### III – Semester (II – Year)

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Subject Code</th>
<th>Subject</th>
<th>Hours Per Week</th>
<th>Credits</th>
<th>Scheme of Evaluation</th>
<th>Maximum Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>L</td>
<td>T</td>
</tr>
<tr>
<td>1</td>
<td>21BS3MT03</td>
<td>Mathematical &amp; Statistical Foundations</td>
<td>3</td>
<td>1</td>
<td>-</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>21HS3EG03</td>
<td>English for Employability</td>
<td>1</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>21PC3CS01</td>
<td>Data Structures</td>
<td>3</td>
<td>1</td>
<td>-</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>21PC3CS02</td>
<td>Data Base Management Systems</td>
<td>3</td>
<td></td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>21PC3CS03</td>
<td>Object Oriented Programming using Java</td>
<td>3</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>21PC3CS04</td>
<td>Database Management Systems-Lab</td>
<td>-</td>
<td></td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>21PC3CS05</td>
<td>Data Structures using C -Lab</td>
<td>-</td>
<td></td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>21PC3CS06</td>
<td>Programming using Java- Lab</td>
<td>-</td>
<td></td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>21HS3EG04</td>
<td>English for Employability-Lab</td>
<td>-</td>
<td></td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>21PR3IN01</td>
<td>Evaluation of Summer Internship-I</td>
<td>-</td>
<td>1</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>TOTAL</strong></td>
<td>21.5</td>
<td>370</td>
<td>630</td>
<td>1000</td>
</tr>
</tbody>
</table>

### Non Credit Courses

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Subject Code</th>
<th>Subject</th>
<th>Hours Per Week</th>
<th>Credits</th>
<th>Scheme of Evaluation</th>
<th>Maximum Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>L</td>
<td>T</td>
</tr>
<tr>
<td>11</td>
<td>21MC3HS02</td>
<td>Environmental Science</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### IV – Semester (II – Year)

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Subject Code</th>
<th>Subject</th>
<th>Hours Per Week</th>
<th>Credits</th>
<th>Scheme of Evaluation</th>
<th>Maximum Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>L</td>
<td>T</td>
</tr>
<tr>
<td>1</td>
<td>21ES4MT04</td>
<td>Matrix Computations and Its Applications</td>
<td>3</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>21ES4CS03</td>
<td>Python Programming</td>
<td>3</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>21PC4CS07</td>
<td>Design &amp; Analysis of Algorithms</td>
<td>3</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>21PC4CS08</td>
<td>Operating Systems</td>
<td>3</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>21PC4CS09</td>
<td>Computer Networks</td>
<td>3</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>21PS4CS11</td>
<td>Operating Systems-Lab</td>
<td>-</td>
<td></td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>21PC4CS10</td>
<td>Design &amp; Analysis of Algorithms-Lab</td>
<td>-</td>
<td></td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>21ES4CS04</td>
<td>Python Programming-Lab</td>
<td>-</td>
<td></td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>21PC4CS12</td>
<td>Computer Networks-Lab</td>
<td>-</td>
<td></td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>21PR4CD01</td>
<td>DS-Doing Engineering-I</td>
<td>-</td>
<td></td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>TOTAL</strong></td>
<td>22.5</td>
<td>300</td>
<td>700</td>
<td>1000</td>
</tr>
</tbody>
</table>
MATHEMATICAL AND STATISTICAL FOUNDATION
(Common to CSC/ CSD/CSM/CSO)

Pre-requisite: Basic knowledge of set and relations theory, permutations, combinations, Venn diagrams, mean, median, mode and testing of hypothesis of different samples

Course Objectives: To provide the student with
   Different random variables and various probability distribution functions.
   • The discrete and continuous of the Bernoulli, binomial, Poisson and the Normal distributions.
   • Measures of Central tendency and curve fitting by the method of least squares.
   • Concept of point and interval estimations and concept of test of hypothesis and test of significance.
   • How to apply the five-step test procedure for test of hypothesis concerning a population mean when the sample size is small.

MODULE I
Unit 1: PROBABILITY

Unit 2: RANDOM VARIABLES

MODULE II
Unit 1: DISCRETE DISTRIBUTIONS
Uniform, Binomial and, Poisson distribution and statistical constants of these distributions using moment generating function

Unit 2: CONTINUOUS DISTRIBUTIONS
Uniform Distribution, Exponential Distribution, Normal Distribution and statistical constants of these distributions using moment generating function

MODULE III
Unit 1: BIVARIATE DISTRIBUTION
Joint Probability distributions - Joint Probability mass function, joint probability density function, Marginal Distribution, Covariance of two random variables.

Unit 2: CORRELATION and REGRESSION
Karl Pearson coefficient of correlation, Rank correlation, Regression coefficient, Lines of regression.
MODULE IV
Unit 1: SAMPLING DISTRIBUTION:
Random Sampling, Some Important Statistics, Sampling Distributions, Sampling Distribution of Means, variance and the Central Limit Theorem.

Unit 2: ESTIMATION:
Estimating the Mean, Standard Error of a Point Estimate, Prediction Intervals, Tolerance Limits, Estimating the Variance, Estimating a Proportion for single mean, Difference between Two Means, between Two Proportions for Two Samples

MODULE V
Unit 1: GREATEST COMMON DIVISORS AND PRIME FACTORIZATION:
Greatest common divisors, The Euclidean algorithm, The fundamental theorem of arithmetic, Factorization of integers and the Fermat numbers

Unit 2: CONGRUENCES:
Introduction to congruences, Linear congruences, The Chinese remainder theorem, Systems of linear congruences.

Text Books:

Reference Books:

MOOC courses:
1. Probability: http://nptel.ac.in/courses/111105041/
2. Probability and Statistics: http://nptel.ac.in/courses/111105035/
3. Probability: https://nptel.ac.in/courses/111/102/111102111/

E-Books:
   https://1lib.in/book/1196320/1672b
2. Miller & Freund’s Probability and Statistics for Engineers
   https://1lib.in/book/2883098/927b28

3. Statistics and Probability for Engineering Applications with Microsoft Excel
   https://1lib.in/book/487476/b6daef

4. Charles M Grinsted, J Laurie Snell “Introduction to Probability”, American Mathematical Society
   https://1lib.in/book/556778/1a0fc2
**Course Outcomes:** After learning the contents of this paper the student must be able to

CO1: Solve the probability and the probability distribution function of random variables.

CO2: Evaluate random processes which occur in engineering applications governed by the Binomial, Poisson, Normal and Exponential distributions.

CO3: Apply Inferential Statistics to make predictions or judgments about the population from which the sample data is drawn.

CO4: Apply the number theory concepts to cryptography.

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PO 1</td>
</tr>
<tr>
<td>CO1</td>
<td>3</td>
</tr>
<tr>
<td>CO2</td>
<td>3</td>
</tr>
<tr>
<td>CO3</td>
<td>3</td>
</tr>
<tr>
<td>CO4</td>
<td>3</td>
</tr>
</tbody>
</table>
English For Employability
(Common to CSE/EEE/ECE/CSM/CSD/CSC/CSO/MECH)

Course Objective:
1. To introduce growing need of English Language Skills for Employability (ELSE).
2. To help the students understand the importance of inter-personal communication, team dynamics and role behaviour at the work place
3. To enhance their Behavioural skill and Entrepreneurship skills.

Module I- Wings of Fire
Unit 1: “Orientation” an excerpt from Wings of fire-An Autobiography of Abdul Kalam by Arun Tiwari.
Unit 2: Grammar
  Vocabulary- Omission of Articles, Collective Nouns, Prepositions, Collocations.
  Reading- Observation Passage, Survey Passage
  Writing- Comprehension Passages based on graphs, charts & other illustrations.

Module II 5 Points Someone
Unit 1: The Gift an excerpt from 5 Points Someone by Chetan Bhagat
Unit 2: Grammar -
  Vocabulary- Advanced Collocations, Proverbs, Idioms, One word Substitute
  Reading – Complex passage, Reading Comparison,
  Writing- Usage of Idioms and Proverbs in Passage

Module III – Wise Leaders Wanted & Shift Your Perspective: Connect to Your Noble Purpose
Unit 1: “Wise Leaders Wanted & Shift Your Perspective: Connect to Your Noble Purpose” an excerpt from From Smart to Wise: Acting and Leading with Wisdom Kaipa, Prasad, and Navi Radjou.
Unit 2: Grammar
  Vocabulary- Technical vocabulary, Auxiliaries and Modals,
  Reading- Technical Comprehension,
  Writing- Creative Resume.

Module IV- Variation Under Nature
Unit 1: “Variation Under Nature” an excerpt from Origin of Species by Charles Darwin
Unit 2: Grammar
  Vocabulary- Coherence-Cohesive devices, Figures of speech
  Reading- Inferring Reading, Survey, Question, Read, Recite and Review (SQ3R)

Module V- Let’s Build a Company: A Start-up Story Minus the Bullshit
Unit 1: Let's Build a Company: A Start-up Story Minus the Bullshit by Harpreet Grover and Vibhore Goyal
Unit 2: **Grammar**  
**Vocabulary**- Topic/Situation based Vocabulary, Tongue Twisters.  
**Reading**- Critical Reading of known/unknown passages  
**Writing**- Common Errors in Tenses, Description of hobbies, Future plans,  
Reported Speech: Direct & Indirect Speech, Email Writing, Formal letter writing (Enquiry, Apology, Leave, Request) Notice Writing, Information Transfer, Technical report writing

**Textbooks**

**References**
5. Murphy, R. *Intermediate English Grammar*.  

**Course Outcomes**: After undergoing this course, the student will be able to;  

**CO1**: **Understand** the concept and process of employability skills.  
**CO2**: **Demonstrate** the employability skills in both verbal and non-verbal communications.  
**CO3**: **Apply** skill identification strategies to bring out the results on social and industry demands.  
**CO4**: **Recognize** right professional, Entrepreneurship skills and social ethical values.

<table>
<thead>
<tr>
<th>CO-PO Mapping Chart</th>
<th>(3/2/1 indicates strength of correlation) 3 – High; 2 – Medium; 1 – Low</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Course Outcomes (COs)</strong></td>
<td><strong>Program Outcomes (POs)</strong></td>
</tr>
<tr>
<td></td>
<td>PO1</td>
</tr>
<tr>
<td>CO1</td>
<td></td>
</tr>
<tr>
<td>CO2</td>
<td></td>
</tr>
<tr>
<td>CO3</td>
<td></td>
</tr>
<tr>
<td>CO4</td>
<td></td>
</tr>
</tbody>
</table>
Subject: 21PC3CS01  
Subject Code: DATASTRUCTURES

Prerequisite: Knowledge of C Programming

Course Objectives:
1: To make student knowledgeable in data structure concepts
2: To make students apply the data structure concepts in relevant applications
3: To make students apply the graph techniques in relevant applications
4: To make students determine the best sorting technique to process the data

Course Outcomes:
At the end of the course the students will be able to:
1: Identify the concepts of data structures in relevant applications
2: Apply the concept of data structures for the new situations
3: Apply the graph traversal techniques to solve map related problems
4: Analyze the best sorting technique pertaining to the area applied

Module 1:
Algorithm Analysis: Asymptotic analysis, simple justification techniques, Recursion, Illustrative examples, Analyzing Recursive algorithms, Designing Recursive algorithms.
Sorting: Bubble sort, selection sort, Insertion sort, Merge sort, Quick Sort, Randomized Quick sort. Comparison of sorting algorithms, Polynomials and Sparse Matrices.

Module 2:

Module 3:
Linked Lists – Singly Linked List, Implementation using arrays, Implementation using Queues, Circular Linked List- Doubly linked list- Implementation of doubly linked list, implementation using De queue.
Trees- General trees, Binary trees, Implementation using array and linked representation, tree traversals- in order, preorder, post order, Breadth-first traversal, Implementation, Application of tree traversals.

Module 4:
Priority Queue-ADT, Implementation, Heap, sorting with a priority Queue, Adaptable priority Queues Hash Tables and Skip Lists-Maps and Dictionaries, Hash tables, Sorted Maps, Skip lists, Sets, Multi sets and Multi maps
Graphs: Definitions, Terminologies, Matrix and Adjacency List Representation of Graphs, Traversal methods: Breadth First Search and Depth First Search.
Module 5:

**Search trees** - Binary search tree, Implementation, Operations, Balanced Search trees - AVL Trees, Splay trees, (2, 4) trees, Red Black trees.


**Text Books:**

**Reference Books:**
2. Data Structures using C by E. Balaguruswamy.
3. Data Structures, S. Lipscutz Schumaum; Outlines, TMH

**CO-PO/PSO Mapping:**

<table>
<thead>
<tr>
<th>CO-PO</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>PSO1</th>
<th>PSO2</th>
<th>PSO3</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO-1</td>
<td>H</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>M</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO-2</td>
<td>M</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>CO-3</td>
<td>M</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>M</td>
</tr>
<tr>
<td>CO-4</td>
<td>M</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Database Management Systems

Prerequisite: Basics of computer programming language.

Course Objectives:
1. To enable students, define and describe basic concepts of database managements and applications.
2. To master the basics of SQL and design queries using SQL.
3. To provide students the theoretical concepts of data models and database design
4. To make students familiarize with relational model, relational algebra, transaction control and concurrency control.
5. To introduce storage structures and access techniques

Course Outcomes:
1. Describe fundamentals of DBMS, database design and normal forms.
2. Design SQL commands and queries for retrieval and management of data.
3. Summarize database storage structures and access techniques.
4. Understand basics of transaction processing and concurrency control.

Module I
Database System Applications: A Historical Perspective, File Systems versus a DBMS, the Data Model, Levels of Abstraction in a DBMS, Data Independence, Structure of a DBMS
Introduction to Database Design: Database Design and ER Diagrams, Entities, Attributes, and Entity Sets, Relationships and Relationship Sets, Additional Features of the ER Model, Conceptual Design with the ER Model

Module II
Relational Model: Integrity constraint over relations, enforcing integrity constraints, querying relational data, logical data base design, introduction to views, destroying/altering tables and views.
Relational Algebra and Calculus: Selection and Projection, Set operations, Joins, Division, More examples on Algebra queries, Tuple relational Calculus, Domain Relational Calculus. Form of Basic SQL Query, Data Definition commands, Data manipulation commands.

Module III
Advanced SQL: SET operators, Data Time Functions, Aggregate Operators, Group by clause, Joins, Advanced Sub queries, Nested Queries, triggers and active data bases.

Module IV
Storage and Indexing: Data on External Storage, File Organization and Indexing, Cluster Indexes, Primary and Secondary Indexes, Index data Structures, Hash Based Indexing, Tree base Indexing, Comparison of File Organizations, Indexes and Performance Tuning
Tree Structured Indexing: Intuitions for tree Indexes, Indexed Sequential Access Methods
Module V
**Transaction Management:** The ACID Properties, Transactions and Schedules, Concurrent Execution of Transactions, Lock-Based Concurrency Control, Performance of Locking, Transaction Support in SQL, Introduction to Crash Recovery

**Concurrency Control:** 2PL, Serializability, and Recoverability, Introduction to Lock Management, Lock Conversions, Dealing with Deadlocks, Specialized Locking Techniques, Concurrency Control without Locking

**Text Books:**


**Reference Books:**


**CO-PO/PSO Mapping:**

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
<th>Program Specific Outcomes*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PO 1</td>
<td>PO 2</td>
</tr>
<tr>
<td>CO1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CO2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CO3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CO4</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>
Prerequisites: Knowledge of Programming

Course Objectives:
1. Understand fundamental concepts and constructs of Java
2. Implement Different object-oriented Concepts in Java.
3. Write error free program using exception handling and multithreading
4. Develop GUI application

Course Outcomes:
1. Solve the given problem using OOPS technique.
2. Explain the concept of Package and Interfaces.
3. Implement Multi threading and Exception handling in java
4. Develop GUI based application using applet, awt, Event handling and swing.

Module I
History and Evolution of java: Java’s lineage, Java and internet, Byte code, Java buzzwords, Evolution of java.
Object oriented programming - data, types, variables, scope and life time of variables, arrays, operators, expressions, control statements, type conversion and casting, simple java programs, concepts of classes, objects, , methods, constructor, this and static keywords , garbage collection, overloading methods, parameter passing, access control, command line arguments, auto boxing, exploring String and String Buffer class
Inheritance: Types of inheritance, member access rules, super and final keywords, the object class and its methods, method overriding, dynamic method dispatch, abstract classes and methods.

Module II
Packages and Interfaces: Defining, Creating and Accessing a Package, understanding CLASSPATH, importing packages, Differences between classes and interfaces, defining an interface, implementing interface, applying interfaces, variables in interface and extending interfaces.
Exception handling: Concepts of exception handling, benefits of exception handling, exception hierarchy, usage of try, catch, throw, throws and finally, built in exceptions, creating own exceptions.

Module III
Multithreading: Differences between multi-threading and multi programming, thread life cycle, creating threads, thread priorities, synchronizing threads, inter thread communication.
I/O Streams: Types of Streams: Byte and character streams, File class, reading and writing files, reading and writing from console, Random access file, automatically close a file, serialization.

Module IV
Applets: Concepts of Applets, differences between applets and applications, life cycle of an applet, creating applets, passing parameters to applets.
**AWT:** class hierarchy, user interface components- labels, buttons, scrollbars, text components, checkbox, checkbox groups, choices, lists panels – scroll pane, dialogs, menu bar, Layout Managers- Flow Layout, Border Layout, Grid Layout, Card Layout, Grid Bag Layout.

**Event Handling:** Events, Event sources, Event classes, Event Listeners, Delegation event model, handling mouse and keyboard events, Adapter classes.

**Module V**

**Swings:** Introduction, limitations of AWT, MVC architecture, components, containers, exploring swing- J Applet, J Frame and J Component, Image Icon, J Label, J Text field, J Button, J Checkbox, J List, J Radio button, J Combo Box, J Tabbed Pane, J Scroll Pane.

**The Collections Framework (java.util)**- Collections overview, Collection Interfaces, Generics The Collection classes- Array List, Linked List, Hash Set, Tree Set, Priority Queue, Array Deque. Accessing a Collection via an iterator, Using an Iterator, The For-Each alternative, Map Interfaces and Classes, Comparators, Collection algorithms, Arrays, The Legacy Classes and Interfaces- Dictionary, Hash table, Properties, Stack, Vector More Utility classes, String Tokenizer, Date, Calendar, Random, Scanner

**Text Books:**

**Reference Books:**
1. An Introduction to JAVA Programming (Chapter 6) Author: Y.Daniel Liang , Publishers: Tata Mec-Hill.
3. Internet and Java Programming, R.KrishnaMurthu and S.Prabhu, New Age Publishers

**Web Resources:**
1. [https://www.w3schools.com/java/](https://www.w3schools.com/java/)
2. [https://www.tutorialspoint.com/java/index.htm](https://www.tutorialspoint.com/java/index.htm)
3. [https://www.javatpoint.com/java-tutorial](https://www.javatpoint.com/java-tutorial)

**E-Books:**
1. java 2 complete reference 9th ed Herbert Schildt.pdf

**CO-PO/PSO Mapping:**

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
<th>Program Specific Outcomes*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PO 1</td>
<td>PO 2</td>
</tr>
<tr>
<td>CO1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>CO2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>CO3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>CO4</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
Database Management System lab

Prerequisite: Nil

Course Objectives:

1. Introduce ER data model, database design and normalization.
2. Learn SQL basics for data definition and data manipulation.

Course Outcomes:

1. Design database schema for a given application and apply normalization.
2. Perform operations on the database using SQL commands for data definition and data manipulation.
3. Develop solutions for database applications using procedures, cursors and triggers.
4. Implement joins on the tables.

List of Experiments:

The following programs are to be developed using My SQL / SQL / PL/SQL

Roadway Travels: "Roadway Travels" is in business since 1997 with several buses connecting different places in India. Its main office is located in Hyderabad.

The company wants to computerize its operations in the following areas:
- Reservations and Ticketing
- Cancellations

Reservations & Cancellation: Reservations are directly handled by booking office. Reservations can be made 30 days in advance and tickets issued to passenger. One Passenger/person can book many tickets (to his/her family). Cancellations are also directly handed at the booking office.

In the process of computerization of Roadway Travels you have to design and develop a Database which consists the data of Buses, Passengers, Tickets, and Reservation and cancellation details. You should also develop query’s using SQL to retrieve the data from the database.

The above process involves many steps like 1. Analyzing the problem and identifying the Entities and Relationships, 2. E-R Model 3. Relational Model 4. Normalization 5. Creating the database 6. Querying. Students are supposed to work on these steps week wise and finally create a complete “Database System” to Roadway Travels. Examples are given at every experiment for guidance to students.
**Experiment 1: E-R Model**
Analyze the problem carefully and come up with the entities in it. Identify what data has to be persisted in the database. This contains the entities, attributes etc.

Identify the primary keys for all the entities. Identify the other keys like candidate keys, partial keys, if any.

**Example:**
**Entities:** 1. BUS 2. Ticket 3. Passenger
**Relationships:** 1. Reservation 2. Cancellation
**Primary Key Attributes:**
1. Ticket ID (Ticket Entity)
2. Passport ID (Passenger Entity)
3. Bus_NO (Bus Entity)

Apart from the above mentioned entities you can identify more. The above mentioned are few.

**Note:** The student is required to submit a document by writing the Entities and Keys to the lab teacher.

**Experiment 2: Concept design with E-R Model**
Relate the entities appropriately. Apply cardinalities for each relationship. Identify strong entities and weak entities (if any). Indicate the type of relationships (total / partial). Try to incorporate generalization, aggregation, specialization etc wherever required.

**Example: E-R diagram for bus**

![E-R Diagram for Bus](image)

**Note:** The student is required to submit a document by drawing the E-R Diagram to the lab teacher.
Experiment 3: Relational Model
Represent all the entities (Strong, Weak) in tabular fashion. Represent relationships in a tabular fashion. There are different ways of representing relationships as tables based on the cardinality. Represent attributes as columns in tables or as tables based on the requirement.

Different types of attributes (Composite, Multi-valued, and Derived) have different way of representation.

Example: The passenger tables look as below. This is an example. You can add more attributes based on your E-R model. This is not a normalized table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Sex</th>
<th>Address</th>
<th>Ticket_id</th>
<th>Passport ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: The student is required to submit a document by Represent relationships in a tabular fashion to the lab teacher.

Experiment 4: Normalization
Database normalization is a technique for designing relational database tables to minimize duplication of information and, in so doing, to safeguard the database against certain types of logical or structural problems, namely data anomalies. For example, when multiple instances of a given piece of information occur in a table, the possibility exists that these instances will not be kept consistent when the data within the table is updated, leading to a loss of data integrity. A table that is sufficiently normalized is less vulnerable to problems of this kind, because its structure reflects the basic assumptions for when multiple instances of the same information should be represented by a single instance only.

For the above table in the First normalization we can remove the multi valued attribute Ticket_id and place it in another table along with the primary key of passenger.

First Normal Form: The above table can be divided into two tables as shown below.

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Sex</th>
<th>Address</th>
<th>Passport ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Passport ID</th>
<th>Ticket_id</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

You can do the second and third normal forms if required. Anyhow Normalized tables are given at the end.
Experiment 5: Installation of Mysql / SQL and practicing DDL commands

Installation of Mysql / SQL. In this session you will learn Creating databases, How to create tables, altering the database, dropping tables and databases if not required. You will also try truncate, rename commands etc.

Example for creation of a normalized “Passenger” table.

CREATE TABLE Passenger (  
    Passport_id  INTEGER PRIMARY KEY,  
    Name VARCHAR (50) Not NULL,  
    Age   Integer Not NULL,  
    Sex Char,  
    Address VARCHAR (50) Not NULL);

Similarly create all other tables.

Note: Detailed creation of tables is given at the end.

Experiment 6: Practicing DML commands

DML commands are used to for managing data within schema objects. Some examples:

- SELECT - retrieve data from the a database
- INSERT - insert data into a table
- UPDATE - updates existing data within a table
- DELETE - deletes all records from a table, the space for the records remain

Inserting values into “Bus” table:

Insert into Bus values (1234,'hyderabad', 'tirupathi');
Insert into Bus values (2345,'hyderabd','Banglore');
Insert into Bus values (23,'hyderabd','Kolkata');
Insert into Bus values (45,'Tirupathi','Banglore');
Insert into Bus values (34,'hyderabd','Chennai');

Inserting values into “Passenger” table:

Insert into Passenger values (1, 45,'ramesh', 45,'M’,’abc123’);
Insert into Passenger values (2, 78,'geetha', 36,'F’,’abc124’);
Insert into Passenger values (45, 90,’ram’, 30,’M’,’abc12’);
Insert into Passenger values (67, 89,’ravi’, 50,’M’,’abc14’);
Insert into Passenger values (56, 22,’seetha’, 32,’F’,’abc55’);

Few more Examples of DML commands:

Select * from Bus; (selects all the attributes and display)
UPDATE BUS SET Bus No = 1 WHERE BUS NO=2;
Experiment 7: Querying
In this week you are going to practice queries (along with sub queries) using ANY, ALL, IN, Exists, NOT EXISTS, UNION, INTERSECT, Constraints etc.

Practice the following Queries:
1. Display unique PNR_no of all passengers.
2. Display all the names of male passengers.
3. Display the ticket numbers and names of all the passengers.
4. Find the ticket numbers of the passengers whose name start with ‘r’ and ends with ‘h’.
5. Find the names of passengers whose age is between 30 and 45.
6. Display the sorted list of passengers names

Experiment 8 and Experiment 9: Querying (continued…)
You are going to practice queries using Aggregate functions (COUNT, SUM, AVG, and MAX and MIN), GROUP BY, HAVING and Creation and dropping of Views.
1. Write a Query to display the Information present in the Passenger and cancellation tables. **Hint:** Use UNION Operator.
2. Display the number of days in a week on which the 9W01 bus is available.
3. Find number of tickets booked for each PNR_no using GROUP BY CLAUSE. **Hint:** Use GROUP BY on PNR_No.
4. Find the distinct PNR numbers that are present.
5. Find the number of tickets booked by a passenger where the number of seats is greater than 1. **Hint:** Use GROUP BY, WHERE and HAVING CLAUSES.
6. Find the total number of cancelled seats.

Experiment 10: Triggers
In this session you are going to work on Triggers. Creation of insert trigger, delete trigger, update trigger. Practice triggers using the above database.

**Eg:**
```
CREATE OR REPLACE TRIGGER SUN_TRI
AFTER INSERT OR UPDATE OR DELETE ON EMP
DECLARE
  DY VARCHAR2 (200);
BEGIN
  DY:=TO_CHAR(SYSDATE,'DY');
  IF DY='SUN' THEN
    RAISE_APPLICATION_ERROR (-20005,'TODAY IS SUNDAY TRANSACTION NOT ALLOWED TODAY');
  END IF;
END;
```

Experiment 11: Procedures
In this session you are going to learn Creation of stored procedure, Execution of procedure and modification of procedure. Practice procedures using the above database.

**Eg:**
```
CREATE PROCEDURE myProc()
BEGIN
  SELECT COUNT (Tickets) FROM Ticket WHERE age>=40;
END;
```
Experiment 12: Cursors

In this session you need to do the following: Declare a cursor that defines a result set.
Open the cursor to establish the result set. Fetch the data into local variables as needed from the
cursor, one row at a time. Close the cursor when done

Eg:

DECLARE
CURSOR A IS
SELECT * FROM EMP;
B A%ROWTYPE;
BEGIN
OPEN A;
LOOP
FETCH A INTO B;
EXIT WHEN A%NOTFOUND;
DBMS_OUTPUT.PUT_LINE('EMP NO IS ' || B.EMPNO);
DBMS_OUTPUT.PUT_LINE('ENAME IS ' || B.ENAME);
DBMS_OUTPUT.PUT_LINE('SAL IS ' || B.SAL);
DBMS_OUTPUT.PUT_LINE('MGR NO IS ' || B.MGR);
DBMS_OUTPUT.PUT_LINE('COMM IS ' || B.COMM);
DBMS_OUTPUT.PUT_LINE('HIREDATE IS ' || B.HIREDATE);
DBMS_OUTPUT.PUT_LINE('DEPTNO IS ' || B.DEPTNO);
DBMS_OUTPUT.PUT_LINE('EMP JOB IS ' || BJOB);
DBMS_OUTPUT.PUT_LINE('*************************');
END LOOP;
CLOSE A;
END;

Text Books:
   3rd Edition

Reference Books:
1. Database Systems design, Implementation, and Management, Peter Rob & Carlos Coronel
   7th Edition.
2. Fundamentals of Database Systems, Elmasri Navrate, Pearson Education
3. Introduction to Database Systems, C. J. Date, Pearson Education

CO-PO/PSO Mapping Chart

<table>
<thead>
<tr>
<th>COs</th>
<th>PO 1</th>
<th>PO 2</th>
<th>PO 3</th>
<th>PO 4</th>
<th>PO 5</th>
<th>PO 6</th>
<th>PO 7</th>
<th>PO 8</th>
<th>PO 9</th>
<th>PO 10</th>
<th>PO 11</th>
<th>PO 12</th>
<th>PSO 1</th>
<th>PSO 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>CO2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>CO3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>CO4</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
</tbody>
</table>
DATA STRUCTURES LAB

Pre-requisite: Programming in C and Problem Solving

Course Objectives:

1. Understand various data representation techniques in the real world.
2. Implement linear and non-linear data structures.
3. Analyze various algorithms based on their time and space complexity.
4. Develop real-time applications using suitable data structure.
5. Identify suitable data structure to solve various computing problems.

Course Outcomes:

1. Identify the appropriate data structures and algorithms for solving real world problems
2. Implement various kinds of searching and sorting techniques.
3. Implement data structures such as stacks, queues, search trees, and hash tables to solve various computing problems
4. Choose appropriate traversal method to find shortest distance

List of Experiments:

1. Develop a program to read a linear list of items and store it in an array.
   a. Copy the contents from one array to another array
   b. Copy the contents from one array to another array in reverse order
   c. Delete the duplicate elements from an array.

2. Develop a program to implement various sorting techniques:
   a. Insertion sort
   b. Selection Sort
   c. Bubble Sort
   d. Merge Sort
   e. Quick Sort

3. a. Develop a program to create a stack and perform various operations on it.
   b. Develop a program to create a queue and perform various operations on it.

4. Develop programs for the following:
   a. Uses Stack operations to convert infix expression into postfix expression.
   b. Uses Stack operations for evaluating the postfix expression.

5. Design a program to create a singly linked list for the following operations
   a. Insert a Node at Beginning, at Ending and at a given Position
   b. Delete a Node at Beginning, at Ending and at a given Position
   c. Search, Count the Number of Nodes and Display
6. Design a program to create a doubly linked list for the following operations
   a. Insert a Node at Beginning, at Ending and at a given Position
   b. Delete a Node at Beginning, at Ending and at a given Position
   c. Search, Count the Number of Nodes and Display

7. Design a program to create a binary tree and perform various traversals.
   a. In order
   b. Pre Order
   c. Post Order

8. Develop a program to Perform Linear Search and Binary Search on a list stored in an array.

9. Design a program to create a Hash table for the following operations
   a. Insert
   b. Delete
   c. Search

10. Develop programs to implement the following graph traversal algorithms:
    a. Depth first search.
    b. Breadth first search.

Text/Reference Books:
2. Data structures using C - A.S. Tanenbaum, Y.L. Angsam, and MJ Augustien, PHI Pearson Education

CO-PO/PSO Mapping Chart:

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
<th>Program Specific Outcomes*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P O 1 P O 2 P O 3 P O 4 P O 5 P O 6 P O 7 P O 8 P O 9 P O 10 P O 11 P O 12 PS O 1 PS O 2 PSO3</td>
<td></td>
</tr>
<tr>
<td>CO1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO2</td>
<td>H</td>
<td>M</td>
</tr>
<tr>
<td>CO3</td>
<td></td>
<td>M</td>
</tr>
<tr>
<td>CO4</td>
<td>H</td>
<td></td>
</tr>
</tbody>
</table>
Problem Solving using C Lab

(Common to CSE/CSM/CDS/CSC/CSO/ECE/EEE/IOT)

Pre-requisite: Nil

Course Objectives:

1. To learn the fundamentals of computers.
2. To understand the various steps in program development.
3. To learn the syntax and semantics of C programming language.
4. To learn the usage of structured programming approach in solving problems.

Week – 1: OPERATORS AND EVALUATION OF EXPRESSIONS

a. Design and develop a flowchart and algorithm to read a number and implement using a C program to check whether the given number is even or odd using ternary operator.
b. Design and develop a flowchart and algorithm to read two integers and implement using a C program to perform the addition of two numbers without using + operator.
c. Develop a C program to evaluate the following arithmetic expressions by reading appropriate input from the standard input device. Understand the priority of operators while evaluating expressions.
   i. $6*2/(2+1*2/3+6)+8*(8/4)$
   ii. $17-8/4*2+3-++2$
   iii. $!(x>10)\&\&(y==2)$
d. Develop a C program to display the size of various built-in data types in C language.

Week – 2: CONTROL STRUCTURES

a. Design and develop a flowchart and algorithm to read a year as an input and find whether it is leap year or not. Implement a C program for the same and execute for all possible inputs with appropriate messages. Also consider end of the centuries.
b. Design and develop a flowchart and algorithm to find the square root of a given number $N$. Implement a C program for the same and execute for all possible inputs with appropriate messages. (Note: Don’t use library function sqrt(n), Hint: Use Newton-Raphson method to find the square root).
c. Design and develop a flowchart and algorithm to generate a Fibonacci sequence up to a given number $N$. A Fibonacci sequence is defined as follows: The first and second terms in the sequence are 0 and 1. Subsequent terms are found by adding the preceding two terms in the sequence. Implement a C program for the developed flowchart/algorithm and execute the same to generate the first $N$ terms of the sequence.
d. Design and develop a flowchart and algorithm that takes three coefficients (a, b, and c) of a Quadratic equation ($ax^2+bx+c=0$) as input and compute all possible roots. Implement a C program for the developed flowchart/algorithm and execute the same to output the possible roots for a given set of coefficients with appropriate messages.
Week – 3: CONTROL STRUCTURES

a. Design and develop an algorithm to find the reverse of an integer number N and check whether it is PALINDROME or NOT. Implement a C program for the developed algorithm that takes an integer number as input and output the reverse of the same with suitable messages. Ex: N: 2020, Reverse: 0202, Not a Palindrome.
b. Draw the flowchart and write C Program to compute sin(x) using Taylor series approximation given by
   \[
   \sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldotsd
   \]
   Compare the result with the built-in Library function and print both the results with appropriate messages.
c. Design and develop an algorithm and flowchart to read a three digit number and check whether the given number is Armstrong number or not. Write a C program to implement the same and also display the Armstrong numbers between the ranges 1 to 1000.
d. Design and develop an algorithm for evaluating the polynomial \( f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0 \), for a given value of x and its coefficients using Horner’s method. Implement a C program for the same and execute the program for different sets of values of coefficients and x.

Week – 4: ARRAYS

a. Develop, implement and execute a C program to read a list of integers and store it in a single dimensional array. Write a C program to print the second largest integer in a list of integers.
b. Develop, implement and execute a C program to read a list of integers and store it in a single dimensional array. Write a C program to count and display positive, negative, odd and even numbers in an array.
c. Develop, implement and execute a C program to read a list of integers and store it in a single dimensional array. Write a C program to find the frequency of a particular number in a list of integers.
d. Develop, implement and execute a C program that reads two matrices A (m x n) and B (p x q) and Compute the product A and B. Read matrix A and matrix B in row major order respectively. Print both the input matrices and resultant matrix with suitable headings and output should be in matrix format only. Program must check the compatibility of orders of the matrices for multiplication. Report appropriate message in case of incompatibility.

Week – 5: STRINGS

a. Develop a user-defined function STRCOPY (str1, str2) to simulate the built-in library function strcpy (str1, str2) that copies a string str2 to another string str1. Write a C program that invokes this function to perform string copying. Also perform the same operation using built-in function.
b. Develop a user-defined function STRCONCT (str1, str2) to simulate the built-in library function strcat (str1, str2) that takes two arguments str1 and str2, concatenates str2 and str1 and stores the result in str1. Write a C program that invokes this function to perform string concatenation. Also perform the same operation using built-in function.
c. Develop a C program that returns a pointer to the first occurrence of the string in a
given string using built-in library function `strstr()`. Example: `strstr()` function is used to locate first occurrence of the string “test” in the string “This is a test string for testing”. Pointer is returned at first occurrence of the string “test”.

d. Develop a C program using the library function `strcmp (str1, str2)` that compares the string pointed to by `str1` to the string pointed to by `str2` and returns an integer. Display appropriate messages based on the return values of this function as follows –

```c
if return value < 0 then it indicates str1 is less than str2.
if return value > 0 then it indicates str2 is less than str1.
if return value = 0 then it indicates str1 is equal to str2.
```

Week – 6: FUNCTIONS

a. Design and develop a recursive and non-recursive function `FACT (num)` to find the factorial of a number, n!, defined by `FACT(n) = 1`, if n = 0. Otherwise `FACT (n) = n * FACT(n-1)`. Using this function, write a C program to compute the binomial coefficient. Tabulate the results for different values of n and r with suitable messages.
b. Design and develop a recursive function `GCD (num1, num2)` that accepts two integer arguments. Write a C program that invokes this function to find the greatest common divisor of two given integers.
c. Design and develop a recursive function `FIBO (num)` that accepts an integer argument. Write a C program that invokes this function to generate the Fibonacci sequence up to `num`.
d. Design and develop a C function `ISPRIME (num)` that accepts an integer argument and returns 1 if the argument is prime, a 0 otherwise. Write a C program that invokes this function to generate prime numbers between the given ranges.
e. Design and develop a function `REVERSE (str)` that accepts a string arguments. Write a C program that invokes this function to find the reverse of a given string.

Week – 7: POINTERS

a. Develop a C program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers.
b. Develop a C program to read a list of integers and store it in an array. Then read the array elements using a pointer and print the value along with the memory addresses.
c. Design and develop non-recursive functions `input_matrix (matrix, rows, cols)` and `print_matrix(matrix, rows, cols)` that stores integers into a two-dimensional array and displays the integers in matrix form. Write a C program to input and print elements of a two dimensional array using pointers and functions.
d. Develop a C program to a store a list of integers in a single dimensional array using dynamic memory allocation (limit will be at run time) using `malloc()` function. Write a C program to read the elements and print the sum of all elements along with the entered elements. Also use `free()` function to release the memory.
Week – 8: STRUCTURES AND UNIONS

a. Write a C program that uses functions to perform the following operations:
   i. Reading a complex number
   ii. Writing a complex number
   iii. Addition and subtraction of two complex numbers
       Note: represent complex number using a structure.

b. Write a C program to compute the monthly pay of 100 employees using each
   employee's name, basic pay. The DA is computed as 52% of the basic pay. Gross
   salary (basic pay + DA). Print the employees name and gross salary.

c. Create a Book structure containing book_id, title, author name and price. Write a C
   program to pass a structure as a function argument and print the book details.

d. Create a union containing 6 strings: name, home_address, hostel_address, city, state and
   zip. Write a C program to display your present address.

Week – 9: ADDITIONAL PROGRAMS

a. Write a C program to read in two numbers, x and n, and then compute the sum of this
   geometric progression: \(1+x+x^2+x^3+\ldots +x^n\). For example: if n is 3 and x is 5, then
   the program computes 1+5+25+125. Print x, n, the sum. Perform error checking. For
   example, the formula does not make sense for negative exponents – if n is less than 0.
   Have your program print an error message if n<0, then go back and read in the next pair
   of numbers of without computing the sum. Are any values of x also illegal? If so, test
   for them too.

b. Develop a C program to find the 2’s complement of a given binary number. 2’s
   complement is obtained by scanning it from right to left and complementing all the bits
   after the first appearance of a 1. Thus 2’s complement of 11100 is 00100. Write a C
   program to find the 2’s complement of a binary number.

c. Develop a C program to convert a Roman numeral to its decimal equivalent. E.g. check
   for the inputs - Roman number IX is equivalent to 9 and Roman number XI is equivalent
   to 11.

Week – 10: PREPROCESSOR DIRECTIVES

a. Define a macro with one parameter to compute the volume of a sphere. Write a C
   program using this macro to compute the volume for spheres of radius 5, 10 and
   15meters.

b. Define a macro that receives an array and the number of elements in the array as
   arguments. Write a C program for using this macro to print the elements of the array.

c. Write symbolic constants for the binary arithmetic operators +, -, *, and /. Write a C
   program to illustrate the use of these symbolic constants.

Week – 11: FILES

a. Create an employee file employee.txt and write 5 records having employee name,
   designation, salary, branch and city. Develop a C program to display the contents of
   employee.txt file.

b. Create a studentolddata.txt file containing student name, roll no, branch, section,
address. Develop a C program to copy the contents of `studentolddata.txt` file to another file `studentnewdata.txt`.

c. Develop a C program to create a text file `info.txt` to store the information given below. Implement using a C program to count the number of words and characters in the file `info.txt`.

**Test Data:**

Input the file name to be opened: info.txt

**Expected Output:**

The content of the file info.txt are:

Welcome to IARE

Welcome to Computer Programming

The number of words in the file info.txt are: 7

The number of characters in the file info.txt are: 46

d. Given two university information files “`studentname.txt`” and “`roll_number.txt`” that contains students Name and Roll numbers respectively. Write a C program to create a new file called “`output.txt`” and copy the content of files “`studentname.txt`” and “`roll_number.txt`” into output file. Display the contents of output file “`output.txt`” on to the screen.

<table>
<thead>
<tr>
<th>studname.txt</th>
<th>roll_number.txt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asha</td>
<td>20951A1201</td>
</tr>
<tr>
<td>Bharath</td>
<td>20951A0502</td>
</tr>
<tr>
<td>Uma</td>
<td>20951A0456</td>
</tr>
<tr>
<td>Shilpa</td>
<td>20951A0305</td>
</tr>
</tbody>
</table>

**Week – 12: COMMAND LINE ARGUMENTS**

a. Develop a C program to read a set of arguments and display all arguments given through command line.

b. Develop a C program to read a file at command line argument and display the contents of the file.

c. Develop a C program to read N integers and find the sum of N integer numbers using command line arguments.

d. Develop a C program to read three integers and find the largest integer among three using command line argument.
Text Books:

REFERENCE BOOKS:


Course Outcomes:

The candidate is expected to be able to

1. CO1 - Formulate the algorithms for simple problems, translate given algorithms to a working and correct program, correct syntax errors as reported by the compilers.
2. CO2 - Identify and correct logical errors encountered during execution, represent and manipulate data with arrays, strings and structures
3. CO3 - use pointers of different types, create, read and write to and from simple text and binary files
4. CO4 - modularize the code with functions so that they can be reused

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
<th>Program Specific Outcomes*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PO 1</td>
<td>PO 2</td>
</tr>
<tr>
<td>CO1</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>CO2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>CO3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
English For Employability Lab
(Common to CSE/EEE/ECE/CSM/CSD/CSC/CSO/MECH)

Prerequisites:
1. Basic knowledge of interpersonal and communication skills.
2. Enriching the speaking and writing ability
3. Basic grammar rules of (LSRW)
4. Basic Spoken English skills

Course Objectives:
1. To improve the communication skills, body language, facial expression and gesture.
2. To be able to understand the concept of employability skills (Quantum dexterity) and enhancing ones’ behavior in the personal, professional and social forum.
3. To evaluate the LSRW (listening, speaking, reading and writing) through assessment.
4. To learn the basic grammar for improving spoken and written communication.
5. To become problem solver, analyze and apply critical and analytical skills.
6. To identify the Employability skills, assigning tasks (Group Discussion, JAM, Role play etc.,) for day today evaluation.

Activity 1- Narration (Historical places, events, Picture narration, Memorable incidents of life)
Self-Introduction, Daily Routine, Likes & Dislikes, Vocabulary, Triangular Activity (Person based- S-P), Imperatives & JAM
Targeted Skills- Listening- Speaking- Audio-Video clips

Activity 2: Quantum of Dexterity (QOD)-- Ability (Personal, Behavioural & Professional)
Request/Permission/Order, Survival kit, Career Objective Professional, Hidden Talents (Personal), Character Traits (Behavioural)
Targeted Skills- Reading-Writing – Concluding an open-ended Story, Creative Writing.

Activity 3: Critical & Analytical Skills
SWOC- (Social & Cultural, Political, Economic, Legal Impact, Technical, Nuances of Pronunciation, Voice Modulation, Neutralizing Mother Tongue Interference, Tongue Twisters for practice,
**Targeted Skills** - Writing SWOC, Self-Introduction, Exposure to a structured talk.

Activity 4: **Flick Flow/Exttempore**

**Targeted Skills** - Speaking Skills

Activity 5: **On Job Training**
Formal & Informal communication, Resume E-mail Etiquette, Telephonic & Interview Etiquette, Situation based- Santa’s Bag, topic/case study-based Group Discussion (Job Consultancy/Role Play)

**Targeted Skills** - Listening-Writing- Speaking

**Course Outcomes:** After undergoing this course, the student will be able to;

- **CO1:** Understand the explicit and implicit of importance of employability skills.
- **CO2:** Demonstrate life skills like team work, learning skills, problem solving, attitude, adaptability and flexibility.
- **CO3:** Apply critical and analytical skills to bring out the solution on problem/case study.
- **CO4:** Recognize the need of appropriate words, Phrases & functional grammar and apply them in both spoken and written communication.

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
<th>PO1</th>
<th>PO2</th>
<th>PO3</th>
<th>PO4</th>
<th>PO5</th>
<th>PO6</th>
<th>PO7</th>
<th>PO8</th>
<th>PO9</th>
<th>PO10</th>
<th>PO11</th>
<th>PO12</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>CO2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2 2</td>
</tr>
<tr>
<td>CO3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3 3</td>
</tr>
<tr>
<td>CO4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3 3</td>
</tr>
</tbody>
</table>
ENVIRONMENTAL SCIENCE
(Common to CSE/EEE/ECE/MECH/CSM/CDS/CSC/CSO)

Course Objectives:
1. Understanding the importance of ecological balance for sustainable development.
2. To educate students about natural resources and their exploitation
3. Understanding the concepts of green chemistry and its applications.

Module I ECOSYSTEMS AND ECOLOGY

UNIT-1: Ecosystem
Definition, Scope, and Importance of ecosystem. Classification, structure, and function of an ecosystem, Food chains, food webs, and ecological pyramids.

UNIT-2: Ecology
Flow of energy, Biogeochemical cycles, Bioaccumulation, Biomagnification, ecosystem value, services and carrying capacity, Field visits.

Module II NATURAL RESOURCES

UNIT-1: Classification of Resources
Living and Non-Living resources, water resources: use and over utilization of surface and ground water, floods and droughts, Dams: benefits and problems. Mineral resources: use and exploitation, environmental effects of extracting and using mineral resources,

UNIT-2: Land and Energy resources
Land resources: Forest resources, Energy resources: growing energy needs, renewable and non-renewable energy sources, use of alternate energy source, case studies.

Module III BIODIVERSITY AND BIOTIC RESOURCES

UNIT-1: Biodiversity
Introduction, Definition, genetic, species and ecosystem diversity. Value of biodiversity; consumptive use, productive use, social, ethical, aesthetic and optional values. India as a mega diversity nation, Hot spots of biodiversity. Field visit

UNIT-2: Biotic Resources
Module IV ENVIRONMENTAL POLLUTION AND SOLID WASTE

UNIT-1: Environmental pollution

Classification of pollution, Air Pollution: Primary and secondary pollutants, Automobile and Industrial pollution, Ambient air quality standards. Water pollution: Sources and types of pollution, drinking water quality standards. Soil Pollution: Sources and types, Impacts of modern agriculture, degradation of soil. Municipal Solid Waste management, composition and characteristics of e-Waste and its management

UNIT-2: Global Environmental Issues and Global Efforts


Module V Green Chemistry & Hazardous chemicals

UNIT-1: Green Chemistry

Introduction & Principles, green solutions for chemical energy storage, green chemistry solutions will be discussed within the fields of Chemical production: choice of feedstock, solvents, catalysts, synthesis routes including microwave and ultrasonic assisted synthesis.

UNIT -2: Hazardous Chemicals

Classification of hazardous chemicals, transportation of hazardous chemicals, Hazchem code, Storage and handling of hazardous substances, Emergency preparedness (on site & offsite), Safety audit, Concept of fire and explosion, Major accidents involving hazardous substances

Text Books:

1. Textbook of Environmental Studies for Undergraduate Courses by Erach Bharucha for University Grants Commission.
2. Environmental Studies by R. Rajagopalan, Oxford University Press.

Course Outcomes:

CO-1: Understand the importance of ecosystem and ecological balance in conservation of biodiversity.

CO-2: Understand the concepts of natural resources and its exploitation.

CO-3: Explain the control of pollution for sustainable environment.

CO-4: Explain the concepts of green chemistry and its applications.
## CO-PO/PSO Mapping Chart
(3/2/1 indicates strength of correlation)
3 – High; 2 – Medium; 1 - Low

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
<th>Program Specific Outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PO1</td>
<td>PO2</td>
</tr>
<tr>
<td>CO1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>CO2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>CO3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>CO4</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
Pre-requisite: Basic knowledge of matrix theory.

Course Objectives: To provide the student with

- Vector spaces.
- Eigen values and Eigen vectors.
- Solution of linear system by Jacobi, Gauss-Seidel, LU decomposition Gaussian elimination and Least square methods.
- Householder transformation, QR factorization and Sensitivity analysis.
- Singular Value Decomposition.
- Moore-Penrose inverse.
- Reduction to Heisenberg and tridiagonal forms and bi diagonal form, Power, inverse power and Rayleigh quotient iterations.
- Sensitivity analysis of singular values and singular vectors, conjugate gradient method.

MODULE I
VECTOR SPACES
Unit 1:
Introduction, Vector spaces, sub-spaces, spanning sets, basis and dimension,
Unit 2:
Linear transformations, matrix of linear transformations, eigen values and eigen vectors.

MODULE II
LINEAR SYSTEM
Unit 1:
Matrix norm, Sensitivity analysis and condition numbers, Linear systems, Jacobi, Gauss-Seidel and successive over relaxation methods.
Unit 2:
LU decompositions, Gaussian elimination with partial pivoting, Banded systems, positive definite systems.
MODULE III

Unit 1:
Cholesky decomposition – sensitivity analysis, Gram-Schmidt orthonormal process, Householder transformation, QR factorization, stability of QR factorization.

Unit 2:
Solution of linear least squares problems, normal equations, singular value decomposition (SVD).

MODULE IV

Unit 1:

Unit 2:
Reduction to Heisenberg and tridiagonal forms; Power, inverse power and Rayleigh quotient iterations

MODULE V

Unit 1:
Explicit and implicit QR algorithms for symmetric and non-symmetric matrices.

Unit 2:
Reduction to bi diagonal form, Sensitivity analysis of singular values and singular vectors, conjugate gradient method.

Text Books:

Reference Books:

MOOC courses:
1. https://nptel.ac.in/courses/111/102/111102152/
E- Books:

1. Matrix Computations by Gene H. Golub, Charles F. Van Loan
   https://1lib.in/book/2190556/88c37f
2. Numerical linear algebra and applications by Biswa Nath Datta
   https://1lib.in/book/5259166/c11b06
   https://1lib.in/book/942630/262edc
4. Numerical Linear Algebra by L N Trefethen, David Bau III
   https://1lib.in/book/2464030/a9a674

Course Outcomes: After learning the contents of this paper the student must be able to

CO1: Determine Eigen values and Eigen vectors of a matrix which arise in several engineering applications.

CO2: Apply various Matrix Techniques to Industrial Data Analysis.

CO3: Determine pseudo inverse of a matrices by using Singular Value Decomposition.

CO4: Understand the QR factorization and Sensitivity analysis of linear system of equations.
B.Tech II Year–IV Sem
Subject Code: 21ES4CS03

PYTHOn PROGRAMMING
(Common To All Branches)

Pre-requisite: Programming for Problem solving Using C

Course Objectives:
1. Acquire programming skills in core Python.
2. Acquire Object-oriented programming skills in Python.
3. Develop the skill of designing graphical-user interfaces (GUI) in Python.
4. Develop the ability to write database applications in Python.

Course Outcomes:
1. Understand operators, precedence of operators, associatively while evaluating expressions in program statements.
2. Distinguish the capabilities of procedural as well as object-oriented programming in Python and demonstrate the same in real world scenario.
3. Demonstrate indexing and slicing mechanisms for extracting a portion of data in a sequence.
4. Extend the knowledge of Python programming to build successful career in software development.

Module I
Python Basics, Objects- Python Objects, Standard Types, Other Built-in Types, Internal Types, Standard Type Operators, Standard Type Built-in Functions, Categorizing the Standard Types, Unsupported Types
Numbers - Introduction to Numbers, Integers, Floating Point Real Numbers, Complex Numbers, Operators, Sequences - Strings, Lists, and Tuples, Mapping and Set Types

Module II
Conditional Statements: Selection/Conditional Branching Statements: if, if-else, nested if, if-elif-else statement(s).
Iterative Statements – while and for loop, Nested loops, break and continue statement, pass Statement, else Statement used with loops.

Module III
Functions: Communicating with functions, Variable Scope and lifetime, return statement, Types of arguments, Lambda functions, and Recursive functions
Classes and Objects – Defining Classes, Creating Objects, Data Abstraction and Hiding through Classes, Class Method and self-Argument, Class variables and Object variables, __init__() and __del__() method.

Module IV
Exceptions: Exceptions in Python, Detecting and Handling Exceptions, Context Management, , Raising Exceptions, Related Modules
Regular Expressions: Introduction, Special Symbols and Characters, Res and Python
Module V
GUI Programming: Introduction, Tkinter and Python Programming, Brief Tour of Other GUIs, Related Modules and Other GUIs
Text Books:

Reference Books:

CO-PO/PSO Mapping

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
<th>Program Specific Outcomes*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PO 1</td>
<td>PO 2</td>
</tr>
<tr>
<td>CO1</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>CO2</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>CO3</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>CO4</td>
<td>M</td>
<td></td>
</tr>
</tbody>
</table>

CO-PO/PSO Mapping Chart
(3/2/1 indicates strength of correlation)
3 – High; 2 – Medium; 1 - Low
DESIGN AND ANALYSIS OF ALGORITHMS

Prerequisites: Data Structure, Discrete Mathematics

Course Objectives

- To analyze the performance of algorithms.
- To choose the appropriate data structure & algorithm design method for specific application.
- To understand how the choice of data structure & algorithm design method impact the performance of program.
- To design efficient algorithms for different problems.

COURSE OUTCOMES:

At the end of the Course the Students will be able to

1. Describe computational solution to well-known problems like searching, sorting etc.
2. Estimate the computational complexity of different algorithms.
3. Apply different designing methods for development of algorithms to realistic problems through greedy, dynamic programming, back tracking.

Module I

Introduction: Algorithm, Pseudo code for expressing algorithms, Performance Analysis-Space complexity, Time complexity, Asymptotic Notation- Big oh notation, Omega notation, Theta notation and Little oh notation, Probabilistic analysis, Amortized Complexity.

Divide and conquer: General method, applications-Binary search, Quick sort, Merge sort, Strassen’s matrix multiplication

Module II

Searching and Traversal Techniques: Efficient non-recursive binary tree traversal algorithms, disjoint set operations, union and find algorithms, spanning trees.

Graph traversals: Breadth First Search and Depth First Search, AND/OR Graphs, game tree, connected components and biconnected components.

Module III

Backtracking: General method, applications-n-queen problem, sum of subsets problem, graph coloring, Hamiltonian cycles.

Greedy method: General method, applications-Job sequencing with deadlines, 0/1 knapsack problem.


Module IV

Dynamic Programming: General method, applications- 0/1 knapsack problem.

All pairs shortest path problem: Travelling sales person problem, Reliability design.
Module V
Branch and Bound:
General method, applications - Travelling sales person problem, 0/1 knapsack problem -
LC Branch and Bound solution, FIFO Branch and Bound solution.
NP-Hard and NP-Complete problems: Basic concepts, non deterministic algorithms, NP - Hard and NP Complete classes, NP-hard problems.

TEXT BOOKS

REFERENCE BOOKS

CO-PO/PSO Mapping:

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
<th>Program Specific Outcomes*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P O Q O Q O P O P O P O</td>
<td>P S O P S O P S O</td>
</tr>
<tr>
<td>CO1</td>
<td>3 2</td>
<td>3 3</td>
</tr>
<tr>
<td>CO2</td>
<td>1 2</td>
<td>3 3</td>
</tr>
<tr>
<td>CO3</td>
<td>3 2</td>
<td>3 3</td>
</tr>
<tr>
<td>CO4</td>
<td>3 2</td>
<td>3 3</td>
</tr>
</tbody>
</table>
OPERATING SYSTEMS

Course Objective
1. To understand the components, operations of the operating system
2. To interpret the scheduling policies and memory management issues
3. To understand the process concurrency and synchronization
4. To understand the concept of file management

Course Outcomes:
At the end of the course student will be able to
1. Understand the structure of Operating System and its architecture
2. Apply the scheduling strategies for real time implementations
3. Illustrate synchronization problems, deadlock and its techniques
4. Infer Security and Protection Mechanism

Module I
Operating System Introduction: Operating Systems objectives and functions, Computer System Architecture, OS Structure, OS Operations,

Module II

Module III
Memory Management and Virtual Memory - Logical & Physical Address Space, Swapping, Contiguous Allocation,

Module IV
Mass Storage Structure – Overview of Mass Storage Structure, Disk Structure, Disk Attachment, Disk Scheduling, Disk Management, Swap space Management
Module V

**Deadlocks** - System Model, Deadlock Characterization, Methods for Handling Deadlocks, Deadlock Prevention, Deadlock Avoidance, Deadlock Detection and Recovery from Deadlock.


**TEXT BOOKS:**


**REFERENCE BOOKS:**


**CO-PO &PSO Mapping:**

<table>
<thead>
<tr>
<th>Course Name - Course Outcomes / Program Outcomes</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>1 0</th>
<th>1 1</th>
<th>1 2</th>
<th>PSO 1</th>
<th>PSO 2</th>
<th>PSO 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO1</td>
<td>M</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO2</td>
<td>H</td>
<td>H</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO3</td>
<td>M</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO4</td>
<td>M</td>
<td>L</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
COMPUTER NETWORKS

Pre requisites:
1. A course on “Programming for problem solving”
2. A course on “Data Structures”

Course Objective
1. The objective of the course is to equip the students with a general overview of the concepts and fundamentals of computer networks.
2. Familiarize the students with the standard models for the layered approach to communication between machines in a network and the protocols of the various layers

Course Outcomes:
At the end of the course student will be able to
1. Gain the knowledge of the basic computer network technology.
2. Gain the knowledge of the functions of each layer in the OSI and TCP/IP reference model.
3. Obtain the skills of subnetting and routing mechanisms.
4. Familiarity with the essential protocols of computer networks, and how they can be applied in network design and implementation.

Module I
Network hardware, Network software, OSI, TCP/IP Reference models, Example Networks: ARPANET, Internet.

Module II
Data link layer: Design issues, framing, Error detection and correction. Elementary data link protocols: simplex protocol, A simplex stop and wait protocol for an error-free channel, A simplex stop and wait protocol for noisy channel.
Sliding Window protocols: A one-bit sliding window protocol, A protocol using Go-Back-N, A protocol using Selective Repeat

Module III
Multiple access protocols: ALOHA, Carrier sense multiple access protocols, collision free protocols. Wireless LANs, Data link layer switching.
Network Layer: Design issues, Routing algorithms: shortest path routing, Flooding, Hierarchical routing, Broadcast, Multicast, distance vector routing, Congestion Control Algorithms

Module IV
Transport Layer: Transport Services, Elements of Transport protocols, Connection management, TCP and UDP protocols.
Application Layer –Domain name system, SNMP, Electronic Mail; the World WEB, HTTP
Module V


TEXT BOOKS:


REFERENCE BOOKS:


CO-PO & PSO Mapping:

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
<th>Program Specific Outcomes*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PO 1       PO 2   PO 3   PO 4   PO 5   PO 6   PO 7   PO 8   PO 9   PO 10  PO 11  PO 12  PSO 1  PSO 2</td>
<td></td>
</tr>
<tr>
<td>CO1</td>
<td>2          1</td>
<td></td>
</tr>
<tr>
<td>CO2</td>
<td>3          2</td>
<td></td>
</tr>
<tr>
<td>CO3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CO4</td>
<td>2          1</td>
<td>1</td>
</tr>
</tbody>
</table>
OPERATING SYSTEM LAB

Course Objective
1. To do the implementation of operating system concepts
2. To write the code to Implement and modify various concepts in operating systems

Course Outcomes:
1. Implement Inter process communication between two processes
2. Design and solve Synchronization problems
3. Implement scheduling, deadlock and file management
4. Simulate and implement operating system concepts such as memory management

List of Programs:
1. Simulate the following CPU scheduling algorithms
   a) Round Robin b) SJF c) FCFS d) Priority
2. Simulate the following CPU scheduling algorithms
   a) FCFS b) Priority
3. Simulate all file allocation strategies
   a) Sequential b) Indexed c) Linked
4. Simulate MVT and MFT
5. Simulate all File Organization Techniques
   a) Single level directory b) Two level
6. Simulate all File Organization Techniques
   a) Hierarchical b) DAG
7. Simulate Bankers Algorithm for Dead Lock Avoidance
8. Simulate Bankers Algorithm for Dead Lock Prevention
9. Simulate all page replacement algorithms
   a) FIFO b) LRU c) LFU Etc.
10. Simulate Paging Technique of memory management.

Text Books:

CO-PO & PSO Mapping

<table>
<thead>
<tr>
<th>CO-PO</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>PSO 1</th>
<th>PSO 2</th>
<th>PSO 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO-1</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO-2</td>
<td>H</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO-3</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO-4</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
DESIGN AND ANALYSIS OF ALGORITHMS – LAB

Course Objectives:
1. Design and implement various algorithms in C / JAVA / Python
2. Employ various design strategies for problem solving.
3. Measure and compare the performance of different algorithms

Course Outcomes: The students should be able to:
1. Design algorithms using appropriate design techniques (brute-force, greedy, dynamic programming, etc.)
2. Develop variety of algorithms such as sorting, graph related, combinatorial, etc., in a high level language.
3. Analyze and compare the performance of algorithms using language features.
4. Apply and implement learned algorithm design techniques and data structures to solve real-world problems.

List of Programs:

1. Sort a given set of elements using the quick sort method and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.
2. Implement merge sort algorithm to sort a given set of elements and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.
3. Write a C program to implement the Stack using arrays. Write Push(), Pop(), and Display() methods to demonstrate its working.
4. Implement a C Program to implement the 0/1 Knapsack problem using (a) Dynamic Programming method (b) Greedy method.
5. Write a program, from a given vertex in a weighted connected graph, find shortest paths to other vertices using Dijkstra's algorithm.
6. Find Minimum Cost Spanning Tree of a given connected undirected graph using Kruskal's algorithm. Use Union-Find algorithms in your program.
7. Find Minimum Cost Spanning Tree of a given connected undirected graph using Prim's algorithm.
8. Write C programs to (a) Implement All-Pairs Shortest Paths problem using Floyd's
algorithm.

(b) Implement **Travelling Sales Person problem** using Dynamic programming.

9. Design an algorithm and implement a program to find a **subset** of a given set \( S = \{S_1, S_2, \ldots, S_n\} \) of \( n \) positive integers whose SUM is equal to a given positive integer \( d \). For example, if \( S = \{1, 2, 5, 6, 8\} \) and \( d = 9 \), there are two solutions \( \{1,2,6\} \) and \( \{1,8\} \). Display a suitable message, if the given problem instance doesn't have a solution.

10. Design and implement in Java to find all **Hamiltonian Cycles** in a connected undirected Graph \( G \) of \( n \) vertices using backtracking principle.

---

### TEXT BOOKS


### CO-PO/PSO Mapping:

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
<th>Program Specific Outcomes*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P O 1</td>
<td>P O 2</td>
</tr>
<tr>
<td>CO1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>CO2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>CO3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>CO4</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
B.Tech II Year – IV Sem

Subject Code: 21ES4CS04

L T P C
0 0 3 1.5

PYTHON PROGRAMMING LAB
(Common To All Branches)

Prerequisite: Basic Programming Skills.

Course Objectives:

To learn the fundamentals of computers.

1. Acquire programming skills in core Python.
2. Acquire Object-oriented programming skills in Python.
3. Develop the skill of designing graphical-user interfaces (GUI) in Python.
4. Develop the ability to write database applications in Python.

Course Outcomes:

1. Acquire programming skills in core Python.
2. Acquire Object-oriented programming skills in Python.
3. Develop the skill of designing graphical-user interfaces (GUI) in Python.
4. Develop the ability to write database applications in Python.

List of Experiments:

Experiment – 1: OPERATORS
a. Read a list of numbers and write a program to check whether a particular element is present or not using membership operators.
b. Read your name and age and write a program to display the year in which you will turn 100 years old.

Experiment – 2: CONTROL STRUCTURES
a. Read your email id and write a program to display the no of vowels, consonants, digits and white spaces in it using if…elif…else statement.
b. Write a Program to find the sum of a Series 1/1! + 2/2! + 3/3! + 4/4! +……+ n/n!. (Input :n = 5, Output : 2.70833)

Experiment – 3: LIST
a. Read a list of numbers and print the numbers divisible by x but not by y (Assume x = 4 and y = 5).
b. Read a list of numbers and print the sum of odd integers and even integers from the list.(Ex: [23, 10, 15, 14, 63], odd numbers sum = 101, even numbers sum = 24)

Experiment – 4: TUPLE
a. Given a list of tuples. Write a program to find tuples which have all elements divisible by K from a list of tuples. test_list = [(6, 24, 12), (60, 12, 6), (12, 18, 21)], K = 6, Output : [(6, 24, 12), (60, 12, 6)]
b. Given a list of tuples. Write a program to filter all uppercase characters tuples from given list of tuples. (Input: test_list = [('GFG', 'IS', 'BEST'), ('GFg', 'AVERAGE'), ('GfG', ), ('Gfg', 'CS')], Output : [('GFG', 'IS', 'BEST')]).
Experiment – 5: SET
a. Write a program to generate and print a dictionary that contains a number (between 1 and n) in the form (x, x*x).
b. Write a program to perform union, intersection and difference using Set A and Set B.

Experiment – 6: DICTIONARY
a. Write a program to do the following operations:
   i. Create a empty dictionary with dict( ) method
   ii. Add elements one at a time
   iii. Update existing key’s value
   iv. Access an element using a key and also get( ) method
   v. Deleting a key value using del( ) method
b. Write a program to create a dictionary and apply the following methods:
   i. pop( ) method
   ii. pop item( ) method
   iii. clear( ) method

Experiment – 7: STRINGS
a. Given a string, write a program to check if the string is symmetrical and palindrome or not. A string is said to be symmetrical if both the halves of the string are the same and a string is said to be a palindrome string if one half of the string is the reverse of the other half or if a string appears same when read forward or backward.
b. Write a program to read a string and count the number of vowel letters and print all letters except ‘e’ and ‘s’.

Experiment – 8: USER DEFINED FUNCTIONS
a. Write a function merge_dict(dict1, dict2) to merge two Python dictionaries.
b. Given a list of n elements, write a linear_search() function to search a given element x in a list.

Experiment – 9: BUILT-IN FUNCTIONS
a. Write a program to demonstrate the working of built-in statistical functions mean ( ), mode ( ), median ( ) by importing statistics library.
b. Write a program to demonstrate the working of built-in trigonometric functions sin ( ), cos ( ), tan ( ), hypot ( ), degrees ( ), radians ( ) by importing math module.

Experiment – 10: CLASS AND OBJECTS
a. Write a program to create a Bank Account class. Your class should support the following methods for
   i) Deposit
   ii) Withdraw
   iii) Get Balance
   iv) Pin Change
b. Write a program to create an employee class and store the employee name, id, age, and salary using the constructor. Display the employee details by invoking employee_info() method and also using dictionary (__dict__).

Reference Books:

CO-PO/PSO Mapping Chart:

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
<th>Program Specific Outcomes*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PO 1</td>
<td>PO 2</td>
</tr>
<tr>
<td>CO1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>CO2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>CO3</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>CO4</td>
<td></td>
<td>2</td>
</tr>
</tbody>
</table>
Course Objective
1. To do the implementation of operating system concepts
2. To write the code to implement and modify various concepts in operating systems
3. To understand the working principle of various communication protocols.
4. To analyze the traffic flow and the contents of protocol frames

Course Outcomes:
1. Implement the functionalities of Data Link Layer protocols
2. Create the socket programming interface for client server programming
3. Design and develop efficient routing algorithms
4. Evaluate the performance of Network Protocols.

List of Programs:
1. Write a program for error detection using CRC-CCITT(16-bits)
2. Write a program to generate Hamming Code for error detection and correction.
3. Using TCP/IP Sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.
4. Write a program for Congestion control using the leaky bucket algorithm
5. Write a program for Distance Vector Algorithm to find suitable path for transmission.
6. Write a program for Link State Algorithm to find suitable path for transmission
7. a) Simulate Capturing and analyzing Ethernet frames.
   b) Simulate HTTP GET/POST interaction
8. Simulate a three nodes point-to-point network with duplex links between them. Set the queue size vary the bandwidth and find the number of packets dropped.
9. Simulate a four node point-to-point network, and connect the links as follows: n0-n2, n1-n2 and n2-n3. Apply TCP agent between n0-n3 and UDP agent between n1-n3. Apply relevant applications over TCP and UDP agents by changing the parameters and determine the number of packets sent by TCP/UDP.
10. Simulate
   a) Analysis of ICMP and PING messages
   b) Analysis of ICMP and Trace route
11. Simulate a wireless network, generate traffic and analyze its performance.
12. Simulate a transmission of ping message over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.

Text Books:

<table>
<thead>
<tr>
<th>Course Name - Course Outcomes / Program Outcomes</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>PS O 1</th>
<th>PS O 2</th>
<th>PS O 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CO4</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
DOING ENGINEERING-I

Pre-requisite: Basic Knowledge on Computers and C language

Course Objectives:
1) To Provide knowledge of different Smart System applications.
2) To provide knowledge of Arduino boards, Arduino as IDE, programming language, platform and basic components
3) Develop skills to design and implement various smart system application.
4) To know the basics of micro-controllers and sensors very quickly and can start building prototype.

Course Outcomes:
1) Able to Design Smart systems applications.
2) Understand about any new IDE, compiler, and MCU chip in Arduino compatible boards or similar types.
3) Understand Arduino programming with embedded C.
4) Learn how to prototype circuits with a breadboard

I-Microprocessor and Arduino
Introduction to Microprocessor, Micro controller, source mode and sink mode.
Arduino pin diagram, Types of Arduino boards and its pin description.

II-LED, LCD, Buzzer & Interfacing
Various types of Actuators, LED introduction, Buzzer introduction, Introduction to LCD 16x2, Types of LCD’s , LCD Pin diagram,Interfacing LED, Buzzer & LCD to Arduino.
LCD: Displaying name on LCD, Setting cursor position in LCD, Display and NO display in LCD, Scrolling Display towards left and right
LED: LED ON & OFF,TWO LED interfacing (Alternate LED ON & OFF), 1234 and 4321 LED pattern. LCD display with LED status.
Buzzer: Interfacing Buzzer with LED and LCD.

III-Motor driver and Interfacing to Arduino
L293D Pin diagram, Introduction to L293D Motor driver. Interfacing DC motors to Arduino, Designing robot

IV-Sensors and Programming
Introduction to soil moister sensor, gas sensor, Ultra Sonic sensor, Temperature sensor, LDR sensor, IR sensor and PIR sensor and its pin configurations.
1. Interfacing Soil moister sensor & gas sensor to Arduino
2. Interfacing LED, LCD and gas sensor
3. Interfacing Ultrasonic sensor to Arduino and displaying distance on screen
4. Interfacing Ultrasonic sensor to LED, LCD
5. Interfacing Temperature sensor to Arduino and Interfacing with LED.
6. Interfacing LDR sensor to Arduino and interfacing with LED.
7. Interfacing IR sensor to Arduino and interfacing with LED and buzzer.
8. Interfacing to PIR sensor to Arduino.

V-Bluetooth & RFID
Introduction, Principle, Bluetooth pin configuration, Two modes of Bluetooth, Interfacing Bluetooth to Arduino and LCD.
RFID: Introduction RFID, Principle, Interfacing to Arduino and LED.
Expecting outcomes through assessments:

1: Designing Automatic Street Light
2: Smart stick for blind person.
3: Design Smart water tank

TEXTBOOKS:

REFERENCE BOOKS:

CO-PO MAPPING:

<table>
<thead>
<tr>
<th>Course Outcomes (COs)</th>
<th>Program Outcomes (POs)</th>
<th>Program Specific Outcomes*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PO 1  PO 2  PO 3  PO 4  PO 5  PO 6  PO 7  PO 8  PO 9  PO 10  PO 11  PO 12  PSO 1  PSO 2</td>
<td></td>
</tr>
<tr>
<td>CO1</td>
<td>H   M   M</td>
<td></td>
</tr>
<tr>
<td>CO2</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>CO3</td>
<td></td>
<td>H</td>
</tr>
<tr>
<td>CO4</td>
<td></td>
<td>H</td>
</tr>
</tbody>
</table>