

**INDEX**  
**DEPARTMENT OF COMPUTER SCIENCE**  
**Semester-III**

<b>S.No.</b>	<b>Contents</b>	<b>Page No.</b>
<b>1</b>	<b>BSc. (Hons.) Computer Science- DSCs</b>  1. Data Structures 2. Operating Systems 3. Numerical Optimization	<b>2-9</b>
<b>2</b>	<b>Pool of Discipline Specific Electives (DSEs) for BSc. (Hons.) Computer Science</b>  1. Data Analysis and Visualization (DAV) 2. Microprocessors	<b>10-14</b>
<b>3</b>	<b>POOL OF GENERIC ELECTIVES</b>  1. Database Management Systems 2. Java Programming	<b>15-20</b>
<b>4</b>	<b>BSc. (Prog.) with Computer Sciences</b> <b>DISCIPLINE SPECIFIC CORE (DSC)</b> 1. Computer System Architecture	<b>21-24</b>
<b>5</b>	<b>B.A. (Prog.) with Computer Science as Major</b>  1. Computer System Architecture 2. Data Mining-I	<b>25-29</b>
<b>6</b>	<b>B.A. (Prog.) with Computer Science as Non-Major</b>  1. Computer System Architecture	<b>30-32</b>

## 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++*, 2<sup>nd</sup> edition, Wiley, 2011.
2. Cormen, T.H., Leiserson, C.E., Rivest, R. L., Stein C. *Introduction to Algorithms*, 4<sup>th</sup> edition, Prentice Hall of India, 2022.

### **Additional references**

1. Sahni, S. *Data Structures, Algorithms and applications in C++*, 2<sup>nd</sup> 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		
<b>DSC 08 Operating Systems</b>	<b>4</b>	<b>3</b>	<b>0</b>	<b>1</b>	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*, 9<sup>th</sup> edition, John Wiley Publications, 2016.
2. Tanenbaum, A. S. *Modern Operating Systems*, 3<sup>rd</sup> edition, Pearson Education, 2007.
3. Stallings, W. *Operating Systems: Internals and Design Principles*, 9<sup>th</sup> edition, Pearson Education, 2018.

## Additional References

1. Dhamdhare, D. M., *Operating Systems: A Concept-based Approach*, 2<sup>nd</sup> 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 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		
<b>DSC09 Numerical Optimization</b>	<b>4</b>	<b>3</b>	<b>0</b>	<b>1</b>	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. Siau, 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		
<b>Data Analysis and Visualization (DAV)</b>	<b>4</b>	<b>3</b>	<b>0</b>	<b>1</b>	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 matplotlib 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*, 2<sup>nd</sup> 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*, 12<sup>th</sup> 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/](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		
<b>Microprocessors</b>	<b>4</b>	<b>3</b>	<b>0</b>	<b>1</b>	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*, 8<sup>th</sup> edition, Pearson education, 2009.

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

### **Additional References**

1. Ramesh S Gaonkar *Microprocessor architecture, programming, and applications with the 8085*, 6<sup>th</sup> 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			
<b>GE3a Database Management Systems</b>	<b>4</b>	<b>3</b>	<b>0</b>	<b>1</b>	Pass in Class XII	NIL	<b>Computer Science</b>

#### 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*, 7<sup>th</sup> Edition, Pearson Education, 2016.
2. Murach J., *Murach's MySQL*, 3<sup>rd</sup> Edition, Pearson, 2019.

### Additional References

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

<b>STUDENT</b>	<b><u>Roll No</u></b>	<b>StudentName</b>	<b>CourseID</b>	<b>DOB</b>
	Char(6)	Varchar(20)	Varchar(10)	Date

<b>COURSE</b>	<b><u>CID</u></b>	<b>CourseName</b>	<b>Course Type</b>	<b>Teacher-in-charge</b>	<b>TotalSeats</b>	<b>Duration</b>
	Char(6)	Varchar(20)	Char(8)	Varchar(15)	Unsigned int	Unsigned int

<b>ADMISSION</b>	<b><u>Roll No</u></b>	<b><u>CID</u></b>	<b>DateOfAdmission</b>
	Char(6)	Char(6)	Date

Here Rollno (ADMISSION) and CID (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			
<b>GE3b:Java Programming</b>	<b>4</b>	<b>3</b>	<b>0</b>	<b>1</b>	Pass in Class XII	NIL	<b>Computer Science</b>

#### 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*, 10<sup>th</sup> edition, Pearson, 2017.
2. James Gosling, Bill Joy, Guy L. Steele Jr, Gilad Bracha, Alex Buckley, *The Java Language Specification, Java SE 7<sup>th</sup> edition*, Addison-Wesley, 2011

### **Additional References**

1. Herbert Schildt, *Java: The Complete Reference*, 10<sup>th</sup> 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*, 3<sup>rd</sup> 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		
<b>DSC03: Computer System Architecture</b>	<b>4</b>	<b>3</b>	<b>0</b>	<b>1</b>	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.



	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		
<b>DSC03: Computer System Architecture</b>	<b>4</b>	<b>3</b>	<b>0</b>	<b>1</b>	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*, 3<sup>rd</sup> edition, Pearson Education, 2017.
2. Linda Null, Julia Lobur, *Essentials of Computer Organization and Architecture*, 5<sup>th</sup> Edition, 2019.

### **Additional References**

1. D. Comer, *Essentials of Computer Architecture*, 2<sup>nd</sup> 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		
<b>A3: Data Mining - I</b>	<b>4</b>	<b>3</b>	<b>0</b>	<b>1</b>	Pass in Class XII	<b>DSC01</b>

#### 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*, 2<sup>nd</sup> edition, Pearson, 2021.
2. Han J., Kamber M. and Pei J. *Data Mining: Concepts and Techniques*, 3<sup>rd</sup> edition, 2011, Morgan Kaufmann Publishers.
3. Zaki M. J. and Meira J. Jr. *Data Mining and Machine Learning: Fundamental Concepts and Algorithms*, 2<sup>nd</sup> 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*, 1<sup>st</sup> 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, Titanic

### 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 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		
<b>DSC03: Computer System Architecture</b>	<b>4</b>	<b>3</b>	<b>0</b>	<b>1</b>	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*, 3<sup>rd</sup> edition, Pearson Education, 2017.
2. Linda Null, Julia Lobur, *Essentials of Computer Organization and Architecture*, 5<sup>th</sup> Edition, 2019.

### Additional References

2. D. Comer, *Essentials of Computer Architecture*, 2<sup>nd</sup> 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.