Punyashlok Ahilyadevi Holkar Solapur University, Solapur

Name of the Faculty: Science and Technology

NEP-2020

Syllabus: - Computer Science

Name of the Programme: B.Sc.(CS) II (Sem.– III & IV) (NEP-2020)

(Syllabus to be implemented w.e.f. June 2025)

Punyashlok Ahilyadevi Holkar Solapur University, Solapur

B. Sc. (Computer Science)- II

Syllabus (Semester – III and IV)

(NEP-2020)

With Effect from June 2025

Punyashlok Ahilyadevi Holkar Solapur University, Solapur
Faculty of Science and Technology
Structure as per NEP-2020 LEVEL – 5.0
B.Sc. (Computer Science)-II

Level	Sem.	Major		Minor		GE/OE	SEC/ VSC	AES, IKS, VEC	CC	Total Credits	Cumulative Credits
		T	P	T	P						
5	III	DSC 1-3 (2)	DSC 1-3 (1)	DSC 2-3 (2)	DSC 2-3 (1)	GE3/ OE3 (2)	VSC1- (2) VSC2- (2)	L2-1 (2)	CC2(2)	22	44 UG Diploma (88)
		DSC 1-4 (2)	DSC 1-4 (1)	DSC 2-4 (2)	DSC 2-4 (1)						
	IV	DSC 1-5 (2)	DSC 1-5 (1)	DSC 2-5 (2)	DSC 2-5 (1)	GE4/ OE4 (2)	VSC3- (2) VSC4- (2)	L2-2 (2)	FP1/ CEP1(2)	22	
		DSC 1-6 (2)	DSC 1-6 (1)	DSC 2-6 (2)	DSC 2-6 (1)						

Subject/ Core Course	Name and Type of the Paper		Hrs./week			Total Marks/ Paper	UA	CA	Credits
	Type	Name	L	T	P				
Sem-III									
Major	DSC1-3	Introduction to Data Structures and Applications	2	-	-	50	30	20	2
	Practical	Practical based on DSC1-3	-	-	2	25	15	10	1
	DSC1-4	Fundamentals of Database Systems with SQL and PL/SQL	2	-	-	50	30	20	2
	Practical	Practical based on DSC1-4	-	-	2	25	15	10	1
Minor	DSC2-3	Data Visualization	2	-	-	50	30	20	2
	Practical	Practical based on DSC2-3	-	-	2	25	15	10	1
	DSC2-4	PHP Programming	2	-	-	50	30	20	2
	Practical	Practical based on DSC2-4	-	-	2	25	15	10	1
Generic/ Open Elective Courses	GE3/ OE3	Software Engineering	2	-	-	50	30	20	2
SEC/ VSC	VSC-1	Based on DSC1-3 and DSC1-4	-	-	4	50	30	20	2
	VSC-2	Based on DSC2-3 and DSC2-4	-	-	4	50	30	20	2
AEC	L2-1	English	2	-	-	50	30	20	2
Field Project/ RP/ CC/ Internship/ Apprenticeship / Community Engagement & Services	CC2	Select any one from bucket of Community Engagement & Services	2	-	-	50	30	20	2
Total			**14**		**16**	**550**	**330**	**220**	**22**

Sem-IV									
Major	DSC1-5	Foundations of Core Java	2	-	-	50	30	20	2
	Practical	Practical based on DSC1-5	-	-	2	25	15	10	1
	DSC1-6	Python Programming	2	-	-	50	30	20	2
	Practical	Practical based on DSC1-6	-	-	2	25	15	10	1
Minor	DSC2-5	Linux Essentials and Shell Programming	2	-	-	50	30	20	2
	Practical	Practical based on DSC2-5	-	-	2	25	15	10	1
	DSC2-6	Software Testing	2	-	-	50	30	20	2
	Practical	Practical based on DSC2-6	-	-	2	25	15	10	1
Generic/ Open Elective Courses	GE4/ OE4	Fundamentals Data Communication and Network Systems	2	-	-	50	30	20	2
SEC/ VSC	VSC-1	Based on DSC1-5 and DSC1-6	-	-	4	50	30	20	2
	VSC-2	Based on DSC2-5 and DSC2-6	-	-	4	50	30	20	2
AES	L2-2	English	2	-	-	50	30	20	2
Field Project/ RP/ CC/ Internship / Apprenticeship / Community Engagement & Services	FP1/CEP1(2)	Select any one from bucket of Community Engagement & Services	-	-	4	50	30	20	2
Total			**12**	**-**	**20**	**550**	**330**	**220**	**22**
Grand Total			**26**	**-**	**36**	**1100**	**660**	**440**	**44**

Abbreviations:

L: Lectures	**T**: Tutorials	**P**: Practical	**UA** : University Assessment	**CA** : College Assessment

Generic/ Open Electives: **GE/OE**	Skill Enhancement Courses: **SEC**
Indian Knowledge System: **IKS**	Ability Enhancement Courses: **AES**
Value Education Courses: **VEC**	Vocational Skill and Skill Enhancement Courses: **VSEC**
Co-curricular Courses: **CC**	

Student contact hours per week: 24 Hours (Min.)

Total Credits for B.Sc. (CS)-II (Semester III and IV): 44

Medium of instruction: English
• Practical Examination is Semester wise before theory Examination. • Duration of Practical Examination as per respective BOS guidelines • Separate passing is mandatory for Theory, Internal and Practical Examination
Exit Option: Award of UG Diploma in Major with 88 credits and an additional 4 credits core NSQF course/ Internship OR Continue with Major

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	III					
Paper Category:	DSC1-3 **(Major)**					
Paper Name:	Introduction to Data Structures and Applications					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives –

The aim of this course is to prepare learners for a foundational understanding of computers, encompassing:

1. To provide foundational knowledge of data structures and their importance in problem-solving.
2. To introduce various types of data structures, their representation, and associated algorithms.
3. To explore the applications of stacks, queues, linked lists, trees, and graphs in computational problems.
4. To develop proficiency in implementing sorting and searching algorithms for effective data manipulation.
5. To enable students to design and analyze efficient algorithms for real-world applications.

Unit I- Introduction of Data Structure [15]

Introduction of Data Structure, Need of Data Structure, Types of Data Structure, ADT,

Stack: Introduction to stack, Representation-static & dynamic, stack Operations, Application -infix to postfix & prefix, postfix evaluation, recursion, expression validity.

Queues: Introduction to Queue, Representation -static & dynamic, Operations, Circular queue, Double ended queue, priority queues, Applications of Queue.

Linked List: Introduction to List, Implementation of List – static & dynamic representation, Types of Linked List, Operations on List, Applications of Linked List – polynomial manipulation

Unit II- Nonlinear Data Structure, Searching and Sorting [15]

Trees: Concept & Terminologies, Binary tree, binary search tree, Operations on BST – create, Insert, delete, traversals (preorder, inorder, postorder)

Graph- Graph terminology, Representation of graphs, Graph Traversal–BFS (breadth first search), DFS (depth first search), Minimum spanning Tree

Sorting: Bubble sort, Quick sort, Simple Insertion sort, Shell sort, Address calculation sort, Selection Sort, Heap Sort, Merge sort, Radix Sort.

Searching: Linear Search, Binary Search, and Tree searching methods, Multiway search tree, Hash function (open and close).

Course Outcomes-

Students will able to:

1. Understand the need for data structures, differentiate between static and dynamic representations, and identify appropriate data structures for specific applications.
2. Perform operations on stacks and queues, including applications like expression evaluation, recursion, and priority queue management.
3. Implement and operate on different types of linked lists, including their application in polynomial manipulation and dynamic memory allocation.
4. Construct and traverse binary trees, binary search trees and perform operations such as insertion, deletion, and traversal.
5. Implement and analyze graph algorithms for traversal and minimum spanning trees.
6. Utilize various Sorting and Searching Techniques like Quick Sort, Merge Sort, and Heap Sort, as well as searching methods like Linear Search, Binary Search, and Hashing.

Reference Books-

1. Data Structures and Algorithmsǁ, Pearson Education, Reprint 2006 by Alfred V. Aho, John E. Hopcroft and Jeffrey D. Ullman
2. Algorithms, data structures, Programs by Nikaulus Wirth:
3. File Systems, Structures and Algorithms (PHI). By Thoms Horbron:
4. Art of computer Programming Vol – I. by D. E. Kunth:
5. Data structures using C and C++ (PHI). By Tanenbaum:
6. Fundamentals of computer algorithms by 2nd edition galgotia publication by Ellis horowitz, sartaj sahni

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	III					
Paper Category:	Practical **(Major)**					
Paper Name:	Practical based on DSC1-3					
Credit:	01		**Practical:**	2 Hrs./Week		
Marks:	**UA:**	15	**CA:**	10	**Total:**	25

Course Objectives –

The objective of this course is to equip students with the practical skills needed to implement and utilize various data structures and algorithms efficiently. By focusing on both fundamental and advanced topics, the course aims to enable students to:

1. Understand and apply the core concepts of data structures, such as stacks, queues, linked lists, trees, and graphs.
2. Develop and implement algorithms for manipulating and traversing these data structures.
3. Analyze and implement sorting and searching algorithms to solve real-world problems.
4. Gain hands-on experience with both static and dynamic representations of data structures.
5. Apply the knowledge of data structures and algorithms to practical applications such as expression evaluation, graph traversal, and tree manipulations.

Practical based on DSC1-3:

1. Implement stack operations (push, pop, peek) using static representation and dynamic representation.
2. Convert infix expression to postfix expression and evaluate a postfix expression.
3. Check the validity of a given mathematical expression using a stack and implement recursion using stack.
4. Implement basic queue operations (enqueue, dequeue) using static representation and dynamic representation.
5. Implement operations of a circular queue, a double-ended queue (deque) and priority queue operations.

6. Implement a singly, doubly, circular and circular doubly linked list with all operations (create, insert, delete, display).
7. Perform polynomial addition and multiplication using a singly linked list.
8. Create and traverse a binary tree using preorder, inorder, and postorder traversals.
9. Implement basic operations on a binary search tree (insert, delete, search).
10. Represent a graph using an adjacency matrix and adjacency list.
11. Implement Breadth-First Search (BFS) and Depth-First Search (DFS) traversal for a graph.
12. Find the minimum spanning tree of a graph using Prim's and Kruskal's algorithm.
13. Implement and compare sorting algorithms: Bubble Sort, Selection Sort, Insertion Sort,
14. Merge Sort, Quick Sort, and Radix Sort.
15. Implement and compare linear search and binary search.

Course Outcomes-

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

1. Implement stack operations (push, pop, peek) using both static (array-based) and dynamic (linked list-based) representations.
2. Convert an infix expression to a postfix expression and evaluate it using a stack and Expression Validation and Recursion with Stack also Check the validity of a given mathematical expression using a stack.
3. Implement basic queue operations (enqueue, dequeue) using both static (array-based) and dynamic (linked list-based) representations.
4. Implement a circular queue, a double-ended queue (deque), and a priority queue with appropriate operations.
5. Implement and perform all basic operations (create, insert, delete, display) on singly, doubly, circular, and circular doubly linked lists also its application like polynomial addition and multiplication using a singly linked list to represent polynomials.
6. Create and traverse a binary tree using preorder, inorder, and postorder traversal techniques and perform basic operations on a binary search tree (insert, delete, search) and understand its properties.

7. Represent a graph using an adjacency matrix and adjacency list and apply these representations to solve graph-related problems and also implement Breadth-First Search (BFS) and Depth-First Search (DFS) to traverse graphs.
8. Find the minimum spanning tree of a graph using Prim's and Kruskal's algorithms.
9. Implement and compare sorting algorithms such as Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, Quick Sort, and Radix Sort and understand their time complexities.
10. Implement linear search and binary search algorithms, understanding when each is best suited for different problem scenarios.

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	III					
Paper Category:	DSC1-4 **(Major)**					
Paper Name:	Fundamentals of Database Systems with SQL and PL/SQL					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives –

The aim of this course is to prepare learners for a foundational understanding of database management system:

1. To provide students with a foundational understanding of database systems, their components, and the various models used for organizing and managing data.
2. To Develop Skills for the Design and Implementation of Database Applications
3. To teach students the concepts of relational databases, normalization, transaction management, concurrency control, and database recovery.
4. To Understand User Requirements and Frame It in a Data Model
5. To Understand Creation, Manipulation, and Querying of Data in Databases
6. To gain practical skills in SQL and PL/SQL for database manipulation, querying, and implementing stored procedures and functions.

Unit I- Introduction to Database System [20]

Introduction to Database System: Definition, Limitations of traditional file system, Advantages of DBMS, Components of DBMS, Database Architecture, Database Users, Schemas and instances, 2 tier and 3 tier architecture, Database languages, Types of data models- relational, Network, Hierarchical, Distributed,

E-R model: E-R Diagram, entities, attributes and its types, Relationship and relationship sets, Cardinality, Degree, Generalization, Specialization, Aggregation. Relational Model and Database design:-Relation, Domain, Tuples, types of keys, relational integrity rules, Dr. Codd's rules,

Relational Algebra operations: - Select, Project, Cartesian Product, Union, Set difference, Natural Join, Outer Join, Dependencies and its types, Normalization and its types1NF, 2NF, 3NF, BCNF, lossless joins.

Transaction Management & Concurrency Control: -Introduction, Definition, properties, transaction states, scheduling and its types, conflict and view serializability. Introduction to Concurrency Control, problems of concurrency control. lock based protocols, timestamp-based protocol, deadlock, deadlock handling.

Database recovery and Atomicity: -Introduction, recovery algorithms, log base recovery, shadow paging, checkpoints or syncpoints or savepoints.

Unit 2: SQL and PL/SQL [10]

SQL: DDL, DML, DCL, select: From, Where, Order by, Group by, Having, Intersect, Union, Distinct, Between, In, Between, Different types of functions, Delete, Update, Insert, Nested queries, joins, create, alter and drop, constrains, index, views, Triggers, Grant, Revoke, Commit, RollBack, Savepoint

Introduction to PL/SQL: Advantages, Architecture, Datatypes, Variable and Constants, Using Built_in Functions, Conditional, Looping and Iterations Statements. Cursor in PL/SQL: Types of Cursors, Cursor Attributes, Cursor with Parameters, Cursors with LOOPs Nested Cursors, Cursors with Sub Queries and procedure. Procedures in PL/SQL: STORED PROCEDURES, PROCEDURE with Parameters (IN,OUT and IN OUT), Dropping a Procedure. Functions in PL/SQL: Difference between Procedures and Functions, types of functions and parameter modes, Exceptions in PL/SQL

Course Outcomes-

Students will able to:

1. Understand the concept of a database system and its role in overcoming the limitations of traditional file systems, such as data redundancy and inconsistency.
2. Identify the components of a Database Management System (DBMS), including storage management, query processing, and security mechanisms.
3. Learn about different database architectures (2-tier and 3-tier models) and the roles of various database users (end users, administrators, developers).
4. Explore different data models (relational, network, hierarchical, distributed) and understand their structures and applications in organizing data.

5. Gain proficiency in creating and interpreting Entity-Relationship (E-R) diagrams, and understand concepts like cardinality, degree, and generalization.
6. Study the relational model, including relations, domains, and tuples, and understand the importance of keys and relational integrity rules.
7. Learn and apply dependency and normalization techniques (1NF, 2NF, 3NF, BCNF) to eliminate redundancy and maintain data integrity.
8. Understand the ACID properties of database transactions (Atomicity, Consistency, Isolation, Durability) and their importance in ensuring data consistency.
9. Study transaction scheduling and serializability (conflict and view) to maintain consistency during concurrent transactions.
10. Explore concurrency control techniques (lock-based, timestamp-based) and recovery mechanisms (log-based, shadow paging, checkpointing) to ensure database reliability and consistency..

Reference Books-

1. Database System Concepts By KorthSilberschetz
2. Fundamentals of Database Systems by Elmsari, Navathe
3. Teach Yourself SQL in 14 Days by Jeff Parkins and Bryan Morgan
4. An Introduction to Database Systems by Bipin Desai
5. SQL and PL/SQL Programming by Ivan Bayross
6. SQL and PL/SQL Programming by Oracle Press

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	III					
Paper Category:	Practical **(Major)**					
Paper Name:	Practical based on DSC1-4					
Credit:	01		**Practical:**	2 Hrs./Week		
Marks:	**UA:**	15	**CA:**	10	**Total:**	25

Course Objectives –

The aim of this course is to enable students to:

1. Understand the creation of different tables with various constraints (primary, foreign keys).
2. Gain proficiency in writing SQL queries to insert, retrieve, and manipulate data, including complex aggregation operations like total salary expenditure.
3. Master the use of aggregate functions (SUM, AVG, MAX, MIN) and their application.
4. Learn to work with multiple tables and apply constraints (foreign keys, defaults) to establish relationships between them.
5. Understand how to write complex SQL queries involving multiple conditions and subqueries.
6. Learn to use joins effectively to retrieve data from multiple related tables (e.g., Book, Distributor, Order, etc.).
7. Gain practical experience in managing database integrity by creating triggers that prevent unauthorized inserts and deletes based on specific conditions.
8. Understand and implement PL/SQL blocks, including writing functions, procedures, and triggers for various business logic and calculations.
9. Learn to implement database packages that group related procedures, functions, and cursors for streamlined database management.

Practical based on DSC1-4:

1. Create table employee contains column eno, name, dept, basic salary, HRA, tax, deduction). Dept are D1, D2, D3 and D4. Apply different constraints and write query to.
    a. Insert 20 records.
    b. Display total amount spend by company on salary.

c. Display name of dept for which company spend maximum amount.

d. Display average salary of employee in company.

e. Display average salary of each dept.

f. Display total salary for each dept.

g. Display highest salary for each dept.

h. Display different between average of max salary for each dept and average of each dept.

i. Display no of dept in the company.

j. Display name of all employee whose basic pay is higher then average salary.

k. Display average, minimum, maximum salary of each dept.

l. Display dept average of dept whose employee >5.

2. Create following table.

   I. Book (id, title, author, publisher, category, year, price)

   II. Distributor(did, name, city, discount) and Order(order_no, title, did, qty)

   Apply different constraints and write query to.

   a. Display title and category of all books.

   b. Display the total no of books per year.

   c. Display list of authors.

   d. Display the books published in 1991,92 and 93.

   e. Display the books published from 1991 to 95.

   f. Display the books whose price is greater than200.

   g. Display the total no of books of each category.

   h. Display titles of all books whose price is greater than average price.

   i. Display the list of all books whose price is greater then average price of “computer” category.

   j. Shoe the name of all the distributors who supply “software testing” books.

   k. Display the details of all books whose price is greater than the maximum of the category average.

   l. Display name of all books who are supplying the books whose author is ‘Pressman’.

3. Create the following table & solve given queries.

Table Name : branch

Column_name	Datatype	Constraint Description
Bno	number(4)	Primary key Branch number
bname	Varchar2(20)	Not null

City	Varchar2(15)	Not null

Table Name : customer

Column_name	Datatype	Constraint Description
Cust_no	Number(6)	Primary key
Cust_name	Varchar2(20)	Not null
City	Varchar2(15)	Not null

Table Name : deposit

Column_name	Datatype	Constraint Description
Acc_no	Varchar2(5)	Primary key Starts from 'D'characcter
Cust_no	Number(6)	Foreign key references table 'customer'
Bno	Number(4)	Foreign key Branch number references from table 'branch'
Amount	Number(9,2)	Not null Default amount is 500.00
Adate	Date	Not null Date of money deposited

Table Name : borrow

Column_name	Datatype	Constraint Description
Loan_no	Number(5)	Primary key
Cust_no	Number(6)	Foreign key references table 'customer'
Bno	Number(4)	Foreign key references from table 'branch'
Amount	Number(9,2)	Not null Default amount is 500.00

a) Insert minimum 10 records.

b) describe tables, which are already created.

c) Give account number and amount of depositors.

d) Give names of borrowers.

e) Give names of customers living in city NAGPUR.

f) Give names of depositors having amount greater than 4000.

g) Give name of customer having living city BOMBAY and branch city DELHI.

h) Give names of customer having the same living city as their branch city.

i) Give name of customers who are borrowers as well as depositors and having living city NAGPUR.

j) Give name of customers who are depositors and have the same branch city as that of sunil.

k) Give names of depositors having the same living city as that of shivani and having deposit amount greater than 200.

l) Give names of borrowers having deposit amount greater than 1000 and loan amount greater than 2000.

m) Give names of borrowers having loan amount greater than the loan amount of anil.

n) Give loanno and loan amount of borrowers having the same branch as that of depositor sunil.

o) Give loanno, loan amount, account no, and deposit amount of customers living in city NAGPUR

4. Write a block to find maximum number.
5. Write a block for check given number is even or odd.
6. Write a procedure for addition of two number.
7. Write a function which return multiplication of two numbers.
8. Define cursor for display information of student.
9. Write a procedure for addition and subtraction of two numbers. (Return result).
10. Create user A and B. create table student (roll_no, name) by user A. Create trigger for avoid update or delete in table by user B.
11. Create a package for addition and multiplication of two numbers.
12. Create trigger for avoiding inserting the records whose address 'solapur' and deleting the records whose address 'satara'.(use any table with address field).
13. Create package for addition, multiplication.
14. Create function with cursor.
15. Create package which contain procedure, function , cursor.

Course Outcomes-

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

1. Students will be able to create tables with appropriate data types and constraints.
2. Students will be able to insert and query data effectively, calculating key metrics.
3. Students will be able to write SQL queries to compute and display information.
4. Students will have the ability to perform data aggregation and analysis using SQL commands to calculate statistics.
5. Students will understand the creation of relationships between tables using foreign keys and can write SQL queries that work across multiple tables to fetch related data.

6. Students will gain the ability to apply joins and subqueries to fetch data from related tables.
7. Students will learn to create triggers that prevent unauthorized inserts or updates, ensuring data integrity (e.g., preventing insertion of records with specific addresses).
8. Students will acquire practical experience in PL/SQL programming, including writing anonymous blocks, functions, and procedures to handle mathematical operations and database logic.
9. Students will be able to create packages in PL/SQL, encapsulating multiple procedures, functions, and cursors, enabling modular and reusable database management logic.

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	III					
Paper Category:	DSC2-3 **(Minor)**					
Paper Name:	Data Visualization					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives –

The aim of this course is to prepare learners:

1. To develop the ability to analyze and interpret data using exploratory visualization techniques.
2. To enable students to build both standard and advanced visualizations through customized, ground-up approaches.
3. To cultivate critical thinking skills for evaluating the quality and integrity of existing visualizations.
4. To train students in creating efficient and meaningful visualizations using real-world, large, and complex datasets.
5. To instill design principles for crafting visually appealing static and interactive visualizations with appropriate perceptual encoding.
6. To enhance visualization effectiveness through usability testing, iterative refinement, and context-sensitive design practices.

Unit I- Introduction to Power BI [15]

Introduction to Power BI: What is Business Intelligence, BI Uses and Users, Various BI Tools, Why Power BI, Introduction to Power BI, Features of Power BI, Power BI Components, Building Blocks of Power BI, Architecture of Power BI, Power BI Desktop Installation,

Loading and Transforming dataset: Load Data from different data sources, Power BI Desktop Data View, Data View and Relationship View, Creating and Deleting Relationships Manually and Automatically Data Models, Manipulating Tables, Manipulating Columns, Power BI Desktop Data Types, Data Changing Data

Type , Detecting Data Types, Categorize Data, Apply a Summarization, Sorting, Adding Hierarchies, Transforming Data Before Loading, Dataset Shaping- Renaming Columns, Reordering Columns, Removing Columns, Merging Columns, Duplicating Columns, Splitting Columns, Removing Records, Removing Duplicate Records, Filtering, Replacing Values, Using the First Row As Headers, Grouping Records, Extending the Data Model with CalculatedColumns, Creating Custom Columns, Index Columns,

Unit 2: DAX, Dashboards and Reports [15]

Power Query Editor - What is DAX, Different type of DAX functions-Aggregate functions, Date functions, Logical functions, Math functions, String functions, Trigonometric functions and other functions.

Adding Measures to the Data Model- Basic Aggregations in Measures, Cross-Table Measures, Filtering Data in Measures,

Data Visualizations- All Charts in Power BI-Types of charts, Maps in Power BI, Table and Matrix in Power BI, Subtotal and Total in Matrix, Cards and Filters in Power BI, Conditional Formatting, Slicers in Power BI slicers

Designing Power BI Dashboards and Reports-Dashboards, reports, Dashboards versus reports, KPI, KPI Visualizations, Visual selection, Layout, Single- dashboard, Multiple-dashboard, Organizational dashboards, Multiple datasets Dashboard tiles, Custom links, Images and text boxes.

Course Outcomes-

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

1. Understand the concept of Business Intelligence and explain the role and features of Power BI in BI workflows.
2. Identify and describe the components, architecture, and building blocks of Power BI.
3. Load data from various sources and perform data transformation operations using Power BI Desktop.
4. Create and manage relationships between tables to design efficient data models.
5. Apply data shaping techniques such as renaming, removing, merging, and splitting columns to prepare datasets.
6. Use Power Query Editor to create custom columns, index columns, and calculated columns.

7. Write and apply DAX (Data Analysis Expressions) functions for advanced data analysis and filtering.
8. Design effective data visualizations using charts, tables, maps, and cards in Power BI.
9. Utilize slicers, conditional formatting, and filters to enhance report interactivity and usability.
10. Build and differentiate between reports and dashboards, including the use of KPIs, multiple datasets, and organizational layouts.

Reference Books-

1. Pro Power BI Desktop-Free interactive data analysis with Microsoft Power BI by AdamAspin, Apress
2. Introducing Microsoft Power BI by Alberto Ferrari and Marco Russo, Microsoft Press
3. Mastering Microsoft Power BI by Brett Powell, Packt BIRMINGHAM– MUMBAI
4. Microsoft Power BI Complete Reference by Devin Knight, Brian Knight, Mitchell Pearson, Manuel Quintana, Brett Powell, Packt
5. Learn Power BI by Greg Deckler,Packt
6. Pro Power BI Desktop-Free interactive data analysis with Microsoft Power BI by AdamAspin, Apress

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	III					
Paper Category:	Practical **(Minor)**					
Paper Name:	Practical based on DSC2-3					
Credit:	01		**Practical:**	2 Hrs./Week		
Marks:	**UA:**	15	**CA:**	10	**Total:**	25

Course Objectives –

The aim of this course is to prepare learners:

1. Understand how to load, transform, and model data from various sources in Power BI.
2. Develop skills in shaping datasets through merging, splitting, and formatting columns.
3. Apply data cleaning techniques such as removing duplicates and replacing nulls using Power Query Editor.
4. Categorize and geo-encode data for visualization in geographic maps.
5. Use DAX functions for basic aggregations and logical operations.
6. Perform advanced text manipulations using string functions in DAX.
7. Create interactive visualizations including charts, tables, and filters.
8. Design and publish insightful dashboards incorporating KPIs and performance metrics.

Practical based on DSC2-3:

1. Load a dataset from different data sources and view it in Data View and Relationship View. Also automatically detect and create relationships between multiple tables.
2. Merge columns to create a full name from first and last name fields and split a column containing full name into first and last name.
3. Remove duplicate records from a dataset using Power Query Editor and Replace null or blank values with default values in a column and rename, reorder, and remove columns in a loaded dataset.
4. Change data types of columns (e.g., string to date, integer to decimal).
5. Categorize a column as Geography and use it in a Map visualization.

6. Create calculated columns using simple expressions (e.g., profit = revenue - cost).
7. Create measures using basic DAX aggregation functions (SUM, AVERAGE, COUNT).
8. Use logical DAX functions (IF, SWITCH) to create conditional calculations.
9. Create a calculated column using string functions (e.g., CONCATENATE).
10. Apply filtering in DAX to compute conditional sums (e.g., total sales for region = 'West').
11. Design a column chart and pie chart based on sales by region and product.
12. Use table and matrix visuals to summarize and compare data categories.
13. Apply conditional formatting to highlight top-selling products.
14. Add slicers for category and year to allow interactive report filtering.
15. Build a complete Power BI Dashboard including KPIs, filters, text boxes, and visuals with multiple pages.

Course Outcomes-

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

1. Load and relate data from multiple sources using Power BI's Data and Relationship views.
2. Apply data cleansing operations like removing duplicates, handling nulls, and column rearrangement.
3. Modify column data types to suit analytical and visualization needs.
4. Classify data fields (e.g., location) for use in geographical maps.
5. Create and apply calculated columns for deriving business insights.
6. Implement DAX aggregation functions to perform statistical analysis.
7. Use conditional and logical DAX functions to build dynamic calculations.
8. Design comprehensive visual reports with charts, slicers, and tables.
9. Build and deploy interactive dashboards using multiple visuals and real-time KPIs.

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	III					
Paper Category:	DSC2-4 **(Minor)**					
Paper Name:	PHP Programming					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives –

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

1. Understand the basic principles of PHP as a server-side scripting language.
2. Learn the syntax, variables, data types, and control structures in PHP.
3. Explore and apply PHP functions and string manipulations.
4. Understand and implement different types of arrays and exception handling in PHP.
5. Grasp the core concepts of object-oriented programming (OOP) using PHP.
6. Handle form data using GET and POST methods and manage user sessions and cookies.
7. Understand database concepts and SQL commands relevant to PHP applications.
8. Connect PHP applications to MySQL and perform CRUD operations.
9. Validate user inputs using JavaScript and display data using HTML tables.

Unit I- Introduction to PHP [10]

Introduction to PHP- Features of PHP, Advantages and Disadvantages, Client-Side vs Server-Side Scripting, Installation and Configuration of PHP

PHP Basics-PHP Syntax and Tags, Variables and Data Types, Operators and Conditional Statements, Loops: for, while, do-while, foreach, Built-in Functions, User-defined Functions, Parameter Passing and Return Values, String Handling and String Functions

Arrays and Exception Handling- Indexed, Associative, and Multidimensional Arrays, Array Functions, Error and Exception Handling, Try/Catch Block, Custom Exceptions

Unit II- Advanced PHP and Database Integration [20]

Object-Oriented Programming in PHP- Classes and Objects, Constructors and Destructors, Inheritance, Polymorphism, Access Modifiers

Handling Form Data and Sessions- HTTP Protocol Basics, GET and POST Methods, Cookies: Set, Read, Delete, Session Management

Database Connectivity with MySQL- Introduction to MySQL and SQL Basics, Connecting PHP with MySQL, CRUD Operations (Select, Insert, Update, Delete), Using PHP to Communicate with MySQL Database, Validating User Input with JavaScript, Creating Forms and Displaying Data using HTML Tables, Integrating Frontend and Backend Logic

Course Outcomes-

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

1. Describe the features, advantages, and server-side capabilities of PHP.
2. Write PHP code using appropriate syntax, variables, control structures, and loops.
3. Use built-in and user-defined functions to perform specific tasks.
4. Implement arrays and handle exceptions gracefully in PHP scripts.
5. Apply object-oriented principles such as encapsulation and inheritance in PHP projects.
6. Design dynamic forms and manage data using sessions and cookies effectively.
7. Use SQL commands for data manipulation within a PHP-MySQL environment.
8. Develop PHP scripts that connect to a MySQL database and perform data transactions.
9. Incorporate input validation and output formatting in web applications.
10. Build and deploy simple full-stack web applications using PHP and MySQL.

Reference Books-

1. The Complete Reference PHP, by Steven Holzner, TAYA McGraw-Hill Publication
2. Beginning PHP and MYSQL, by W. Jason Gilmore, Apress Publication
3. Beginning PHP, Apache, MySQL Web Development, Michael K. Glass, Yann Le Scouarnec, Elizabeth

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	III					
Paper Category:	Practical **(Minor)**					
Paper Name:	Practical based on DSC2-4					
Credit:	01		**Practical:**	2 Hrs./Week		
Marks:	**UA:**	15	**CA:**	10	**Total:**	25

Course Objectives –

The aim of this course is to prepare learners:

1. To introduce conditional logic and looping constructs in PHP for decision-making and iteration.
2. To explore built-in string functions for text processing and manipulation.
3. To enable students to work with different types of arrays and associated functions.
4. To familiarize students with error handling and custom exception mechanisms in PHP.
5. To provide a strong foundation in object-oriented programming using PHP.
6. To demonstrate the handling of web forms and form data using GET and POST methods.
7. To teach session and cookie management for maintaining user state.
8. To connect and manipulate data using PHP and MySQL for dynamic web applications.
9. To integrate front-end validation using JavaScript for secure and clean data entry.

Practical based on DSC2-4:

1. Write a PHP script using conditional statements (if, if-else, switch-case) and demonstrate the use of loops (for, while, do-while) in PHP.
2. Create and use built-in PHP string functions (strlen, strrev, strpos, etc.).
3. Write a program to perform operations on indexed arrays, also create and manipulate associative arrays and display key-value pairs.
4. Demonstrate multidimensional arrays with a student mark list example.
5. Use array functions such as array_merge, array_push, and array_pop.
6. Define a custom exception class and demonstrate throwing exceptions.

7. Create a PHP class with properties and methods and instantiate objects.
8. Use constructors and destructors in a PHP class with sample output.
9. Demonstrate all types of inheritance in PHP.
10. Implement polymorphism using method overriding in PHP.
11. Create a form using POST method and display submitted data.
12. Demonstrate session and cookies
13. Connect PHP to a MySQL database and perform CRUD Operations.
14. Validate form inputs using JavaScript (e.g., email, phone number).
15. Build a mini project that integrates HTML, PHP, JavaScript, and MySQL (e.g., login system or product catalog).

Course Outcomes-

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

1. Apply control structures (conditional statements and loops) to build dynamic logic in PHP scripts.
2. Use PHP's built-in functions to manipulate and manage strings effectively.
3. Construct and manage indexed, associative, and multidimensional arrays using appropriate functions.
4. Implement custom error and exception handling using try-catch blocks and exception classes.
5. Demonstrate object-oriented programming concepts such as classes, constructors, inheritance, and polymorphism in PHP.
6. Design and handle web forms using GET/POST methods and manage form data securely.
7. Implement sessions and cookies to maintain user state across web pages.
8. Develop dynamic web applications by connecting PHP to MySQL and performing CRUD operations.
9. Use JavaScript to validate user input on web forms, ensuring data integrity and user experience.
10. Build and deploy a mini project that integrates HTML, PHP, JavaScript, and MySQL for real-world application.

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	III					
Paper Category:	Generic/ Open Elective **(GE-3/ OE-3)**					
Paper Name:	Software Engineering					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives –

The aim of this course is to prepare learners for:

1. Understand the fundamental concepts, characteristics, and types of systems, along with the role of system elements in system development.
2. Explore the responsibilities of a System Analyst, focusing on their critical role in system analysis and design.
3. Gain foundational knowledge in Software Engineering, including software characteristics, qualities, and essential development practices.
4. Study various System Development Life Cycle (SDLC) models, including Waterfall, V-Shape, Spiral, Prototyping, Incremental, RAD, and Agile.
5. Learn to identify and define different types of software requirements (system, functional, non-functional, user) and their impact on the development process.
6. Master fact-finding techniques and design tools, such as interviews, questionnaires, flowcharts, decision tables, and structured English, for efficient system analysis and design.

Unit I- Introduction to System [15]

System concepts: Introduction system, characteristics, Elements of system, Types of system, System Analysis, Role of System Analyst. Software Engineering: Definition, Characteristics of software, Qualities of software. System Development life cycle- Waterfall model, V-shape model, Spiral model, Prototyping, incremental, RAD, Agile.

Software requirements: Types of Requirements: System, Functional, Non-functional, User.

Fact finding techniques: Interviews, Questionnaire, Record reviews, Observation. Analysis and Design Tools: Flow chart, Decision tables and Trees, Structured English, HIPO.

Unit II- System Design and Testing [15]

System Design: Data Dictionary, structured chart, Input and output design,

Coding: Coding standards, Size Estimation, Effort Estimation, and Cost Estimation,

Software Testing: Need of Testing, types of testing, Software Implementation and Maintenance: Traditional and incremental approaches, conversion methods, Overview of maintenance process, types of maintenance.

Software Quality Assurance: SQA Tasks, Goals and Metrics, Software Reliability. Software risk management: definition, types of risk, risk identification-risk monitoring and management.

Case studies: Pay Roll, Fixed Deposit, Inventory system, College Admission System, Library System, Loan system etc.

Course Outcomes-

Students will able to:

1. Students will be able to define and explain system concepts, including the characteristics and types of systems.
2. Students will understand the role and importance of system analysts in system analysis and design.
3. Students will identify and explain software characteristics and the qualities necessary for building effective software.
4. Students will be proficient in using various SDLC models, such as Waterfall, V-Shape, Spiral, Prototyping, Incremental, RAD, and Agile.
5. Students will be able to identify and define different types of software requirements and understand their role in development.
6. Students will apply fact-finding techniques and design tools, including interviews, questionnaires, flowcharts, decision tables, and structured English in system analysis and design

Reference Books-

1. Analysis and Design of Information Systems By James Senn.
2. System analysis and Business application (for case studies) By Rajesh Nike / swapna kishore.
3. Software Engineering By Pressman.
4. System Analysis and Design By Parthsarty / Khalkar.
5. Practical guide to structure System Design By Miller/Page/jones.

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	IV					
Paper Category:	DSC1-5 **(Major)**					
Paper Name:	Foundations of Core Java					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives –

The aim of this course is to prepare learners to:

1. Teach basic Java concepts like variables, data types, operators, and control flow.
2. Introduce object-oriented programming using classes, objects, constructors, and methods.
3. Explain inheritance and polymorphism using access modifiers, overriding, and interfaces.
4. Teach exception handling with try-catch, throw/throws, and file handling using Java I/O.
5. Cover multithreading with thread lifecycle, synchronization, and communication.
6. Explain Java Collection Framework using classes like ArrayList, HashMap, and Vector.
7. Teach database interaction using JDBC architecture and drivers.
8. Introduce Java Swing for building GUIs and explain differences from AWT.

Unit I- Introduction to Java Programming [20]

Introduction to Java Programming: Overview of Java, Features of Java as programming language /Platform, JDK Environment and Tools

Java Programming Fundaments: -Data types, Variables, Operators, Keywords, Naming Conventions, Structure of Java Program, Flow Control- Decision, Iterations, Arrays,

Object oriented programming in Java: Class – Members access control, Objects, Constructors, Use of 'this' keyword, Static, non-static data members and methods., public, private & protected data members Inheritance & Polymorphism-Access/Scope specifiers protected, Super, extends, single, multiple inheritance, Method overriding, Abstract classes & ADT, 'final' keyword, Extending interfaces

Exception Handling: Exceptions and Types, try..catch, finally block, throw & throws statement, user-defined exceptions, Java I/O package, byte & character stream, reader & writer, file reader & writer

Threading-Java thread lifecycle, Thread class & run able interface Thread priorities & synchronization, Usage of wait & notify

Collection framework: - Collection overview, Collection interfaces, Collection classes Vector, Array list, Hash map, Hash table

Unit II- JDBC & Swing [10]

Introduction to JDBC: Components of JDBC, Architecture of JDBC, JDBC Drivers

Introduction to swing, difference between AWT and swing, hierarchy of Swing classes, Swing controls: - JButton, JTextfield, JLabel, JCheckBox, JRadionButton, JFame, Jtable, JList, JoptionPane, JMenuitem and JMenu ,etc

Course Outcomes-

Students will able to:

1. Set up the JDK environment and explain basic Java programming concepts.
2. Write Java programs using correct syntax, data types, control structures, and arrays.
3. Define and use classes, objects, constructors, and access modifiers, including static and non-static members.
4. Implement inheritance and polymorphism using super keyword, method overriding, abstract classes, and interfaces.
5. Handle exceptions, create custom exceptions, and perform file input/output operations in Java.
6. Develop multithreaded applications, managing thread lifecycle, synchronization, and communication.
7. Utilize Java Collection Framework classes like ArrayList, Vector, HashMap, and Hashtable for data management.
8. Connect Java applications to databases using JDBC and understand its architecture and drivers.

Reference Books-

1. Java for professional developers by Michael Morgen
2. Core Java Vol 1 and vol 2 by Cay. S. Horstmann, Gray Cornell.
3. Java by Nutshell
4. Java The complete Reference by Herbert Schildt
5. Thinking in java by Brucel

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	IV					
Paper Category:	Practical **(Major)**					
Paper Name:	Practical based on DSC1-5					
Credit:	01		**Practical:**	2 Hrs./Week		
Marks:	**UA:**	15	**CA:**	10	**Total:**	25

Course Objectives –

The objective of this course is to equip students with the practical skills needed to implement and utilize Java programming language efficiently. By focusing on both fundamental and advanced topics, the course aims to enable students to:

1. Demonstrate creation and manipulation of one-dimensional and two-dimensional arrays in Java.
2. Create and use classes and constructors, including the this pointer and constructor overloading.
3. Explain object cloning and reference variables and use constructors for object initialization.
4. Implement single, multiple, and hierarchical inheritance through Java programs.
5. Define and implement abstract classes and methods in concrete classes.
6. Handle common built-in exceptions in Java with appropriate exception handling techniques.
7. Develop multithreaded programs demonstrating thread synchronization using the synchronized keyword.
8. Use Java Collections with iterators, custom sorting comparators, and manipulate various collection classes.
9. Demonstrate basic GUI programming using Swing and database connectivity using JDBC..

Practical based on DSC1-5:

1. Create a program to demonstrate one dimensional and two dimensional array.
2. Write a program to create a distance class with methods where distance is computed in terms of feet and inches, how to create objects of a class and to see the use of this pointer also demonstrates constructor overloading.

3. Modify the —distanceǁ class by creating constructor for assigning values (feet and inches) to the distance object. Create another object and assign second object as reference variable to another object reference variable. Further create a third object which is a clone of the first object.
4. Create a program to demonstrate the use of static and non-static methods/variables.
5. Create a java program to demonstrate all types of inheritance.
6. Create an abstract class Shape with an abstract method area(), and implement it in Rectangle and Circle classes.
7. Write a program that show any 5 built in exceptions.
8. Create a multithreaded program that demonstrates thread synchronization using synchronized keyword.
9. Write a program that demonstrates inter-thread communication using wait() and notify().
10. Write a program that demonstrates iteration over a collection using Iterator and sorts a collection of objects using a custom comparator.
11. Write a program to connect to a database using JDBC and perform insert, delete, update and select operation.
12. Write a simple program that creates a student information form which uses different swing components.
13. Write a program to demonstrate JTable, JOptionPane, JMenuBar and JMenuItem.
14. Write a program to demonstrate collection classes.
15. Write a program to implement interface.

Course Outcomes-

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

1. Create and manipulate one-dimensional and two-dimensional arrays in Java.
2. Create Java classes, use constructors to initialize objects, and demonstrate the use of this pointer and constructor overloading.
3. Understand and implement object cloning and reference variables, creating copies of objects and assigning them to reference variables.
4. Understanding of static and non-static methods and variables, implementing both types in Java programs.
5. Implement and differentiate between single, multiple, and hierarchical inheritance in Java.

6. Understand abstract classes and methods and implement them to solve real-world problems like calculating area using Java.
7. Identify and handle common built-in exceptions in Java, and create custom exception.
8. Create multi-threaded programs and use synchronization to ensure thread safety and prevent data inconsistency and understand and implement inter-thread communication using wait() and notify() for synchronized thread execution.
9. Learn how to iterate through collections, apply sorting with custom comparators, and manipulate data in different collection classes like ArrayList, HashMap, etc.
10. Create GUI application with any database.

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	IV					
Paper Category:	DSC1-6 **(Major)**					
Paper Name:	Python Programming					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives –

The aim of this course is to prepare learners to:

1. Understand the features and architecture of Python, including Python Virtual Machine and memory management.
2. Install and configure Python environment to write and execute basic Python programs.
3. Learn and apply Python data types, operators, and expressions effectively.
4. Develop skills to implement control statements such as conditional branching and loops in Python.
5. Gain proficiency in using core data structures like strings, lists, tuples, sets, and dictionaries.
6. Define and use functions in Python, including recursion and anonymous functions.
7. Understand the use of modules and packages for code reuse and organization.
8. Learn how to handle exceptions and errors gracefully in Python programs.
9. Acquire knowledge of file handling operations including reading, writing, and managing files.

Unit I- Introduction to Python and Core Programming Concepts [15]

Introduction to Python- Features of Python, Python Virtual Machine, memory management, garbage collection, installation and setup, writing and executing first Python programs.

Data Types and Operators-Python datatypes, literals, constants, identifiers, naming conventions, type conversions (implicit & explicit), operators, operator precedence, input/output statements, command-line arguments.

Control Statements-if, if-else, if-elif-else statements; loops: while, for, nested loops, infinite loops; indentation rules; break, continue, pass, assert, and return statements.

Data Structures- Strings, Lists, Tuples, Sets, Dictionaries-Creation and manipulation of strings, lists (including comprehensions), tuples, sets, dictionaries and their common operations.

Unit II: Functions, OOP, Exception Handling, and File I/O [15]

Functions and Modules- Functions vs methods, defining and calling functions, return values (including multiple), argument types, scope (local, nonlocal, global), recursive functions, anonymous/lambda functions, use with filter(), map(), reduce().

Modules and packages: import statement, creating and using modules, built-in modules (math, time, random), creating and importing packages.

Exception Handling-Errors and exceptions, try-except-else-finally, built-in and user-defined exceptions, assert statement.

File Handling- File types and modes, opening/closing files, reading/writing text and binary files, with statement, pickling/unpickling, file seek() and tell(), random access, zip/unzip files, directory handling.

Course Outcomes-

Students will able to:

1. Describe the features of Python and explain its memory management and execution model.
2. Demonstrate the ability to install Python and write simple programs using basic syntax and data types.
3. Write Python programs using operators, input/output statements, and command-line arguments.
4. Implement conditional statements and loops to control program flow effectively.
5. Create and manipulate Python data structures including strings, lists, tuples, sets, and dictionaries.
6. Define and invoke functions with different types of arguments and use recursion and lambda functions.
7. Create, import, and use modules and packages to structure Python programs.
8. Apply exception handling techniques to manage runtime errors in Python programs.

9. Perform file operations, including reading, writing, pickling, and handling binary files.

Reference Books-

1. Python: The Complete Reference by Martin C. Brown.
2. Core Python Programming, Dreamtech publications, by R. Nageswara Rao.
3. Python Programming, A modular approach, First Edition, Pearson, by Taneja Sheetal
4. Learning with Python, Dreamtech publications, by Allen Downey
5. Python Programming for the Absolute Beginner by Michael Dawson-Cengage Learning.

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	IV					
Paper Category:	Practical **(Major)**					
Paper Name:	Practical based on DSC1-6					
Credit:	01		**Practical:**	2 Hrs./Week		
Marks:	**UA:**	15	**CA:**	10	**Total:**	25

Course Objectives –

The objective of this course is to equip students with the practical skills needed to implement and utilize python programming language efficiently. By focusing on both fundamental and advanced topics, the course aims to enable students to:

1. Understand and apply implicit and explicit type conversions in Python programming.
2. Develop proficiency in using control flow statements, including conditional branching and loops.
3. Utilize Python data structures such as lists, sets, and dictionaries for data manipulation.
4. Design and implement functions, including the use of lambda expressions and functional programming tools.
5. Handle exceptions effectively, including creating custom exceptions for robust Python applications.
6. Perform file input/output operations and manage the file system using Python modules..

Practical based on DSC1-6:

1. Perform implicit and explicit type conversions in Python.
2. Write a program using if-elif-else to assign grades based on marks.
3. Write a nested loop program to print a multiplication table.
4. Demonstrate the use of break, continue and pass statement.
5. Use list comprehensions to generate a list of squares of numbers.
6. Create sets and perform operations like union, intersection, difference.
7. Create dictionaries and perform insert, update, delete, and search operations.
8. Write a program defining and calling a function with positional and keyword arguments.
9. Use lambda functions with filter(), map(), and reduce() on a list of numbers.

10. Create a custom Python module and import it in another program.
11. Write a program to handle multiple exceptions using try-except-else-finally.
12. Create and raise a user-defined exception class.
13. Write a program demonstrating file reading and writing in text mode and binary mode.
14. Use file seek() and tell() methods for random access in files.
15. Write a Python program to list files in a directory and create/delete directories.

Course Outcomes-

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

1. Perform type conversions and write Python programs using control statements like if-elif-else and nested loops.
2. Demonstrate the use of loop control statements (break, continue, pass) to control program flow.
3. Use list comprehensions and manipulate sets and dictionaries to implement efficient data handling.
4. Define and invoke functions with different argument types and apply lambda functions with built-in functional tools.
5. Implement exception handling in Python programs, including the creation and raising of custom exceptions.
6. Demonstrate file handling operations in text and binary modes and use file and directory management functions in Python.

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	IV					
Paper Category:	DSC2-5 **(Minor)**					
Paper Name:	Linux Essentials and Shell Programming					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives –

The aim of this course is to prepare learners to:

1. Understand the history and evolution of Linux, its architecture, and the unique features that distinguish it from other operating systems.
2. Explore the components of Linux such as the kernel, shell, and file system hierarchy, and their functionalities.
3. Learn the differences between Linux and Windows operating systems and the concept of Linux distributions and gain practical knowledge of Linux working environments like KDE, GNOME, and Xface4, including installation procedures.
4. Develop skills to manage users and groups, including creating users/groups, managing permissions, and understanding file and directory access controls (chmod, chown, chgrp).
5. Learn Linux file system management, including mounting and unmounting various storage devices.
6. Master the use of basic Linux commands for I/O operations, file and directory handling, command piping/redirection and filter commands such as grep, sed, sort, and text editors like vi and vim.
7. Understand and manage system processes, including creating, monitoring, and terminating processes, and changing process priorities.
8. Gain hands-on experience in shell programming, writing scripts for automation, and implementing control structures and loops.

Unit I- Introduction of Linux [15]

Introduction of Linux: History of Linux, Architecture of Linux system & features, Kernel, Shell & its type, Hierarchy of File system Difference between Windows and Linux. Linux Distributions, Working environments: KDE, GNOME, Xface4, Installation procedure of Linux

Users & Groups Management: - Create Users, Create groups, Special groups, Assigning permissions to users and Groups, File and Directory permissions- chmod, chown, chgrp.,

Linux File System:-, Mounting devices (CD/DVD, usb, hard drive partition, file system)

Linux Command: I/O and Redirection, Piping, mkdir, rmdir, cd and pwd, file, ls, cat, more, less, File and Directory Operations: find, cp, mv, rm, ln etc, Printing the files - lpr, lpq, lprm etc.,

Filter Commands & Editor:- Filters: head, tail , pr, cut, paste, sort, uniq, tr, grep, egrep, fgrep, sed., Communication commands:- mesg, talk, write, wall, mail., Text Editors- vi, vim, Archive and File compression commands,

Unit 2: - Linux System Management and Administration [15]

Process Management: Shell process, Parent and children, Process status, System process, Multiple jobs in background and foreground, Changing process priority with nice. Listing processes, ps, kill, premature termination of process.,

Disk management and System Administration:-Disk Partitioning- RAID, LVM etc., disk related Management Tools- Fdisk, Parted etc. , Boot Loaders- GRUB, LILO, Custom Loaders, System administrator, Configuration and log files, Chkconfig, Security Enhanced Linux, Installing and removing packages with rpm command, Understanding various Servers:- DHCP, DNS, Squid, Apache, Telnet, FTP, Samba.

Shell Programming: -Introduction to shell scripting, Writing and executing simple shell scripts, Variables, data types, and operators in shell scripting, Meta characters, Control structure, Loop structure and case statement, Writing and using shell functions, passing arguments to shell scripts and functions, Returning values from functions

Course Outcomes-

Students will able to:

1. Understand and explain the architecture, features, and history of Linux, along with its kernel, shell, and file system hierarchy.
2. Differentiate between Linux and Windows operating systems and evaluate various Linux distributions and working environments.
3. Demonstrate the ability to create users and groups, assign permissions, and manage file and directory permissions using relevant commands.
4. Manage and navigate the Linux file system effectively, including mounting storage devices and performing file operations.
5. Utilize Linux commands for file manipulation, redirection, and piping to perform efficient system operations and also manipulate data using filter commands and text editors like vi and vim for efficient file editing and data processing.
6. Understand system processes, including their hierarchy, status, and priority management, using tools like ps and kill.
7. Configure system settings, manage disk partitions (RAID, LVM), and work with boot loaders like GRUB and LILO and also configure and understand various Linux servers, including Apache, Samba, FTP, and DNS, and manage software packages using commands like rpm.
8. Write and execute shell scripts using variables, control structures, loops, and functions, enabling automation of repetitive tasks.

Reference Books-

1. Official Red Hat Linux Users guide by Redhat, Wiley Dreamtech India
2. Beginning Linux Programming by Neil Mathew & Richard Stones, Wiley Dreamtech India
3. Red Hat Linux Bible by Cristopher Negus, Wiley Dreamtech India
4. UNIX Shell Programming by Yeswant Kanethkar, BPB
5. Shell Scripting: Expert Recipes for Linux, Bash, and More" by Steve Parker
6. Classic Shell Scripting" by Arnold Robbins and Nelson H.F. Beebe
7. Learning the bash Shell: Unix Shell Programming" by Cameron Newham and Bill Rosenblatt

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	IV					
Paper Category:	Practical **(Minor)**					
Paper Name:	Practical based on DSC2-5					
Credit:	01		**Practical:**	2 Hrs./Week		
Marks:	**UA:**	15	**CA:**	10	**Total:**	25

Course Objectives –

The aim of this course is to enable students to:

1. To develop a strong foundation in Linux file system navigation and directory management using basic commands.
2. To explore file viewing and manipulation techniques, ensuring efficient management and handling of data.
3. To understand and implement file permissions and ownership settings for effective file security.
4. To equip students with the skills to monitor and manage running processes and system performance metrics.
5. To introduce students to network configuration and interface management using Linux commands.
6. To provide knowledge of managing users and groups for better system administration.
7. To enable students to write shell scripts that include variables, control structures, loops, and functions for automating tasks.
8. To teach students to process and analyze data using Linux tools and commands like grep, awk, and sed.

Practical based on DSC2-5:

1. Write a script to demonstrate navigating the file system using commands like ls, cd, and pwd. Create, list, and remove directories using mkdir and rmdir.
2. Write a script to view file contents using commands such as cat, less, more, head, and tail with appropriate options.
3. Create a script that performs file and directory operations using cp, mv, rm, and touch. Include error checking for file existence.

4. Write a script to demonstrate changing file permissions with chmod and changing ownership using chown and chgrp also accepts a file name and displays its permissions in symbolic and numeric formats.
5. Write a script to display running processes and their resource usage using ps, top, and htop and demonstrate managing processes using kill, killall, pkill, and pgrep. Also monitor disk space, directory sizes, and memory usage using df, du, and free.
6. Develop a script that shows how to configure network settings using ifconfig, ip, and netstat. Include commands to display network interfaces.
7. Write a script to manage users and groups using useradd, usermod, groupadd, and passwd.
8. Create a script that demonstrates variable declaration, manipulation, and printing the results.
9. Write a shell script using if-else to check if a given file exists and is readable, writable, or executable.
10. Develop a script to demonstrate for, while, and until loops by iterating over files in a directory and performing operations on them.
11. Write a shell script to define and call functions for common tasks like addition and subtraction of two numbers.
12. Write a script to pass arguments to a function and return values using the return statement.
13. Create a script to read data from a file and process it using commands like grep, awk, and sed.
14. Write a script that demonstrates the use of redirection operators (>, >>, <) and pipes to redirect and process input/output streams.
15. Write a shell script to handle errors gracefully by using exit codes and displaying custom error messages.

Course Outcomes-

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

1. Demonstrate the ability to navigate and manipulate the Linux file system using commands like ls, cd, mkdir, and rmdir.
2. Efficiently view and analyze file content using commands such as cat, less, head, and tail with appropriate options.
3. Perform file and directory operations like copy, move, and remove, while implementing error handling for non-existent files.

4. Apply knowledge of file permissions and ownership changes using commands like chmod, chown, and chgrp.
5. Monitor and manage processes, disk usage, and memory usage using ps, top, df, du, and free commands.
6. Configure and display network settings and interfaces using ifconfig, ip, and netstat.
7. Write and execute efficient shell scripts to automate tasks, incorporating loops, conditionals, and functions.
8. Process and manipulate data using advanced Linux commands and demonstrate effective error handling and stream redirection in scripts.

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	IV					
Paper Category:	DSC2-6 **(Minor)**					
Paper Name:	Software Testing					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives –

The aim of this course is to prepare learners to:

1. Understand the importance, types, and principles of software testing including differences between manual and automation testing.
2. Learn and apply various white box and black box testing techniques.
3. Gain knowledge of different testing levels including functional and non-functional testing.
4. Develop the skills needed to design and write effective test cases and reports.
5. Understand and implement different stages of the Software Testing Life Cycle (STLC).
6. Study the defect life cycle and learn techniques to identify, track, and report bugs.
7. Explore the basics of automated testing using tools like Selenium and frameworks.

Unit I- Fundamentals of Software Testing and Manual Testing Techniques [15]

Introduction to Software Testing: Importance and need for software testing, Differences between Manual and Automation Testing,

White Box Testing: Advantages and Disadvantages, Static Techniques: Informal Reviews, Walkthroughs, Technical Reviews, Inspection, Dynamic/Structural Techniques: Statement Coverage Testing, Branch Coverage Testing, Path Coverage Testing, Conditional Coverage Testing, Loop Coverage Testing,

Black Box Testing: Advantages and Disadvantages, Techniques: Boundary Value Analysis, Equivalence Class Partitioning, State Transition Technique, Cause-Effect Graph, Decision Table Testing, Use Case Testing, Experience-Based Techniques: Error Guessing, Exploratory Testing,

Levels of Testing: Functional Testing: Integration Testing (Top Down, Bottom Up, Non-Incremental), System Testing, Acceptance Testing: Alpha, Beta Testing, Smoke Testing, Regression Testing: Unit, Regional, Full

Regression Testing, Non-Functional Testing: Adhoc, Performance Testing (Load, Stress, Volume, Soak, Recovery Testing)

Unit II: Test Design, Life Cycles & Automation Testing [15]

Test Case Design Techniques: Introduction to Test Cases and Types, Test Case Template, Writing Effective Test Cases with Examples, Preparing Review Reports

Software Testing Life Cycle (STLC): Writing Test Plan, Preparing Traceability Matrix, Writing Test Execution Report and Summary Report,

Defect Life Cycle: Difference between Bug, Defect, Failure, Error, Defect Tracking and Reporting

Types of Bugs: Identifying and Reporting Bugs, Introduction to Automated Testing, Installing and Configuring Selenium Testing Tool,

Case Studies: Test Case Design for Login Page, Internet Banking Login, Online Shopping

Course Outcomes-

Students will able to:

1. Explain the purpose of software testing and differentiate between manual and automation testing approaches.
2. Apply white box and black box testing techniques for thorough test coverage.
3. Demonstrate the ability to perform various types and levels of testing including system, regression, and performance testing.
4. Create test cases using appropriate templates and write review reports effectively.
5. Execute all phases of STLC including test planning, traceability, and reporting.
6. Identify, document & manage software defects using standard defect lifecycle practices.
7. Install and use Selenium to create and automate test cases for real-world applications..

Reference Books-

1. The art of Software Testing–Glenford J. Myers
2. Lessons learned in Software Testing– Cem Kaner, James Bach, Bret Pettichord
3. A Practitioner's Guide to Software Test Design- Lee Copeland.

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	IV					
Paper Category:	Practical **(Minor)**					
Paper Name:	Practical based on DSC2-6					
Credit:	01		**Practical:**	2 Hrs./Week		
Marks:	**UA:**	15	**CA:**	10	**Total:**	25

Course Objectives –

The course aims to enable students to:

1. To understand and apply various white-box testing techniques such as statement, branch, and conditional coverage.
2. To evaluate and implement black-box testing techniques like decision tables and use case testing.
3. To develop and execute integration testing strategies using stubs and drivers.
4. To formulate and manage different levels of testing like system testing, acceptance testing, and smoke testing.
5. To plan and perform regression testing including unit, regional, and full testing.
6. To design effective manual test cases, test plans, and use standard templates for documentation.
7. To install, configure, and apply automated testing tools like Selenium WebDriver and TestNG in real-world scenarios.

Practical based on DSC2-6:

1. Perform a statement coverage analysis on a given code snippet.
2. Conduct branch coverage testing for conditional logic code.
3. Design a conditional coverage test with multiple Boolean conditions.
4. Test a simple loop for loop coverage (zero, one, many iterations).
5. Create and use a decision table for testing a discount policy.
6. Perform use case testing for a login system.
7. Demonstrate top-down integration testing using stubs.
8. Perform bottom-up integration testing using drivers.
9. Prepare a system test plan for a Library Management System.

10. Create alpha and beta test plans for a product release.
11. Conduct smoke testing for a sample web application.
12. Design and demonstrate unit, regional, and full regression testing.
13. Write effective test cases using a standard template.
14. Install and configure Selenium WebDriver and execute a sample automation script.
15. Design automated test cases using TestNG for Login, Banking, and Shopping scenarios..

Course Outcomes-

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

1. Analyze and achieve coverage goals through statement, branch, and conditional testing.
2. Design and apply black-box testing techniques such as decision tables and use case testing.
3. Perform integration testing using both top-down and bottom-up approaches.
4. Prepare system and acceptance test plans for software applications.
5. Demonstrate effective regression testing at various levels.
6. Write well-structured manual test cases using professional templates and documentation methods.
7. SImplement automated testing with Selenium and TestNG for web applications.

B.Sc. (Computer Science)-II, Level – 5.0 UG Diploma Level						
Sem:	IV					
Paper Category:	Generic/ Open Elective **(GE-4/ OE-4)**					
Paper Name:	Fundamentals Data Communication and Network Systems					
Credit:	02		**Theory:**	2 Hrs./Week		
Marks:	**UA:**	30	**CA:**	20	**Total:**	50

Course Objectives –

The aim of this course is to prepare learners:

1. To introduce the fundamentals of data communication, including components, data flow, protocols, standards, and network models like ISO-OSI and TCP/IP.
2. To impart knowledge of signal properties, transmission impairments, and various transmission media, both guided and unguided.
3. To explain digital transmission techniques, modulation methods, and error detection and correction mechanisms.
4. To familiarize students with multiple access protocols, channelization methods, and data link control protocols.
5. To discuss network design issues, routing algorithms, congestion control strategies, and network devices.
6. To explore the functionalities of TCP/IP protocols, upper layers, and concepts of audio and video compression.

Unit I- Introduction to Data Communication & Networking [15]

Data Communication: Components, Data Flow, Protocols & Standards, Design Issues of Layers, Connection oriented and connection less services, Network models :- ISO-OSI reference model, TCP/IP reference model.

Physical layer: Signals: Analog & Digital Signals, Period, Frequency, Phase, Amplitude, Bandwidth, Bit Rate, Bit, Length, Fourier analysis. Transmission Impairment: Attenuation, Distortion, Noise, Nyquiest Theorem, Shannon Capacity Theorem.

Transmission Media:-Guided Media-Magnetic Media, Twisted Pair, Coaxial Cable, Fiber Optic Cable, Unguided Media:-Wireless- Radio Waves, Microwaves, Infrared, Satellite Communication, Digital Transmission: Manchester & Differential Manchester Coding, Pulse Code Modulation., Modulation:-Amplitude Modulation, Frequency Modulation, Phase Modulation,. Transmission Mode: Parallel, Serial, Synchronous Transmission, Asynchronous Transmission., Multiplexing- Frequency Division Multiplexing, Time Division Multiplexing, Wavelength Division Multiplexing., SwitchingCircuit Switching, Message Switching, Packet Switching.

Unit 2: - Data link layer [15]

Error Detection & Correction: Types of Errors, Hamming Distance, Error Detection: Parity Check, Cyclic Redundancy Check, Checksum Check, hamming code., Data Link Control: Framing, Flow & Error Control, Protocols: Simplex, Stop and Wait, Stop and Wait ARQ, Go Back N ARQ, Selective repeat ARQ, HDLC, Point to Point protocol. Multiple Access

Protocol: ALOHA, CSMA, CSMA/CD, CSMA/CA Channelization, FDMA, TDMA, CDMA

Network layer , Transport, Session, Presentation & Application layers: Network layer Design issues, Routing Algorithm: Optimality Principle, Shortest Path Routing, Distance Vector Routing, Link State Routing., Congestion Control Algorithm: General principle of congestion control, Congestion, prevention policies, Congestion Control in Virtual-Circuit Subnets, Congestion Control in Datagram Subnets, Network DevicesHubs, Switches, Repeaters, Bridges, Routers, Gateways, Transport, Session, Presentation & Application layers. TCP/IP

protocol suite :- UDP,TCP,SCTP, IP, RTP, FTP, DNS, TELNET, SMTP, POP, HTTP, WWW, SNMP,ARP, RARP., Data Compression:-Audio Compression, Video Compression

Course Outcomes-

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

1. Understand and describe data communication components, protocols, standards, and network models (ISO-OSI, TCP/IP).

2. Analyze signal properties, transmission impairments, and principles of guided and unguided transmission media.
3. Apply digital transmission coding techniques, modulation methods, and error detection/correction techniques.
4. Implement and evaluate multiple access protocols, channelization schemes, and data link control methods.
5. Analyze routing algorithms, congestion control mechanisms, and functionalities of TCP/IP protocols.
6. Explain and apply concepts of audio and video compression in data communication systems.

Reference Books-

1. Computer Networking by Tannenbaum.
2. Data communication and networking by William Stallings
3. Data communication and networking by B A Forouzan
4. Data communication and networking by Jain

Equivalence subject list

Sem-III	
Data Structures	Introduction to Data Structures and Applications **(Sem-III)**
Software Engineering	Software Engineering **(GE3/OE3)**
Sem-IV	
Core Java	Foundations of Core Java **(Sem-IV)**
DBMS Using Oracle	Fundamentals of Database Systems with SQL and PL/SQL **(Sem-III)**