) Use subtotal command.
vii) Extract records where Grade is 1 or 2 and salary is between 10000 and 20000 both inclusive.

7. Create workbook related to sales of Business Company having various product in last ten quarters for 20 sales persons. Perform the following on workbook:
   i) Create and modify a Pivot-table
   ii) Apply Pivot-table styles and formatting
   iii) Filter a Pivot-table
   iv) Insert a slicer to filter a Pivot-table
   v) Create a Pivot Chart

8. Create a PivotTable showing Total Sales breakdown by Region, Product Category, and Product Sub-Category. Use information from the PivotTable to answer the following questions:
   i) What was the Total Sales figure included in this data set?
   ii) Which Product Category had the highest sales?

iii) Which Region had the lowest sales?
iv) What was the Total Sales of Appliances in Delhi?

9. You are required to prepare a payroll statement in the given format making maximum use of cell referencing facility:

Code	Name	Category	Is HRA to be Paid	Basic	DP	DA	HRA	TA	CCA	Gross
1			Y							
2			N							
	Total									

Required:

- Basic salary (Allow any Basic salary in the range of Rs.10000-35000)
- DP is 50% of Basic Salary.
- DA (as a Percentage of Basic + DP) is more than 35000 then 40% of basic else 30% of basic.
- HRA is to be paid @ 40% of (Basic plus DP) to those whom HRA payable is yes.
- TA is to be paid @ Rs. 800 PM if Basic Salary is Less than Rs.12000, otherwise the TA is Rs. 1000 PM)
- CCA is to be paid @ Rs. 300 PM if Basic Salary is less than Rs.12000/- otherwise the CCA is Rs. 500 PM)
- Gross salary is the sum of Salary and all other allowances
- Deduction: a) GPF 10% of (Basic +DP) subject to a minimum of Rs.2000/- b) IT 10% of Gross Salary
- Net salary is Gross salary minus total deductions.

10. Consider the following worksheet for APS 1st year students:

S.No.	Name	Physics	Chem	Bio	Maths	CS	Total	%	Grade

1									
2									
3									
4									
5									

The value of Grade is calculated as follows:

If % >=90	Grade A

If % >=80 & <90	Grade B
If % >=70 & <80	Grade C
If % >=60 & <70	Grade D

Otherwise, students will be declared fail.

i) Calculate Grade using if function
ii) Sort the data according to total marks
iii) Apply filter to display the marks of the students having more than 65% marks.
iv) Enter the S.No. of a student and find out the Grade of the student using VLOOKUP.
v) Extract all records where name
   a) Begins with “A”
   b) Contains “A”
   c) Ends with “A”

Essential/recommended readings

1. Gupta, S.P., *Elementary Statistical Methods*, Sultan Chand and Sons, New Delhi, 2017.

2. Goldmeier, J., *Advanced Excel Essentials*, Apress, 2014.
3. Slager, D., *Essential Excel 2016: A Step-by-Step Guide*, Apress, 2016.
4. Valerie M. Sue and Matthew T. Griffin, *Data Visualization and Presentation with Microsoft Office*, SAGE, 2016.
5. Schmuller, J., *Statistical Analysis with Excel for Dummies*, 4th edition., Wiley India Pvt Ltd., 2020.

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

ANNEXURE-4.01.48 (Revised)
AC-25.01.2023

INDEX
DEPARTMENT OF COMPUTER SCIENCE
Semester-III

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**	Pass in Class XII	DSC-01/DSC04

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.

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**	Pass in Class XII	DSC01/DSC04, DSC02

Learning Objectives

The course provides concepts that underlie all operating systems not tied to any particular operating system. The emphasis is done to explain 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 and structures/techniques used by operating systems to provide these services.

Learning outcomes

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

- Understand the need of an Operating System & Define Multiprogramming and Multithreading concepts.
- Implement Process Synchronization service (Critical Section, Semaphores), CPU scheduling service with various algorithms.
- Learn 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. Write a program to implement FCFS scheduling algorithm.

8. Write a program to implement SJF scheduling algorithm.

9. Write a program to implement non-preemptive priority based scheduling algorithm.

10. Write a program to implement SRTF scheduling algorithm.

11. Write a program to calculate sum of n numbers using Pthreads. A list of n numbers is divided into two smaller list of equal size, two separate threads are used to sum the sublists.

12. Write a program to implement first-fit, best-fit and worst-fit allocation strategies.

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**	Pass in Class XII	DSC01/DSC04

Learning Objectives

The course aims to provide students an 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 the 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, 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 Methods.

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.

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 Class XII	DSC01/ CBSE083/ CBSE065

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 interpret 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.

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 Class XII	NIL

Learning Objectives

This course introduces 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.

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-3a): 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			
GE3a 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 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.

SYLLABUS OF GE-3a

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**	**StudentName**	**CourseID**	**DOB**
	Char(6)	Varchar(20)	Varchar(10)	Date

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

ADMISSION	**Roll No**	**CID**	**DateOfAdmission**
	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 date of admission of 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'.

GENERIC ELECTIVES (GE-3b): 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			
GE3b: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 graphical user interface (GUI).

SYLLABUS OF GE-3b

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 concept.

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 application.
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 Frame container.

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

DISCIPLINE SPECIFIC CORE COURSE (DSC-3): 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		
DSC03: Computer System Architecture	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objectives

This course introduces the students to the fundamental concepts of digital computer organization, design and architecture. It aims to 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 the instruction cycle and describe interrupts and their handling.
- Explain how the CPU communicates with memory and I/O devices.
- Simulate the design of a basic computer using a software tool.

SYLLABUS OF DSC-3

Unit 1 (9 hours)

Digital Logic Circuits: Digital Logic Gates, Flip flops and their characteristic table, Logic circuit simplification using Boolean algebra and Karnaugh map, Don't care conditions, Combinational circuits, Introduction to Sequential circuits.

Unit 2 (7 hours)

Digital Components: Decoders, Encoders, Multiplexers, Binary Adder, Binary Adder Subtractor, Binary Incrementer, Registers and Memory units.

Unit 3 (13 hours)

Data Representation: Binary representation of both numeric and alphanumeric data, representation of numeric data in different number systems, (Binary, Octal, Decimal and Hexadecimal), conversion from one number system to another, complements, representation of signed and unsigned numbers, addition and subtraction of signed and unsigned numbers and overflow detection.

Unit 4 (9 hours)

Basic Computer Organization and Design: Stored program organization, Computer registers, Instruction set and their completeness, Instruction cycle, Memory reference instructions, Register reference instructions, Input- Output reference instructions, Interrupt cycle, Addressing modes.

Unit 5 (7 hours)

Input-Output Organization: I/O interface, I/O vs. Memory Bus, Isolated I/O, Memory Mapped I/O, Direct Memory Access.

Essential/recommended readings

1. M. Morris Mano, *Computer System Architecture*, 3rd edition, Pearson Education, 2017.
2. Linda Null, Julia Lobur, *Essentials of Computer Organization and Architecture*, 5th Edition, 2019.

Additional References

1. D. Comer, *Essentials of Computer Architecture,* 2nd edition, CRC Press, 2017.

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

Practical exercises such as

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

1. Create a machine based on the following architecture:

Registers

IR	DR	AC	AR	PC	I	E
16 bits	16 bits	16 bits	12 bits	12 bits	1 bit	1 bit

Memory 4096 words 16 bits per word	Instruction format 15 12 11 0

	Opcode	Address

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
			HLT	7001

Refer to Chapter-5 for description of instructions.

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

1. Implement fetch sequence
2. Write an assembly program to simulate addition of two numbers when one number is stored in memory and another is entered by the user.
3. Write an assembly program to simulate addition of two numbers when both numbers are taken as inputs from user.
4. Write an assembly program to simulate subtraction of two numbers when one number is stored in memory and another is entered by the user.
5. Write an assembly program to simulate subtraction of two numbers when both numbers are taken as inputs from user
6. Write an assembly program to simulate the following logical operations on two user-entered numbers.

i.AND

ii.OR

iii.NOT

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. CLE

ii. CLA

iii. CMA

iv. CME

Computer Science Courses for Undergraduate Programme of study with Computer Science discipline as one of the two Core Disciplines
(For e.g. courses for B.A. Programmes with Computer Science as Major discipline)

DISCIPLINE SPECIFIC CORE COURSE (DSC-3): 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		
DSC03: Computer System Architecture	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objectives

This course introduces the students to the fundamental concepts of digital computer organization, design and architecture. It aims to 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 the instruction cycle and describe interrupts and their handling.
- Explain how the CPU communicates with memory and I/O devices.
- Simulate the design of a basic computer using a software tool.

SYLLABUS OF DSC-3

Unit 1 (7 hours)

Introduction: Logic gates, Boolean algebra, combinational circuits, circuit simplification, flip-flops and sequential circuits, decoders, multiplexers, registers, memory units.

Unit 2 (9 hours)

Data Representation and Computer Arithmetic (of integers): Number systems, complements, fixed and floating-point representation, character representation, Addition, subtraction, Magnitude comparison.

Unit 3 (6 hours)

Central Processing Unit: Register organization, arithmetic and logical micro-operations, stack organization, micro programmed control

Unit 4 (9 hours)

Basic Computer Organization and Design: Computer registers, bus system, instruction set, timing and control, instruction cycle, memory reference, input-output and interrupt.

Unit 5 (8 hours)

Programming the Basic Computer: Instruction formats, addressing modes, instruction codes, machine language, assembly language.

Unit 6 (6 hours)

Input – Output Organization: Peripheral devices, I/O interface, asynchronous data transfer, priority interrupt.

Essential/recommended readings

1. M. Morris Mano, *Computer System Architecture*, 3rd edition, Pearson Education, 2017.
2. Linda Null, Julia Lobur, *Essentials of Computer Organization and Architecture*, 5th Edition, 2019.

Additional References

1. D. Comer, *Essentials of Computer Architecture,* 2nd edition, CRC Press, 2017.

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

Practical exercises such as

1. Write a program to convert unsigned numbers from binary to octal, hex to decimal.
2. Write a program to convert unsigned numbers from decimal to binary, octal, hex.
3. Write a program that will prompt for the input of two integer values. Then using the

   bitwise shift operators show the result of

   a) Left shifting the first number by the second.
   b) Right shifting the first number by the second.

4. Write a program that will prompt for the input of two integer values. Then using the bitwise shift operators show the result of
    a) Exclusive OR of the first number by the second bitwise.
    b) OR of the first number by the second bitwise.
    c) AND of the first number by the second bitwise.

5. Write a program that will prompt for the input of a binary value and display its 1's complement value.
6. Write a program that will prompt for the input of a binary value and display its 2's complement value.
7. Write a program that will prompt for the two inputs to implement binary addition operation.
8. Write a program that will prompt for the two input to implement binary subtraction using 2's complement.

-

DISCIPLINE SPECIFIC CORE COURSE (A3): 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		
A3: Data Mining - I	**4**	**3**	**0**	**1**	Pass in Class XII	**DSC01**

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:

- Pre-process the data for subsequent data mining tasks
- Apply a suitable classification algorithm to train the classifier and evaluate its performance.
- Apply appropriate clustering algorithm to cluster the data and evaluate clustering quality

- Use association rule mining algorithms and generate frequent item-sets and association rules

SYLLABUS OF A3

Unit 1 (8 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 (9 hours)

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

Unit 3 (11 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 (8 hours)

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

Unit 5 (9 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.

Recommended Datasets for :

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

Clustering: Grammatical Facial Expressions, HTRU2, Perfume data

Association Rule Mining: MovieLens, Titanics

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

Practical exercise such as

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.

Computer Science Courses for Undergraduate Programme of study with Computer Science discipline as one of the two Core Disciplines
(For e.g. courses for B.A. Programmes with Computer Science as Non-major iscipline)

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		
DSC03: Computer System Architecture	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objectives

This course introduces the students to the fundamental concepts of digital computer organization, design and architecture. It aims to 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 the instruction cycle and describe interrupts and their handling.
- Explain how the CPU communicates with memory and I/O devices.
- Simulate the design of a basic computer using a software tool.

SYLLABUS OF DSC-3

Unit 1 (7 hours)

Introduction: Logic gates, Boolean algebra, combinational circuits, circuit simplification, flip-flops and sequential circuits, decoders, multiplexers, registers, and memory units.

Unit 2 (9 hours)

Data Representation and Computer Arithmetic (of integers): Number systems, complements, fixed and floating-point representation, character representation, Addition, subtraction, Magnitude comparison.

Unit 3 (8 hours)

Central Processing Unit: Register organization, arithmetic and logical micro-operations, stack organization, microprogrammed control.

Unit 4 (7 hours)

Basic Computer Organization and Design: Computer registers, bus system, instruction set, timing and control, instruction cycle, memory reference, input-output and interrupt.

Unit 5 (7 hours)

Programming the Basic Computer: Instruction formats, addressing modes, instruction codes, machine language, assembly language.

Unit 6 (7 hours)

Input – Output Organization: Peripheral devices, I/O interface, asynchronous data transfer, priority interrupt.

Essential/recommended readings

1. M. Morris Mano, *Computer System Architecture*, 3rd edition, Pearson Education, 2017.
2. Linda Null, Julia Lobur, *Essentials of Computer Organization and Architecture*, 5th Edition, 2019.

Additional References

2. D. Comer, *Essentials of Computer Architecture,* 2nd edition, CRC Press, 2017.

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

Practical exercises such as

1. Write a program to convert unsigned number from binary to octal, hex to decimal.
2. Write a program to convert unsigned number from decimal to binary, octal, hex.
3. Write a program that will prompt for the input of two integer values. Then using the

   bitwise shift operators show the result of

   a) Left shifting the first number by the second.
   b) Right shifting the first number by the second.

4. Write a program that will prompt for the input of two integer values. Then using the

   bitwise shift operators show the result of
   a) Exclusive OR of the first number by the second bitwise.
   b) OR of the first number by the second bitwise.
   c) AND of the first number by the second bitwise.

5. Write a program that will prompt for the input of a binary value and display its 1's complement value.
6. Write a program that will prompt for the input of a binary value and display its 2's complement value.
7. Write a program that will prompt for the two inputs to implement binary addition operation.
8. Write a program that will prompt for the two input to implement binary subtraction using 2's complement.

Annexure-55
Item No. 4-1-7
EC-09.06.2023

INDEX

DEPARTMENT OF COMPUTER SCIENCE

SEMESTER-IV

INDEX

DEPARTMENT OF COMPUTER SCIENCE

SEMESTER-V

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**		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		
	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

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

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

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

DISCIPLINE SPECIFIC CORE COURSE (DSC04): 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 (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC04: Operating Systems	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 01 Programming using Python/ A course in C/C++/Python at plus 2 level.

Learning Objectives

This course introduces the students to Operating Systems 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.
- learn about shell scripts and would be able to use the system in an efficient manner.

SYLLABUS OF DSC04

Unit 1 (4 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 (10 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 (9 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 (9 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 (5 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
3. Das, S., *Unix: Concepts and Applications*, 4th edition, TMH, 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 (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
   iii. Directory Management : cd, mkdir, rmdir, ls

2. Execute various LINUX commands for:
   i. Process Control: fork, getpid, ps
   ii. Communication: Input-output redirection, Pipe
   iii. Protection Management: chmod, chown, chgrp
3. Write a program(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 calculate sum of n numbers using Pthreads.
5. Write a program to generate a Fibonacci Series of numbers using Pthreads.
6. Write a program to implement best-fit and worst-fit allocation strategies
7. Write a program to copy files using system calls and using pthreads and compare timings.
8. Write a program to implement FCFS scheduling algorithm.
9. Write a program to implement SJF scheduling algorithm.
10. Write a program to implement non-preemptive priority based scheduling algorithm.

Computer Science Courses for Undergraduate Programme of study with Computer Science discipline as one of the two Core Disciplines
(For e.g. courses for B.A. Programmes with Computer Science as Major discipline)

Computer Science Major

DISCIPLINE SPECIFIC CORE COURSE (DSC04): 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 (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC04: Operating Systems	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 01 Programming using Python/ A course in C/C++/Python at plus 2 level

Learning Objectives

This course introduces the students to Operating Systems 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.
- learn about shell scripts and would be able to use the system in an efficient manner.

SYLLABUS OF DSC04

Unit 1 (4 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 (10 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 (9 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 (9 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 (5 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
3. Das, S., *Unix: Concepts and Applications*, 4th edition, TMH, 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 (If any): (30 Hours)

Practical exercises such as

1. Execute various LINUX commands for:
   i. Information Maintenance: wc, clear, cal, who, date, pwd
   iv. File Management: cat, cp, rm, mv, cmp, comm, diff, find, grep
   v. Directory Management : cd, mkdir, rmdir, ls
2. Execute various LINUX commands for:
   iv. Process Control: fork, getpid, ps
   v. Communication: Input-output redirection, Pipe
   vi. Protection Management: chmod, chown, chgrp
3. Write a program(using fork() and/or exec() commands) where parent and child execute:
   iv. same program, same code.
   v. same program, different code.
   vi. before terminating, the parent waits for the child to finish its task.
4. Write a program to calculate sum of n numbers using Pthreads.
5. Write a program to generate a Fibonacci Series of numbers using Pthreads.
6. Write a program to implement best-fit and worst-fit allocation strategies
7. Write a program to copy files using system calls and using pthreads and compare timings.
8. Write a program to implement FCFS scheduling algorithm.
9. Write a program to implement SJF scheduling algorithm.
10. Write a program to implement non-preemptive priority based scheduling algorithm.

DISCIPLINE SPECIFIC CORE COURSE (A4): 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 (if any)
		Lecture	Tutorial	Practical/ Practice		
A4: 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 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 modeling

SYLLABUS OF A4

Unit 1 (9 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 (8 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 Practicals List (If any): (30 Hours)

Practical exercise such as

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 should be promoted to take up one project covering at least one unit of the syllabus on any UCI/kaggle/data.gov.in or a dataset verified by the teacher. 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.

Computer Science Courses for Undergraduate Programme of study with Computer Science discipline as one of the two Core Disciplines
(For e.g. courses for B.A. Programmes with Computer Science as Non-major discipline)

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: Operating Systems	**4**	**3**	**0**	**1**	Pass in Class XII	DSC 01 Programming using Python/ A course in C/C++/Python at plus 2 level

Learning Objectives

This course introduces the students to Operating Systems 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.
- learn about shell scripts and would be able to use the system in an efficient manner.

SYLLABUS OF DSC04

Unit 1 (4 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 (10 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 (9 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 (9 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 (5 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
3. Das, S., *Unix: Concepts and Applications*, 4th edition, TMH, 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 (If any): (30 Hours)

Practical exercises such as

1. Execute various LINUX commands for:
   i. Information Maintenance: wc, clear, cal, who, date, pwd
   vi. File Management: cat, cp, rm, mv, cmp, comm, diff, find, grep
   vii. Directory Management : cd, mkdir, rmdir, ls

2. Execute various LINUX commands for:

vii. Process Control: fork, getpid, ps

viii. Communication: Input-output redirection, Pipe

ix. Protection Management: chmod, chown, chgrp

3. Write a program(using fork() and/or exec() commands) where parent and child execute:

vii. same program, same code.

viii. same program, different code.

ix. before terminating, the parent waits for the child to finish its task.

4. Write a program to calculate sum of n numbers using Pthreads.

5. Write a program to generate a Fibonacci Series of numbers using Pthreads.

6. Write a program to implement best-fit and worst-fit allocation strategies

7. Write a program to copy files using system calls and using pthreads and compare timings.

8. Write a program to implement FCFS scheduling algorithm.

9. Write a program to implement SJF scheduling algorithm.

10. Write a program to implement non-preemptive priority based scheduling algorithm.

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.

SYLLABUS OF DSC-5

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 (6 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 (6 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 queries using them.

Unit 4 (8 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 (10 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.

Unit 6 (10 hours)

File indexing and Transaction Processing: Need of file indexes, types of indexes, file organizations, single- and multi-level indexing, concurrent execution of transactions, ACID properties, need of data recovery.

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 : (30 Hours)

Practical exercises such as

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**	**StudentName**	**CourseID**	**DOB**
	Char(6)	Varchar(20)	Varchar(10)	Date

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

ADMISSION	**Roll No**	**CID**	**DateOfAdmission**
	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.

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 date of admission of 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'.

Students are also encouraged to implement the database given in the textbook and do the related queries.

Computer Science Courses for Undergraduate Programme of study with Computer Science discipline as one of the two Core Disciplines
(For e.g. courses for B.A. Programmes with Computer Science as Major discipline)

Computer Science Major

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

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		
DSC05: Database Management System	**4**	**3**	**0**	**1**		NIL

Learning Objectives

The course will give an overview of categories of data models used by database management systems and writing queries in relational algebra. The importance of file indexing and controlled execution of transactions will be taught. The course would give students hands-on practice to write complex queries, nested queries using aggregate functions in SQL and to use basic database administration commands.

Learning outcomes

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

- Write queries using relational algebra, a procedural language.
- Use the concept of functional dependencies to remove data anomalies and arrive at normalized database design.
- Understand the data storage system, file organization and need of file indexing.
- Learn the importance of transaction processing and concurrency control.
- Write complex and nested SQL queries, and learn basic database administration commands.
- Acquire information about emerging technologies.

SYLLABUS OF DSC 05

Unit 1 (6 hours)

Data models and Relational Algebra: Categories of data models, types of database users, Relational Algebra Operations from SET Theory, SELECT, PROJECT, JOIN, DIVISION Operations.

Unit 2 (8 hours)

Normalization: Functional dependencies, minimal cover, normalizing database with multiple keys till 3NF, Boyce-Codd Normal Form

Unit 3 (8 hours)

Database Storage and index Structures: Storage of data, file structure, file organization and its types: Sequential, Heap and Indexed, Hash file, need for file Indexing, single- and multi-level indexing.

Unit 4 (8 hours)

Transaction Processing: Concurrent execution of transactions and their handling, ACID properties, need of data recovery and log files.

Unit 5 (8 hours)

Advanced SQL: Nested and complex queries using Inner JOIN, Left JOIN, Right JOIN, Full JOIN, views, Database Administration Commands: COMMIT, ROLLBACK, drop database, control permissions etc.

Unit 6 (7 hours)

Emerging Technology: Distributed Database Concepts, Introduction to emerging technologies like Data Warehousing and OLAP, Data Mining.

Essential/recommended readings

1. Elmasri R. and 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. Silberschatz, A., Korth, H.F., & Sudarshan, S. Database System Concepts, 8th Edition, McGraw Hill, 2019.

2. Ramakrishnan, R. & Gehrke, J. Database Management Systems, 3rd Edition, Tata McGraw Hill Education, 2014.

Suggested Practical List : (30 Hours)

MySQL might be used for practical purposes.

The following concepts must be introduced to the students:

1. Multiple table queries (join on different and same tables)
2. Nested select statements
3. Set manipulation using (any, in, contains, all, not in, not contains, exists, not exists union, intersect, minus, etc.)
4. Categorization using group by... having
5. Arranging using order by

List of sample queries to be done using the above schema.

Create tables with relevant column and integrity constraints and populate the tables with data.

Perform the following queries on the database:

1. Display all the details of all employees working in the company.
2. Display ssn, lname, fname, address of employees who work in department no 7.
3. Retrieve the birthdate and address of the employee whose name is 'Franklin T. Wong' and who is a manager
4. Retrieve the name and salary of every employee working in department ‘Production’
5. Display maximum and minimum salaries of each department
6. Retrieve all employee names whose address is in ‘Bellaire’ and working on at least one project
7. Modify the datatype of hours column to float in works_on table.
8. Retrieve all employees who age is more than 35 years
9. Retrieve all employees in department 5 whose salary is between 50,000 and 60,000(inclusive)
10. Add another column named “Join_Date” in the employee table and “Address” in the department table
11. Retrieve the names of all employees who do not have supervisors
12. Retrieve SSN and department name for all employees
13. Retrieve the name and address of all employees who work for the 'Research' department
14. For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and birthdate

15. For each employee, retrieve the employee's name, and the name of his or her immediate supervisor
16. Retrieve all combinations of Employee Name and Department Name
17. Make a list of all project numbers for projects that involve an employee whose last name is 'Narayan' either as a worker or as a manager of the department that controls the project
18. Increase the salary of all employees working on the 'ProductX' project by 15%. Retrieve employee name and increased salary of these employees
19. Retrieve a list of employees and the project name each works in, ordered by the employee's department, and within each department ordered alphabetically by employee first name
20. Select the names of employees whose salary does not match with salary of any employee in department 10
21. Drop the column "Join_Date" from the employee table
22. Retrieve the name of each employee who has a dependent with the same first name and same sex as the employee
23. Retrieve the employee numbers of all employees who work on project located in Bellaire, Houston, or Stafford.
24. Change the name of existing table DEPT_LOCATIONS to DLOCATIONS
25. Find the sum of the salaries of all employees, the maximum salary, the minimum salary and the average salary for each department. Display with proper headings
26. Find the sum of the salaries and number of employees of the 'Marketing' department, as well as the maximum salary, the minimum salary, and the average salary in this department
27. Select the names of employees whose salary is greater than the average salary of all employees in department 10
28. For each department, retrieve the department number, the number of employees in the department and their average salary
29. For each project, retrieve the project number, the project name, and the number of employees who work on that project for more than 10 hours
30. Change the location and controlling department number for all projects having more than 5 employees to 'Bellaire' and 6 respectively
31. For each department having more than 10 employees, retrieve the department name, number of employees drawing more than 40,000 as salary
32. Display employee names having no dependent in descending order along with their age
33. For each department, find the number of female and number of male employees along with name of manager of that department
34. Find the name and age of youngest employee in each department
35. Change the name of MINIT attribute of an employee table to MNAME using alter command
36. Create a view to keep names of employees and their department names if they are not working on any project
37. Create a view to keep track of all employee names who are working on same projects as that of employee 'Franklin T. Wong'
38. Execute system administrative commands like commit, rollback, granting control permissions, etc.

DISCIPLINE SPECIFIC CORE COURSE (A5): Machine Learning

0CREDIT 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		
A5: Machine Learning	**4**	**3**	**0**	**1**	Pass in Class XII	A1 Programming Fundamentals using Python, A2 Data Visualization using Python

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.
- Appreciate the need of preprocessing, feature scaling and feature selection.
- Understand the fundamentals of classification, regression and clustering
- Implement various machine learning algorithms learnt in the course.

SYLLABUS OF A5

Unit 1 (8 hours)

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

Unit 2 (10 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 (15 hours)

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

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 Practicals List : (30 Hours)

Practical exercise such as

Use Python for practical labs for Machine Learning. Utilize publically 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:

i. Scale/Normalize the data
ii. Reduce dimension of the data with different feature selection techniques

iv. Perform k-cross-validation on datasets for evaluation

Report the efficacy of the machine learning models as follows:

i. MSE and R^2 score for regression models
ii. 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

Computer Science Courses for Undergraduate Programme of study with Computer Science discipline as one of the two Core Disciplines
(For e.g. courses for B.A. Programmes with Computer Science as Non-major discipline)

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		
DSC05: Database Management System	**4**	**3**	**0**	**1**	Pass in Class XII	

Learning Objectives

The course will give an overview of categories of data models used by database management systems and writing queries in relational algebra. The importance of file indexing and controlled execution of transactions will be taught. The course would give students hands-on practice to write complex queries, nested queries using aggregate functions in SQL and to use basic database administration commands.

Learning outcomes

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

- Write queries using relational algebra, a procedural language.
- Use the concept of functional dependencies to remove data anomalies and arrive at normalized database design.
- Understand the data storage system, file organization and need of file indexing.
- Learn the importance of transaction processing and concurrency control.
- Write complex and nested SQL queries, and learn basic database administration commands.
- Acquire information about emerging technologies.

SYLLABUS OF DSC 05

Unit 1 (6 hours)

Data models and Relational Algebra: Categories of data models, types of database users, Relational Algebra Operations from SET Theory, SELECT, PROJECT, JOIN, DIVISION Operations.

Unit 2 (8 hours)

Normalization: Functional dependencies, minimal cover, normalizing database with multiple keys till 3NF, Boyce-Codd Normal Form

Unit 3 (8 hours)

Database Storage and index Structures: Storage of data, file structure, file organization and its types: Sequential, Heap and Indexed, Hash file, need for file Indexing, single- and multi-level indexing.

Unit 4 (8 hours)

Transaction Processing: Concurrent execution of transactions and their handling, ACID properties, need of data recovery and log files.

Unit 5 (8 hours)

Advanced SQL: Nested and complex queries using Inner JOIN, Left JOIN, Right JOIN, Full JOIN, views, Database Administration Commands: COMMIT, ROLLBACK, drop database, control permissions etc.

Unit 6 (7 hours)

Emerging Technology: Distributed Database Concepts, Introduction to emerging technologies like Data Warehousing and OLAP, Data Mining.

Essential/recommended readings

1. Elmasri R. and 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. Silberschatz, A., Korth, H.F., & Sudarshan, S. Database System Concepts, 8th Edition, McGraw Hill, 2019.

Suggested Practical List : (30 Hours)

MySQL might be used for practical purposes.

The following concepts must be introduced to the students:

1. Multiple table queries (join on different and same tables)
2. Nested select statements
3. Set manipulation using (any, in, contains, all, not in, not contains, exists, not exists union, intersect, minus, etc.)
4. Categorization using group by.....having
5. Arranging using order by

List of sample queries to be done using the above schema.

Create tables with relevant column and integrity constraints and populate the tables with data.

Perform the following queries on the database:

1. Display all the details of all employees working in the company.
2. Display ssn, lname, fname, address of employees who work in department no 7.
3. Retrieve the birthdate and address of the employee whose name is 'Franklin T. Wong' and who is a manager
4. Retrieve the name and salary of every employee working in department ‘Production’
5. Display maximum and minimum salaries of each department
6. Retrieve all employee names whose address is in ‘Bellaire’ and working on at least one project
7. Modify the datatype of hours column to float in works_on table.
8. Retrieve all employees who age is more than 35 years
9. Retrieve all employees in department 5 whose salary is between 50,000 and 60,000(inclusive)
10. Add another column named “Join_Date” in the employee table and “Address” in the department table
11. Retrieve the names of all employees who do not have supervisors
12. Retrieve SSN and department name for all employees
13. Retrieve the name and address of all employees who work for the 'Research' department
14. For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and birthdate
15. For each employee, retrieve the employee's name, and the name of his or her immediate supervisor
16. Retrieve all combinations of Employee Name and Department Name
17. Make a list of all project numbers for projects that involve an employee whose last name is 'Narayan’ either as a worker or as a manager of the department that controls the project
18. Increase the salary of all employees working on the 'ProductX' project by 15%. Retrieve employee name and increased salary of these employees
19. Retrieve a list of employees and the project name each works in, ordered by the employee's department, and within each department ordered alphabetically by employee first name

20. Select the names of employees whose salary does not match with salary of any employee in department 10
21. Drop the column “Join_Date” from the employee table
22. Retrieve the name of each employee who has a dependent with the same first name and same sex as the employee
23. Retrieve the employee numbers of all employees who work on project located in Bellaire, Houston, or Stafford.
24. Change the name of existing table DEPT_LOCATIONS to DLOCATIONS
25. Find the sum of the salaries of all employees, the maximum salary, the minimum salary and the average salary for each department. Display with proper headings
26. Find the sum of the salaries and number of employees of the ‘Marketing’ department, as well as the maximum salary, the minimum salary, and the average salary in this department
27. Select the names of employees whose salary is greater than the average salary of all employees in department 10
28. For each department, retrieve the department number, the number of employees in the department and their average salary
29. For each project, retrieve the project number, the project name, and the number of employees who work on that project for more than 10 hours
30. Change the location and controlling department number for all projects having more than 5 employees to ‘Bellaire’ and 6 respectively
31. For each department having more than 10 employees, retrieve the department name, number of employees drawing more than 40,000 as salary
32. Display employee names having no dependent in descending order along with their age
33. For each department, find the number of female and number of male employees along with name of manager of that department
34. Find the name and age of youngest employee in each department
35. Change the name of MINIT attribute of an employee table to MNAME using alter command
36. Create a view to keep names of employees and their department names if they are not working on any project
37. Create a view to keep track of all employee names who are working on same projects as that of employee ‘Franklin T. Wong’
38. Execute system administrative commands like commit, rollback, granting control permissions, etc.