NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY, GREATER NOIDA
(An Autonomous Institute)

Affiliated to

DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, LUCKNOW

Evaluation Scheme & Syllabus

For

MCA (Integrated)
Second Year

(Effective from the Session: 2023-24)

NOIDA INSTITUTE OF ENGINEERING & TECHNOLOGY,GREATER NOIDA
(An Autonomous Institute)

Evaluation Scheme MCA (Integrated)

SEMESTER III

S.No	Subject Codes	Subjects	Periods			Evaluation Schemes				End Semester		Total	Credit
			L	T	P	CT	TA	Total	PS	TE	PE		
1	**AMICA0301**	**Operating Systems**	3	1	0	30	20	50		100		150	4
2	**AMICA0302**	**Data Structures**	3	1	0	30	20	50		100		150	4
3	**AMICA0303**	**Accounting and Financial Management**	3	1	0	30	20	50		100		150	4
4	**AMICA0304**	**Emerging Trends and Technology**	2	1	0	30	20	50		100		150	3
5	**AMICA0355**	**Advanced Python Lab**	0	0	8				50		100	150	4
6	**AMICA0351**	**Operating Systems Lab**	0	0	4				50		50	100	2
7	**AMICA0352**	**Data Structures Lab**	0	0	4				50		50	100	2
8	**AMICA0359**	**Internship Assessment-I**	0	0	2				25		25	50	1
		MOOCs											
		TOTAL										**1000**	**24**

Please Note: -

Internship(2-3weeks) shall be conducted during summer break after II semester and will be assessed during III semester

List of MOOCs (Coursera) Based Recommended Courses for Second Year (Semester-III) MCA(Integrated)

S.No.	Subject Code	Course Name	University/Industry Partner Name	No. of Hours
1	AMC0012	Human Centered Design for Inclusive Innovation	University of Toronto	14
2	AMC0004	Python Basics	University of Michigan	36

Abbreviation Used: -

L:Lecture,T:Tutorial,P:Practical,CT:ClassTest,TA:TeacherAssessment,PS:PracticalSessional,TE:TheoryEndSemester Exam. ,PE: Practical End Semester Exam.

NOIDA INSTITUTE OF ENGINEERING & TECHNOLOGY,GREATER NOIDA
(An Autonomous Institute)

Evaluation Scheme MCA (Integrated)

SEMESTER IV

S. No	Subject Codes	Subjects	Periods			Evaluation Schemes				End Semester		Total	Credit
			L	T	P	CT	TA	Total	PS	TE	PE		
1	AMICA0401	Cloud Computing	3	1	0	30	20	50		100		150	4
2	AMICA0402	Database Systems	3	1	0	30	20	50		100		150	4
3	AMICA0403	Software Engineering & Design	3	0	0	30	20	50		100		150	3
4	AMICA0404	Design Thinking-II	2	1	0	30	20	50		100		150	3
5	AMICA0452	Database Systems Lab	0	0	4				50		50	100	2
6	AMICA0455	Object Oriented Techniques Using Java Lab	0	0	8				50		100	150	4
7	AMICA0453	Project Based on Software Engineering & Design Lab	0	0	4				50		50	100	2
8	AMICA0451	Cloud Computing Lab	0	0	4				50		50	100	2
		MOOCs											
		TOTAL										1050	24

List of MOOCs (Coursera) Based Recommended Courses for Second Year (Semester-IV) MCA(Integrated)

S.No.	Subject Code	Course Name	University/Industry Partner Name	No. of Hours
1	AMC0206	Introduction to Java and Object-Oriented Programming	University of Pennsylvania	19
2	AMC0207	AWS Cloud Technical Essentials	AWS	25

PLEASE NOTE: -

Internship(2-3weeks) shall be conducted during summer break after semester-IV and will be assessed during Semester-V.

Abbreviation Used: -

L:Lecture,T:Tutorial,P:Practical,CT:ClassTest,TA:TeacherAssessment,PS:PracticalSessional,TE: TheoryEndSemester Exam. ,PE: Practical End Semester Exam.

(Effective from the Session: 2023-2024)

Course Code	AMICA0301	L	T	P	Credit
Course Title	Operating Systems	3	1	0	4

Course Objective: Objective of this course is to: **Duration: 46 Hours**

The objective of this course is to provide an understanding of the basic structure and functions of an operating system and deliver the skills needed to develop Unix/Linux shell programs.

Pre-requisites: Basic knowledge of computer fundamentals, Data structure and Computer organization.

Course Contents / Syllabus

UNIT-I **Fundamentals of Operating Systems** **8 Hours**

Fundamentals of Operating Systems: Operating System, Operatic System characteristics, Functions of Operating Systems, Types of Operating System, Layered Structure, System call, Kernel, Multiprogramming and Multitasking, Overview of Windows OS, Unix/Linux OS Process Management: Process Management: Process Concepts, State Transition Diagram. Types of Schedulers: Long Term, Mid Term, Short Term Process Control Block, Inter process communication Process Management: Process Concepts, State Transition Diagram. Types of Schedulers: Long Term, Mid Term, Short Term Process Control Block, Inter process communication CPU Scheduling: CPU Scheduling Criteria, Pre-emptive and Non Pre-emptive Scheduling, Scheduling Algorithm: FCFS, SJF, SRTF, Round Robin, Priority Scheduling, Multilevel Queue Scheduling and Multilevel Feedback Queue Scheduling, Context Switching.

UNIT-II **Process Synchronisation** **8 Hours**

Process Synchronisation: Critical Section problem & their solutions, Introduction to Semaphores Classical Problems of Synchronization (Producer Consumer Problem, Readers Writer Problem, Dining philosophers' problem) Dead Locks: Dead locks: – Characterization, Deadlock concepts & Handling Techniques (Prevention and Detection & Recovery), Dead Lock Avoidance: Banker's Algorithm.

UNIT-III **Memory management** **8 Hours**

Memory Management: Memory Management: Background, Swapping, Contiguous and Non Contiguous memory allocation, Paging, Segmentation, Segmentation with paging. Virtual Memory: Background, Demand paging, Allocation of frames: First Fit, Best Fit, and Worst Fit, Page replacement algorithms (FCFS, Optimal, LRU), Balady's Anomaly, Thrashing Disc Scheduling: Disk Scheduling: FCFS, SSTF, SCAN, C-SCAN, LOOK and C-LOOK File Management System: File Management: Concept and Organization, Access Methods, File System Implementation Directory Structures, Allocation Methods, Free Space Management, Secondary Storage Structure, File System Security and Protection.

UNIT-IV **Linux Administration** **8 Hours**

Linux Administration: Linux Components, Shells, Installation of Linux, Virtualization: Definition, Types, Advantages, Virtualization tools. User Administration, Files: Type, Ownership, Permissions and manipulations Commands: Internal and External, Directory and File commands, I/O commands, Pipes, Filters, shell commands. Linux Tools Linux Networking Commands: ipconfig, traceroute, tracepath, ping, host, hostname, iwconfig. System Admin: man, uptime, users, service, pkill, ps.

UNIT-V **Shell Programming & VI Editor** **8 Hours**

Shell Programming & VI Editor: Shell Programming - shell script features, shell variables, writing and executing a shell script, positional parameters. Introduction to VI editor, VI editor Models, Invoking VI editor, Configuring the vi environment, The process - parent and child process, process creation, process related commands, Branching control structures- if, case etc., Loop control structures– while, until, for, etc., Jumping control structures – break, continue, exit, etc., Integer and Real arithmetic in shell programs

Course Outcome:

Course outcome: After completion of this course students will be able to:

CO 1	Understand operating system concepts, functions and design CPU Scheduling algorithms.
CO 2	Analyse the various issues related to inter process communication like Synchronization and Deadlocks
CO 3	Simplify the concepts of Memory Management and Implement disk scheduling algorithms.
CO4	Implement and use Linux utilities to create and manage simple file processing operations.

CO5	Demonstrate shell scripts to perform more complex tasks in shell programming environment.

Textbooks:

1. Silberschatz Abraham, Galvin Peter Baer and Gagne Greg, "Operating System Concepts Essentials" 8th edition, 2010
2. Tanenbaum Andrew S., "Modern Operating Systems", Pearson Education, 4th edition, 2014
3. Cannon Jason, "Linux for Beginners: An Introduction to the Linux Operating System and Command Line", Kindle edition, 2014
4. Sobell Marks G., "A practical guide to Linux: Commands, Editors and Shell Programming",, 4th edition, 2017

Reference Books:

1. Stallings William "Operating Systems: Internals and Design Principles", 8th edition, 2014
2. Crowley Charles Patrick, "Operating System: A Design-oriented Approach", 9th edition, 2017
3. Nutt Gary J., "Operating Systems: A Modern Perspective", Longman Publishing Group, 3rd edition, 1997
4. Bach Maurice J,,"Design of the UNIX Operating Systems", 1st edition, 2015
5. Bovet Daniel Pierre and Cesati Marco, "Understanding the Linux Kernel", O'Reilly Media 1st edition, 2000
6. Tanenbaum A.S. and Woodhull A.S., "Operating Systems Design and Implementation", Prentice Hall, 3rd edition, 2006.

Link:

https://www.youtube.com/watch?v=783KAB-tuE4
https://www.youtube.com/watch?v=Bxx2_aQVeeg
https://www.youtube.com/watch?v=ZaGGKFCLNc0
https://nptel.ac.in/courses/106/105/106105214/
https://www.youtube.com/watch?v=NShBeqTkXnQ
https://www.youtube.com/watch?v=4hCih9eLc7M
https://www.youtube.com/watch?v=9YRxhlvt9Zo
https://www.youtube.com/watch?v=UczJ7misUEk
https://www.youtube.com/watch?v=_IxqinTs2Yo
https://www.youtube.com/watch?v=IwESijQs9sM
https://www.youtube.com/watch?v=-orfFhvNBzY
https://www.youtube.com/watch?v=2OobPx246zg&list=PL3-wYxbt4yCjpcfUDzTgD_ainZ2K3MUZ&index=10
https://www.youtube.com/watch?v=AnGOeYJCv6s
https://www.youtube.com/watch?v=U1Jpvni0Aak

Course Code	AMICA0302	L	T	P	Credit
Course Title	DATA STRUCTURES	3	1	0	4

Course objective: Learn the basic concepts of algorithm analysis, along with implementation of linear and non-linear data structures.

Pre-requisites: Knowledge of programming languages, basics of mathematics, organising and problem-solving ability.

Course Contents / Syllabus

UNIT-I	Introduction to Data Structures	8 hours

Data Types: Types of Data Structures- Linear & Non-Linear Data Structures, List, Tuple, Set, Dictionary. Arrays: Derivation of Index Formulae for 1-D,2-D,3-D and n-D Array. Analysis of algorithms: Time and Space Complexity of an algorithm, Asymptotic notations (Big Oh, Big Theta and Big Omega).

UNIT-II	Stack & Queues	8 hours

Stacks: Primitive Stack operations: Push & Pop, mutual conversion of Infix, Prefix, Postfix, Evaluation of postfix expression. Recursion: Principles of recursion, Types of Recursion, Problem solving using iteration, Tower of Hanoi, Trade-offs between iteration and recursion. Queues: Operations on Queue: Create, Insert, Delete, Full and Empty, Circular queues, Dequeue and Priority Queue.

UNIT-III	Linked Lists	8 hours

Linked lists: Comparison of Array, List and Linked list Types of linked list: Singly Linked List, Doubly Linked List, Circular Linked List, Polynomial Representation and Addition of Polynomials**.**

UNIT-IV	Trees	8 hours

Trees: Basic terminology, Binary Trees, Binary Tree Representation, Binary Search Tree, Strictly Binary Tree, Complete Binary Tree, Extended Binary Tree, Tree Traversal algorithms: In-order, Pre-order and Post-order. Constructing Binary Tree from given Tree Traversal, Binary Heaps, Heap Operations, Threaded Binary trees, Traversing Threaded Binary trees, AVL Tree, B-Tree.

UNIT-V	Graphs	8 hours

Graphs: Terminology used with Graph, Graph Sorting Techniques: Representations: Adjacency matrices, Adjacency List. Connected Component, Spanning Trees, Prim's and Kruskal's algorithm, Shortest Path algorithms: Dijkstra Algorithm, Floyd Warshall's Algorithm. Hashing: Sorting Algorithms. Hashing: Hash Functions, Collision-Resolution Techniques.

Course Outcome:

CO1	Describe the need of data structure and algorithms in problem solving and Analyse Time space trade-off.	K4
CO2	Design, implement and evaluate the real-world applications using stacks, Queues.	K5
CO3	Compare and contrast the advantages and disadvantages of linked lists over arrays and implement operations on different types of linked list.	K4
CO4	Implement and evaluate the real-world applications using non-linear data structures.	K5
CO5	Identify and analyse the computational efficiencies of searching and sorting algorithms in real world problems.	K4

Reference Books:

1. Thareja, "Data Structure Using C" Oxford Higher Education.
2. AK Sharma, "Data Structure Using C", Pearson Education India.
3. P. S. Deshpandey, "C and Data structure", Wiley Dreamtech Publication.
4. R. Kruse etal, "Data Structures and Program Design in C", Pearson Education.
5. Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, "Data Structures and Algorithms in Python

(An Indian Adaptation)", Wiley Publication.
6. Aaron M. Tenenbaum, Yedidyah Langsam and Moshe J. Augenstein, "Data Structures Using C and C++", PHI Learning Private Limited, Delhi India.
7. Horowitz and Sahani, "Fundamentals of Data Structures", Galgotia Publications Pvt Ltd Delhi India.
8. Lipschutz, "Data Structures" Schaum's Outline Series, Tata McGraw-hill Education (India) Pvt. Ltd.

Text Books:

1. Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, "Data Structures and Algorithms in Python (An Indian Adaptation)", Wiley Publication
2. Aaron M. Tenenbaum, Yedidyah Langsam and Moshe J. Augenstein, "Data Structures Using C and C++", PHI Learning Private Limited, Delhi India
3. Horowitz and Sahani, "Fundamentals of Data Structures", Galgotia Publications Pvt Ltd Delhi India.
4.Lipschutz, "Data Structures" Schaum's Outline Series, Tata McGraw-hill Education (India) Pvt. Ltd.

Links:

https://nptel.ac.in/courses/106/106/106106127/
https://www.youtube.com/watch?v=zWg7U0OEAoE&list=PLBF3763AF2E1C572F
https://www.youtube.com/watch?v=4OxBvBXon5w&list=PLBF3763AF2E1C572F&index=22
https://www.youtube.com/watch?v=cR4rxllyiCs&list=PLBF3763AF2E1C572F&index=23
https://nptel.ac.in/courses/106/106/106106127/
https://www.youtube.com/watch?v=9zpSs845wf8&list=PLBF3763AF2E1C572F&index=24
https://www.youtube.com/watch?v=hk5rQs7TQ7E&list=PLBF3763AF2E1C572F&index=25
https://www.youtube.com/watch?v=KW0UvOW0XIo&list=PLBF3763AF2E1C572F&index=5

Course code	AMICA0303	L T P	Credit
Course title	Accounting and Financial Management	3 1 0	4

Course objective:

This course aims to develop cognizance about the importance of accounting in organization financial statements. Students will become acquainted with the mechanics of financial statement preparation, understanding corporate financial statements, their analysis and interpretation, the significance of IFRS in accounting discipline, and the notion of management quality analysis and wealth generation.

Pre-requisites:

Basic knowledge of finance and accounting.

Course Content / Syllabus

UNIT-I	Introduction to Accounting	7 Hours

Overview: Accounting Concepts, Conventions and Principles; Meaning and Scope of Accounting: Evolution and Users of Accounting, Types of Accounting, Basic Accounting terminologies, Accounting Equation; Artificial Intelligence in Accounting.

UNIT-II	Mechanics of Accounting	10 Hours

Accounting principles and standards; Matching of Indian Accounting Standards with International Accounting Standards, IAS, IFRS, Ind AS Double entry system of Accounting, Golden Rules of Accounting, journalizing of transactions; Ledger posting and Trial Balance.

UNIT-III	Preparation of Financial Statements	5 Hours

Financial Statements and its presentation, Preparation of Income Statement and Balances sheet as per companies Act 2013, Use of Excel in preparation of Balance sheet.

UNIT-IV	Financial Statement Analysis	8 Hours

Analysis of financial statement: Ratio Analysis- solvency ratios, profitability ratios, activity ratios, liquidity ratios, market capitalization ratios ; Common Size Statement ; Comparative Balance Sheet and Trend Analysis.

UNIT-V	Working Capital Management	10 Hours

Concept of Gross and Net Working Capital, Preparation of Schedule of Changes in Working Capital, Cash Flow Statement: Various cash and non-cash transactions, flow of cash, preparation of Cash Flow Statement and its analysis.

Course outcome:

At the end of the course students will be able to

CO 1	Understand the basic objective of the course and comprehend texts for professional reading tasks in preparation for an International Certification in Business English.	L2
CO 2	Write professionally in simple and correct English.	L5
CO 3	Interpret listening tasks for better professional competence.	L3
CO 4	Recognize the elements of effective speaking with emphasison applied phonetics.	L1
CO 5	Apply the skill of speaking at the workplace.	L3

Textbooks

1.Maheshwari, S. N., & Maheshwari, S. K. (2001). Advanced Accountancy Volume-I. Vikas Publishing House.

2.Tulsian, P. C. (2002). Financial Accounting. Pearson Education India.

Reference Books

1.Gupta, K. (2011). Khan, MY and Jain, PK, Financial Management: Text, Problems and Case. Journal of Services Research, 11(2).

2.Hasan, A. R. (2021). Artificial Intelligence (AI) in accounting & auditing: A Literature review. Open Journal of Business and Management, 10(1), 440-465.

3.Schroeder, R. G., Clark, M. W., & Cathey, J. M. (2022). Financial accounting theory and analysis: text and cases. John Wiley & Sons.

4.Collier, P. M. (2015). Accounting for managers: Interpreting accounting information for decision making. John Wiley & Sons.

5.Gupta, K. (2011). Khan, MY and Jain, PK, Financial Management: Text, Problems and Case. Journal of Services Research, 11(2).

6.Hasan, A. R. (2021). Artificial Intelligence (AI) in accounting & auditing: A Literature review. Open Journal of Business and Management, 10(1), 440-465.

Links:

https://abmagazine.accaglobal.com/middle-east-south-asia/en.html
https://www.icai.org/category/e-journal
https://www.journalofaccountancy.com/news.html
https://www.accountingseed.com/resource/blog/learn-accounting-for-free-with-these-resources/

Course Code	AMICA0304	L	T	P	Credit
Course Title	Emerging Trends & Technologies	2	1	0	3

Course objective:

Provide an understanding of the role computation can play in solving problems.
Position students so that they can compete for research projects and excel in subjects with programming components.

Pre-requisites:

The student must understand basic computer terminology.
The student must have knowledge of some programming language.

Course Contents / Syllabus

UNIT-I	Introduction to Computer	8 hours

Definition Computer: Computer Hardware & Computer Software Components: Hardware – Introduction, Input devices, Output devices, Central Processing Unit Memory – Primary and Secondary Software – Introduction, Types– System and Application Computer Languages Introduction, Concept of Compiler, Interpreter & Assembler Algorithms – Introduction, Definition, Characteristics, Limitations, Conditions in pseudo-code, Loops in pseudo code.

UNIT-II	Operating system	8 hours

Operating system Definition, Functions, Types, Classification, Elements of command based and GUI based operating system. Windows Operating System Commands Computer Network: Overview, Standalone, Types (LAN, WAN and MAN), Data communication, topologies.

UNIT-III	Internet	8 hours

Internet: Overview, Architecture, Functioning, Basic services like WWW, FTP, Telnet, Gopher etc., Search engines, E-mail, Web Browsers. Internet of Things (IoT) Definition, Sensors, their types and features, Smart Cities, Industrial Internet of Things.

UNIT-IV	MS-Office	8 hours

MS-Office: Basic Concepts, Features, Applications and handling of MS-Word, MS-PowerPoint and MS-Excel.

UNIT-V	Emerging Technologies	8 hours

Emerging Technologies: Introduction, overview, features, limitations and application areas of Cloud Computing, Big data, Grid Computing, Artificial Intelligence and Virtual Reality.

Course outcome: At the end of course, the student will be able to

CO 1	Understand basics of programming	K1
CO 2	Understand the problem-solving process and apply concepts to real-life situations and data-oriented problem analysis.	K2
CO 3	Use of recursion, searching and sorting algorithm to arrange the data.	K2
CO 4	Understand to evaluate performance of algorithm.	K2
CO 5	Understand the concept of Object-Oriented Programming.	K2

Reference Books

Balagurusamy E.,"Fundamentals of Computers", McGraw-Hill.

Thareja R.,"Fundamentals of Computers", Oxford University Press.

Bindra J.,"The Tech Whisperer-on Digital Transformation and the Technologies that Enableit" Penguin

Balagurusamy E.,"Fundamentals of Computers", McGraw-Hill.

Textbooks: -

1. Raja Raman V., "Fundamentals of Computers", Prentice-Hall of India.
2. Norton P., "Introduction to Computers", McGraw Hill Education.
3. Goel A., "Computer Fundamentals", Pearson.

Link:
https://www.youtube.com/watch?v=eEo_aacpwCw
https://www.youtube.com/watch?v=WJ-UaAaumNA
https://www.youtube.com/watch?v=cNwEVYkx2Kk
https://www.youtube.com/watch?v=W3yttwGE-C0
https://www.youtube.com/watch?v=yCVy5Kw0l8s

Course Code	AMICA0355	L T P	Credit
Course Title	**Advanced Python Lab**	0 0 8	4

Lab Course Outcome:

At the end of the course students will be able to -

CO 1	Write simple python programs.	K_2, K_3
CO 2	Implement python programs using decision control statements	K_3, K_6
CO 3	Writing python programs using user defined functions and modules	K_2
CO 4	Implement programs using python data structures –lists, tuples, set, dictionaries	K_3
CO 5	Write programs to perform input/output operations on files	K_3, K_4

List of Experiment:

List of Fundamental Programs

S.N.	Program Title	Category
1	Write a program illustrating class definition and accessing class members.	Class and object
2	Write a program to implement default constructor, parameterized constructor, and destructor.	Class and object
3	Create a Python class named Rectangle constructed by a length and width. a. Create a method called area which will compute the area of a rectangle.	Class and object
4	Create a class called Numbers, which has a single class attribute called MULTIPLIER, and a constructor which takes the parameters x and y (these should all be numbers). a. Write an instance method called add which returns the sum of the attributes x and y. b. Write a class method called multiply, which takes a single number parameter a and returns the product of a and MULTIPLIER.	Class and object
5	Create a class named as Student to store the name and marks in three subjects. Use List to store the marks. a. Write an instance method called compute to compute total marks and average marks of a student. b. Write a method called display to display student information.	Class and object
6	Create a Python class named Circle constructed by a radius. Use a class variable to define the value of constant PI. a. Write two methods to be named as area and circum to compute the area and the perimeter of a circle respectively by using class variable PI. b. Write a method called display to print area and perimeter.	Class and object
7	Write a program that has a class called Fraction with attributes numerator and denominator. a. Write a method called get data to enter the values of the attributes. b. Write a method show to print the fraction in simplified form.	Class and object
8	Write a program that has a class Numbers with a list as an instance variable. a. Write a method called insert_element that takes values from user. b. Write a class method called find_max to find and print largest value in the list.	Class and object
9	Create a class called Complex. Write a menu driven program to read, display, add and subtract two complex numbers by creating corresponding instance methods.	Class and object

10	Write a program that has a class Point with attributes x and y. a. Write a method called midpoint that returns a midpoint of a line joining two points. b. Write a method called length that returns the length of a line joining two points.	Class and object
11	Create a class called Complex. Write a menu driven program to read, display, add and subtract two complex numbers by creating corresponding instance methods.	Class and object
12	Write a Python program to create a class called "Rectangle" with attributes length and width. Include methods to calculate the perimeter and area of the rectangle.	Class and object
13	Implement a Python class called "Bank Account" with attributes account number, account holder name, and balance. Include methods to deposit and withdraw money from the account.	Class and object
14	Write a Python program to create a class called "Student" with attributes roll number, name, and marks in three subjects. Include a method to calculate the average marks of the student.	Class and object
15	Implement a Python class called "Car" with attributes make, model, and year. Include methods to start the car, stop the car, and display its details.	Class and object
16	Write a Python program to create a class called "Book" with attributes title, author, and price. Include methods to calculate the discounted price of the book based on a discount percentage provided.	Class and object
17	Implement a Python class called "Bank" with attributes bank name and branch. Include methods to add a new account, display all accounts, and search for an account based on the account number.	Class and object
18	Write a Python program to create a class called "Rectangle" with attributes length and width. Include a method to check if the rectangle is a square or not.	Class and object
19	Implement a Python class called "Employee" with attributes name, designation, and experience. Include methods to promote an employee to a higher designation based on their experience.	Class and object
20	Write a Python program to create a class called "Employee" with attributes name, employee ID, and salary. Include a method to display the employee details.	Class and object
21	Write a program to illustrate the use of following built-in methods: a. hasattr(obj,attr) b. getattr(object, attribute_name [, default]) c. setattr(object, name, value) d. delattr(class_name, name)	Magic Method
22	Write a Program to illustrate the use of __str__(), __repr__(), __new__, __doc__, __dict__, __name__ and __bases__ methods.	Magic Method
23	Write a program to create class Employee. Display the personal information and salary details of 5 employees using single inheritance.	Inheritance
24	WAP that extends the class Employee. Derive two classes Manager and Team Leader from Employee class. Display all the details of the employee working under a particular Manager and Team Leader.	Inheritance
25	Write a program that has a class Point. Define another class Location which has two objects (Location and destination) of class Point. Also, define a function in Location that prints the reflection on the y-axis.	Inheritance

26	Write a program that create a class Distance with member's km and metres. Derive classes School and office which store the distance from your house to school and office along with other details.	Inheritance
27	Write a program to create an abstract class Vehicle. Derive three classes Car, Motorcycle and Truck from it. Define appropriate methods and print the details of vehicle.	Inheritance
28	Write a program to demonstrate hybrid inheritance and show MRO for each class.	Inheritance
29	Write a program to overload + operator to multiply to fraction object of fraction class which contain two instance variable numerator and denominator. Also, define the instance method simplify() to simplify the fraction objects.	Polymorphism
30	26. Write a program to compare two-person object based on their age by overloading > operator.	Polymorphism
31	Write a program to overload in operator.	Polymorphism
32	WAP to create a Complex class having real and imaginary as it attributes. Overload the +,-,/,* and += operators for objects of Complex class	Polymorphism
33	WAP to Show the concept of inner function.	Functional Programming
34	WAP to create a decorator which will convert a string into upper case string.	Functional Programming
35	WAP to show the concept of nested decorator.	Functional Programming
36	WAP to decorate a function with arguments.	Functional Programming
37	WAP to decorate instance method	Functional Programming
38	WAP to calculate sum of 1,2,3,4,5 using reduce function.	Functional Programming
39	WAP to generate numbers from 1 to 10 using generator.	Functional Programming
40	WAP to decide number is even or odd using generator.	Functional Programming
41	WAP to generate square of 1,2,3,4,5,6,7,8,9,10 using generator.	Functional Programming
42	WAP to generate square of even number up to 10 using generator and save in list.	Functional Programming
43	WAP to make a co-routine which will print all name with prefix Dear.	Functional Programming
44	WAP to close a co-routine.	Functional Programming
45	WAP to iterate tuple using iter() and next() method.	Functional Programming
46	WAP to iterate a string using iter and next method.	Functional Programming
47	WAP to print numbers from 1 to 20 using iterator and generate Stop Iteration exception once we reach limit.	Functional Programming
48	Hello World: Display a simple "Hello, World!" message box.	GUI
49	Button: Create a button that displays a message when clicked.	GUI Programming

50	Entry: Create a text entry field and display the entered text.	GUI Programming
51	Check button: Create a checkbox and display the selected options	GUI Programming
52	Radio button: Create radio buttons and display the selected option.	GUI Programming
53	List box: Create a list box and display the selected items.	GUI Programming
54	Text: Create a text area and display the entered text.	GUI Programming
55	Menu: Create a menu with different options.	GUI Programming
56	Message: Display a message in a dialog box.	GUI Programming
57	Progress bar: Create a progress bar that updates over time python	GUI Programming
58	Scale: Create a scale widget and display the selected value.	GUI Programming
59	Spin box: Create a spin box and display the selected value.	GUI Programming
60	Canvas: Create a canvas and draw shapes on it.	GUI Programming
61	Label Frame: Create a labeled frame with widgets inside.	GUI Programming
62	Scrollbar: Add a scrollbar to a widget like a text area or list box	GUI Programming
63	Frame: Create a frame and place widgets inside it.	GUI Programming
64	Tree view: Create a tree view widget to display hierarchical data	GUI Programming
65	Notebook: Create a notebook widget with tabs.	GUI Programming
66	File Dialog: Open a file dialog to select a file.	GUI Programming
67	Color Dialog: Open a color dialog to select a color.	GUI Programming
68	Button Counter: Create a button that increments a counter when clicked.	GUI Programming
69	Checkbox List: Display a list of checkboxes and show selected options.	GUI Programming
70	Dropdown Menu: Create a dropdown menu with multiple options.	GUI Programming
71	Slider Value Display: Display the current value of a slider widget.	GUI Programming
72	Text Input and Button: Take user input in a text box and display it when a button is clicked.	GUI Programming
73	Radio Buttons: Present a set of options as radio buttons and display the selected option.	GUI Programming
74	Progress Bar: Show the progress of a task using a progress bar widget.	GUI Programming
75	Password Input: Create a password input field that hides the entered characters.	GUI Programming
76	File Uploader: Enable users to upload files and display the selected file name.	GUI Programming
77	Creating Arrays: Create NumPy arrays using various methods like np.array(), np.zeros(), np.ones(), np.arange(), etc.	NumPy
78	Array Shape and Size: Get the shape and size of a NumPy array using the shape and size attributes.	NumPy

79	Array Indexing: Access and modify individual elements of a NumPy array using indexing	NumPy
80	Array Slicing: Extract a subset of elements from a NumPy array using slicing.	NumPy
81	Array Reshaping: Change the shape of a NumPy array using the reshape() function.	NumPy
82	Array Arithmetic: Perform basic arithmetic operations (addition, subtraction, multiplication, division) on NumPy arrays.	NumPy
83	Array Broadcasting: Perform element-wise operations on arrays with different shapes using broadcasting rules.	NumPy
84	Array Aggregation: Calculate aggregate values on arrays, such as sum(), min(), max(), mean(), etc. using NumPy	NumPy
85	Array Transposition: Transpose a NumPy array using the transpose() function.	NumPy
86	Write a program that demonstrates advanced array indexing techniques, such as indexing with boolean arrays or using fancy indexing to select specific elements or subsets of an array.	NumPy
87	Write a program using NumPy to perform data manipulation tasks, such as sorting arrays, removing duplicates, or finding unique elements in an array.	NumPy
88	Array Sorting: Sort the elements of a NumPy array using the sort() function.	NumPy
89	Array Filtering: Filter elements in a NumPy array based on a condition using Boolean indexing.	NumPy
90	Array Statistics: Calculate statistical measures like mean, median, standard deviation using functions like np.mean(), np.median(), np.std().	NumPy
91	Array Randomization: Generate random numbers or arrays using functions from the np.random module.	NumPy
92	Array Dot Product: Compute the dot product of two NumPy arrays using the dot() function.	NumPy
93	Array Matrix Operations: Perform matrix operations like matrix multiplication, matrix inverse using functions from the np.linalg module.	NumPy
94	Array File I/O: Save and load NumPy arrays from files using functions like np.save() and np.load().	NumPy
95	Array Masking: Create a mask array to select or manipulate specific elements of a NumPy array based on a condition.	NumPy
96	Array Broadcasting: Understand and utilize broadcasting rules in NumPy for efficient computations.	NumPy
97	Write a program to finds the cube root of values using scipy library.	Scipy
98	Write a program to computes the 10**x element-wise using scipy library .	Scipy
99	Write a SciPy program to calculate Permutations and Combinations.	Scipy
100	Write a SciPy program to calculates the inverse of any square matrix.	Scipy
101	Write a SciPy program to calculates the Eigenvalues and Eigenvector.	Scipy
102	Read and Load a CSV File into a Pandas DataFrame using pandas.read_csv.	Panda
103	Access and Display the First N Rows of a DataFrame using	Panda

	DataFrame.head(N).	
104	Access and Display the Last N Rows of a DataFrame using DataFrame.tail(N).	Panda
105	Retrieve Basic Information about a DataFrame using DataFrame.info.	Panda
106	Perform Descriptive Statistics on a DataFrame using DataFrame.describe.	Panda
107	Filter Rows of a DataFrame based on a Condition using Boolean Indexing.	Panda
108	Rename Columns in a DataFrame using DataFrame.rename.	Panda
109	Group Data in a DataFrame using DataFrame.groupby.	Panda
110	Perform Aggregation on Grouped Data using GroupBy.agg.	Panda
111	Sort a DataFrame by One or Multiple Columns using DataFrame.sort_values.	Panda
112	Perform Basic Arithmetic Operations on Columns of a DataFrame.	Panda
113	Apply a Function to Each Element or Column of a DataFrame using DataFrame.apply or Data Frame.applymap.	Panda
114	Reshape Data using Pivot Tables using Data Frame.pivot_table.	Panda
115	Perform Data Visualization using pandas.plotting or matplotlib.pyplot.	Panda
116	Save a DataFrame to a CSV File using DataFrame.to_csv.	Panda
117	Perform Data Sampling or Random Selection using DataFrame.sample.	Panda
118	Find the roots of a mathematical equation using SciPy's root-finding functions, such as scipy.optimize.root.	SciPy
119	Fit a polynomial function to a set of data points using SciPy's curve fitting functions, such as scipy. optimize. curve_ fit	SciPy
120	Perform linear regression on a dataset using SciPy's linear regression functions, such as scipy.stats.linregress.	SciPy
121	Calculate the Fast Fourier Transform (FFT) of a signal using SciPy's FFT functions, such as scipy.fft.fft.	SciPy
122	Solve a system of linear equations using SciPy's linear algebra functions, such as scipy.linalg.solve.	SciPy
123	Perform numerical integration using SciPy's integration functions such as scipy.integrate.quad.	SciPy
124	Calculate the eigenvalues and eigenvectors of a square matrix using SciPy's linear algebra functions, such as scipy.linalg.eig.	SciPy
125	Create a Simple Line Plot using matplotlib.pyplot.plot.	matplotlib
126	Create a Scatter Plot using matplotlib.pyplot.scatter.	matplotlib
127	Create a Bar Chart using matplotlib.pyplot.bar.	matplotlib
128	Create a Histogram using matplotlib.pyplot.hist.	matplotlib
129	Create a Pie Chart using matplotlib.pyplot.pie.	matplotlib
130	Create a Box Plot using matplotlib.pyplot.boxplot.	matplotlib
131	Create a Heatmap using matplotlib.pyplot.imshow.	matplotlib
132	Customize Plot Labels and Titles using matplotlib.pyplot.xlabel, matplotlib.pyplot.ylabel, and matplotlib.pyplot.title.	matplotlib
133	Customize Plot Colors, Line Styles, and Marker Styles using matplotlib.pyplot.plot parameters.	matplotlib

134	Add Gridlines to a Plot using matplotlib.pyplot.grid.	matplotlib
135	Add Legends to a Plot using matplotlib.pyplot.legend.	matplotlib
136	Create Subplots using matplotlib.pyplot.subplots.	matplotlib
137	Save a Plot as an Image File using matplotlib.pyplot.savefig.	matplotlib
138	Create 3D Plots using mpl_toolkits.mplot3d module.	matplotlib
139	Create Error Bars on a Plot using matplotlib.pyplot.errorbar.	matplotlib
140	Customize Axis Ticks and Tick Labels using matplotlib.pyplot.xticks and matplotlib.pyplot.yticks.	matplotlib
141	Create a Bar Plot with Stacked Bars using matplotlib.pyplot.bar and the bottom parameter.	matplotlib
142	Create a Scatter Plot using seaborn.scatterplot.	seaborn
143	Create a Line Plot using seaborn.lineplot.	seaborn
144	Create a Bar Plot using seaborn.barplot.	seaborn
145	Create a Histogram using seaborn.histplot.	seaborn
146	Create a Box Plot using seaborn.boxplot.	seaborn
147	Create a Violin Plot using seaborn.violinplot.	seaborn
148	Create a Heatmap using seaborn.heatmap.	seaborn
149	Create a Pair Plot using seaborn.pairplot.	seaborn
150	Create a Joint Distribution Plot using seaborn.jointplot.	seaborn
151	Create a KDE (Kernel Density Estimate) Plot using seaborn.kdeplot.	seaborn
152	Create a Categorical Scatter Plot using seaborn.stripplot.	seaborn
153	Create a Categorical Bar Plot using seaborn.countplot.	seaborn
154	Create a Facet Grid using seaborn.FacetGrid.	seaborn
155	Customize Plot Colors and Styles using seaborn.set_palette and seaborn.set_style.	seaborn
156	Add Error Bars to a Plot using seaborn.barplot or seaborn.pointplot with the ci parameter.	seaborn
157	Create a Clustered Heatmap using seaborn.clustermap.	seaborn
158	Create a Regression Plot using seaborn.regplot.	seaborn
159	Create a Pairwise Relationship Plot using seaborn.pairplot or seaborn.scatterplot with multiple variables.	seaborn
160	Create a Boxen Plot using seaborn.boxenplot.	seaborn
161	Create a Stacked Bar Plot using seaborn.barplot with the hue parameter.	seaborn
162	Write a program to draw a line chart using Plotly	Plotly
163	Write a program to draw a Bar chart using Plotly	Plotly
164	Write a program to draw a Histogram chart using Plotly	Plotly
165	Write a program to draw a scatter plot using Plotly	Plotly
166	Write a program to draw a Bubble chart using Plotly	Plotly
167	Write a program to draw a pie chart using Plotly	Plotly
168	Write a program to draw a Boxplot using Plotly	Plotly
169	Write a program to draw a Violin Plots using Plotly	Plotly
170	Write a program to draw a Gant chart using Plotly	Plotly

171	Write a Python program to find the title tags from a given html document.	Web scrapping
172	Write a Python program to retrieve all the paragraph tags from a given html document.	Web scrapping
173	Write a Python program to get the number of paragraph tags of a given html document.	Web scrapping
174	Write a Python program to extract the text in the first paragraph tag of a given html document.	Web scrapping
175	Write a Python program to find the length of the text of the first <h2> tag of a given html document.	Web scrapping
176	Write a Python program to find the text of the first <a> tag of a given html text.	Web scrapping
177	Write a Python program to find the href of the first <a> tag of a given html document.	Web scrapping
178	Write a Python program to a list of all the h1, h2, h3 tags from the webpage python.org.	Web scrapping
179	Write a Python program to extract all the text from a given web page.	Web scrapping
180	Write a Python program to print the names of all HTML tags of a given web page going through the document tree.	Web scrapping
181	Write a Python program to retrieve children of the html tag from a given web page.	Web scrapping
182	Write a Python program to retrieve all descendants of the body tag from a given web page.	Web scrapping
183	Write a Python program to print content of elements that contain a specified string of a given web page.	Web scrapping
184	Write a Python program to print the element(s) that has a specified id of a given web page.	Web scrapping
185	Write a Python program to create a Beautiful Soup parse tree into a nicely formatted Unicode string, with a separate line for each HTML/XML tag and string.	Web scrapping
186	Write a Python program to find the first tag with a given attribute value in an html document.	Web scrapping
187	Write a Python program to find tag(s) beneath other tag(s) in a given html document.	Web scrapping
188	Write a Python program to find tag(s) directly beneath other tag(s) in a given html document.	Web scrapping
189	Write a Python program to find the siblings of tags in a given html document.	Web scrapping
190	Write a Python program to find tags by CSS class in a given html document.	Web scrapping
191	Write a Python program to change the tag's contents and replace with the given string.	Web scrapping
192	Write a Python program to add to a tag's contents in a given html document.	Web scrapping
193	Write a Python program to insert a new text within a url in a specified position.	Web scrapping
194	Write a Python program to insert tags or strings immediately before specified tags or strings.	Web scrapping

195	Write a Python program to insert tags or strings immediately after specified tags or strings.	Web scrapping
196	Write a Python program to remove the contents of a tag in a given html document.	Web scrapping
197	Write a Python program to extract a tag or string from a given tree of html document.	Web scrapping
198	Write a Python program to remove a tag from a given tree of html document and destroy it and its contents.	Web scrapping
199	Write a Python program to remove a tag or string from a given tree of html document and replace it with the given tag or string.	Web scrapping
200	Write a Python program to wrap an element in the specified tag and create the new wrapper.	Web scrapping
201	Write a Python program to replace a given tag with whatever's inside a given tag.	Web scrapping

Reference Books:

1. Dusty Phillips, Python 3 Object-oriented Programming - Second Edition, O'Reilly
2. Burkhard Meier, Python GUI Programming Cookbook - Third ,Packt
3. DOUG HELLMANN, THE PYTHON 3 STANDARD LIBRARY BY EXAMPLE, :Pyth 3 Stan Libr Exam _2 (Developer's Library) 1st Edition, Kindle Edition.

1. **Text Books:**

1. Magnus Lie Hetland, "Beginning Python-From Novice to Professional"—Third Edition, Apress
2. Peter Morgan, Data Analysis from Scratch with Python, AI Sciences.
3. Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", 2nd edition, Updated for Python 3, Shroff/O'Reilly Publishers, 2016.
4. Miguel Grinberg, Developing Web applications with python, OREILLY

Links:

https://nptel.ac.in/courses/106/106/106106145/
https://www.python-course.eu/python3_inheritance.php
https://realpython.com/courses/functional-programming-python/
https://realpython.com/python-gui-tkinter/
https://nptel.ac.in/courses/106/107/106107220/
https://nptel.ac.in/courses/106/106/106106212/

Course Code	AMICA0351	L	T	P	Credit
Course Title	Operating System Lab	0	0	4	2

Lab Course outcome: At the end of course, the student will be able to

CO 1	Gain all round knowledge of various Linux Commands	K_2
CO 2	Analyze and implement Process Synchronization technique.	K_4, K_5
CO 3	Analyze and implement CPU scheduling algorithms	K_4, K_5
CO 4	Analyze and implement Memory allocation and Memory management techniques.	K_4, K_5
CO 5	Analyze and implement Disk Scheduling Policies.	K_4, K_5

List of Experiment:

List of Fundamental Programs

S.N.	Program Title	Category
1.	Implement FCFS CPU Scheduling algorithm.	CPU Scheduling Algorithms
2.	Implement the given CPU Scheduling algorithm a) SJF b) Priority Based	CPU Scheduling Algorithms
3.	Implement Multi-level Queue CPU Scheduling algorithm.	CPU Scheduling Algorithms
4.	Implement PRIORITY CPU Scheduling Algorithm (For both Pre-emptive and non-pre-emptive).	CPU Scheduling Algorithms
5.	Implement Round-Robin CPU Scheduling Algorithm	CPU Scheduling Algorithms
6.	Implement Multilevel Queue CPU Scheduling Algorithm.	CPU Scheduling Algorithms
7.	Execute the RACE Condition of Process Synchronization.	CPU Scheduling Algorithms
8.	Implement the Producer–consumer problem using semaphores.	Process Synchronization
9.	Implement the Producer–consumer problem using semaphores.	Process Synchronization
10.	Design a code and implement the Dinning Philosopher problem	Process Synchronization
11.	Execute an algorithm for deadlock detection.	Deadlock
12.	Implement Banker's algorithm of Deadlock Avoidance	Deadlock

13.	Implement Contiguous memory fixed size partition scheme.	Contiguous Memory Allocation Techniques
14.	Implement Contiguous memory variable size partition scheme.	Contiguous Memory Allocation Techniques
15.	Simulate the First-Fit contiguous memory allocation technique.	Continuous Memory Allocation
16.	Simulate the Best-Fit contiguous memory allocation technique.	Continuous Memory Allocation
17.	Simulate the Worst-Fit contiguous memory allocation technique.	Continuous Memory Allocation
18.	Implement the Non Continuous Memory Allocation by using Paging.	Non Continuous Memory Allocation
19.	Write a Program to simulate the FIFO page replacement algorithm.	Page Replacement Techniques
20	Write a Program to simulate the LRU page replacement Algorithm.	Page Replacement Techniques
21.	Write a Program to simulate the Optimal page replacement Algorithm.	Page Replacement Techniques
22.	Write a Program to simulate the FCFS Disk Scheduling Algorithm.	Disc Scheduling
23.	Write a Program to simulate the SSTF Disk Scheduling Algorithm.	Disc Scheduling
24.	Implement SCAN and C-SCAN Disk Scheduling Algorithms.	Disc Scheduling
25.	Implement LOOK and C-LOOK Disk Scheduling Algorithms.	Disc Scheduling
26	Design an algorithm and implement to organize the file using the single-level directory.	File Management System
27.	Write a program to organize the file using two-level directories.	File Management System
28.	Write a C program to Sequential files for processing the student information.	File Management System
29.	Write a C program for random access files for processing the employee details.	File

		Management System
30.	Execute Various types of Linux Commands (Miscellaneous, File oriented, Directory oriented)	Linux permissions for users, groups, and others
31.	Execute a shell program, which accepts the name of a file from standard input and performs the File Readable test on it.	Linux permissions for users, groups, and others
32.	Design and execute the code to accept the name of a file from standard input and performs the File Writable test on it.	Linux permissions for users, groups, and others
33.	Implement a shell program, which accepts the name of a file from standard input and performs the File Writable test on it.	Linux permissions for users, groups, and others
34.	Case Study	Linux File Management
35.	Case Study	Linux File Management
36.	Implement Linux Networking Commands: ipconfig, traceroute, tracepath, ping, host, hostname, iwconfig.	Linux Networking Commands
37.	Implement the following system admin commands in Linux: man, uptime, users, service, pkill, ps.	Linux System Admin Commands
38.	Implement the following in Unix: a) Process creation, b) Sleep Command c) Sleep command using getpid.	Unix Commands
39.	Analysis system calls of unix operating system (fork and exit)	Unix Commands
40.	Implement Unix commands for a) Signal handling using kil, b) Wait command, c)top	Unix Commands
41.	Write a program to simulate UNIX commands like cp, ls, and grep.	Unix Commands
42.	Implement Unix Shell programming for concatenation of two strings.	Unix Shell programming
43.	Implement Unix Shell programming for a) Comparison of two strings b) Maximum of three numbers.	Unix Shell programming
44.	Implement Unix Shell programming for Fibonacci series	Unix Shell programming

45.	Write a program in Unix to whether the given year is a) a leap year or not b) Arithmetic operation using cases.	Unix Shell programming
46.	Write a program in Unix for factorial of a number.	Unix Shell programming
47.	Write a program in Unix to swap the two integers	Unix Shell programming
48.	Write a program in Unix to whether the given number is prime or not.	Unix Shell programming

Textbooks:

1. Abraham Silberschatz, Peter Baer Galvin and Greg Gagne, "Operating System Concepts Essentials" 8th Edition, 2010
2. Andrew S. Tanenbaum, "Modern Operating Systems", Pearson Education, 4th Edition, 2014
3. Jason Cannon, "Linux for Beginners: An Introduction to the Linux Operating System and Command Line", 2014
4. Marks G. Sobell, "A practical guide to Linux: Commands, Editors and Shell Programming" Fourth Edition, 2017

Reference Books:

1. "Operating Systems: Internals and Design Principles", William Stallings, 8th Edition, 2014
2. "Operating System: A Design-oriented Approach", Charles Patrick Crowley, 9th Edition, 2017
3. "Operating Systems: A Modern Perspective", Gary J. Nutt, 1997
4. "Design of the UNIX Operating Systems", Maurice J. Bach., 1st Edition, 2015

Course Code	AMICA0352	L	T	P	Credit
Course Title	**Data Structure Lab**	0	0	4	2

Course outcome: At the end of course, the student will be able to

CO 1	Implement operations on single and multi-dimensional array	K_3
CO 2	Implement various linear data structures like single Linked-list, doubly Linked-list, Circular linked-list.	K_3, K_6
CO 3	Implement Stack and Queue using array and linked list.	K_3
CO 4	Analyze and Implement sorting and searching algorithms	K_4, K_6,
CO 5	Solve complex problems using non-linear data structures like tree and graph.	K_6

List of Experiment:

List of Fundamental Programs

S.N.	Program Title	Category
1	Construct a Code to find the maximum element in an array.	Array
2	Construct a Code to calculate the sum of all elements in an array.	Array
3	Construct a Code to reverse the elements of an array.	Array
4	Construct a Code to check if an array is sorted in ascending order.	Array
5	Construct a Code to count the occurrence of a specific element in an array.	Array
6	Construct a Code creation and traversal of 2D Array in row major and column major order.	Array
7	Construct a code to print the transpose of a given matrix using function	Array
8	Program to find if a given matrix is Sparse or Not and print Sparse Matrix	Array
9	Construct a code to Implement Linear Search	Searching
10	Construct a code to implement Binary Search	Searching
11	Implementation of stack using a list	Stacks
12	Construct a python code to Infix to postfix conversion using a stack	Stacks
13	Construct a code for Balanced parentheses checker using a stack	Stacks
14	Implement Reverse a string using a stack.	Stacks
15	Implement Binary Search using Recursion.	Recursion

16	Construct a python program to print Fibonacci Series using Recursion.	Recursion
17	Queue implementation using a list	Queue
18	Construct a code for Simulating a printer queue using a queue.	Queue
19	Construct a code for Implementing a circular queue.	Queue
20	Implement queue using stack	Queue
21	Create a single linked list and perform basic operations (insertion, deletion, traversal).	Linked List
22	Create a double linked list and perform basic operations (insertion, deletion, traversal).	Linked List
23	Create a circular linked list and perform basic operations (insertion, deletion, traversal).	Linked List
24	Reverse a single linked list.	Linked List
25	Check if a linked list is palindrome.	Linked List
26	Reverse a double linked list.	Linked List
27	Find the middle element of a single linked list.	Linked List
28	Find the middle element of a double linked list.	Linked List
29	Merge two sorted single linked lists.	Linked List
30	Detect and remove a loop in a circular linked list.	Linked List
31	Construct a code to Insert, Delete and search and update a data in Binary Search Tree (BST)	Binary Tree
32	Construct a code for Tree Traversal (Preorder, Inorder, Postorder)	Binary Tree
33	Construct a code Count the number of Leaves in a Binary Tree	Binary Tree
34	Construct a code to find the Height of a Binary Tree	Binary Tree
35	Construct a code to print all Paths from the Root to Leaf Nodes in a Binary Tree	Binary Tree
36	Construct a code to convert a Binary Tree to its Mirror Tree	Binary Tree
37	Construct a code to find the Node with Minimum Value in a Binary Search Tree.	BST

38	Construct a code for Binary Search Tree (BST) Implementation	BST
39	A program to check if a Binary Tree is a Binary Search Tree (BST)	BST
40	Construct a code to check if a Binary Tree is a Balanced Binary Tree	AVL Tree
41	Construct a code to represent graph using adjacency matrix and adjacency list.	Graph
42	Implement BFS and DFS algorithm.	Graph
43	Implement the minimum cost spanning tree.	Graph
44	Implement bubble sort in a non-recursive way.	Sorting
45	Implement selection sort in a non-recursive way.	Sorting
46	Implement insertion sort in a non-recursive way.	Sorting
47	Implement Merge sort in a non-recursive way.	Sorting
48	Implement Merge sort in a recursive way.	Sorting

Textbooks:

1. Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, "Data Structures and Algorithms in Python (An Indian Adaptation)", Wiley Publication
2. Aaron M. Tenenbaum, Yedidyah Langsam and Moshe J. Augenstein, "Data Structures Using C and C++", PHI Learning Private Limited, Delhi India
3. Horowitz and Sahani, "Fundamentals of Data Structures", Galgotia Publications Pvt Ltd Delhi India.
4. Lipschutz, "Data Structures" Schaum's Outline Series, Tata McGraw-hill Education (India) Pvt.ltd

1. **Reference Books:**

1. Thareja, "Data Structure Using C" Oxford Higher Education.
2. AK Sharma, "Data Structure Using C", Pearson Education India
3. P. S. Deshpandey, "C and Data structure", Wiley Dreamtech Publication
4. R. Kruse etal, "Data Structures and Program Design in C", Pearson Education.
5. Berztiss, AT: Data structures, Theory and Practice, Academic Press.
6. AS Tanenbaum, AS Woodhull, Operating Systems Design and Implementation, 3rd Ed., Prentice Hall, 2006.

Course Code	AMICA0401	L	T	P	Credit
Course Title	Cloud Computing	3	1	0	4

Course objective:

- To provide comprehensive knowledge of Cloud Computing concepts, technologies, and applications by introducing and researching state-of-the-art in Cloud Computing fundamental issues, technologies, applications and implementations.

Pre-requisites:

- Adequate knowledge of Basics of Computers along with an online course "Google Cloud Computing Foundation Course", IIT Kharagpur, NPTEL.

Course Contents / Syllabus

UNIT-I	CLOUD COMPUTING AND ITS INFRASTRUCTURE	8 hours

Introduction to Cloud Computing, Definition of Cloud, Evolution of Cloud Computing, Underlying Principles of Parallel and Distributed Computing, Cloud Characteristics, Scalability & Elasticity in Cloud, On-demand Provisioning, Multitenancy, Cloud economics.

UNIT-II	CLOUD VIRTUALIZATION BASICS	8 hours

Basics and need of Virtualization, Types of Virtualizations, Implementation Levels of Virtualization, Virtualization Structures, Tools and Mechanisms, Virtualization of CPU, Memory – I/O Devices, VMM and its types, Virtual Machines, Virtualization tools, Virtualization Support and Disaster Recovery, Resource Provisioning and Resource Provisioning Methods

UNIT-III	SERVICE MODELS AND REFERENCE ARCHITECTURES	8 hours

Service Oriented Architecture, Systems of Systems, Web Services, REST, Publish Subscribe Model, Deployment Model- Public, Private and Hybrid Clouds, IaaS, PaaS, SaaS, Layered Cloud Architecture Design, Challenges and NIST Cloud Computing Reference Architecture, Benefits of CCRA, Architecture Overview – The conceptual Reference Model, Cloud Consumer, Cloud provider, Cloud Auditor, Cloud carrier, Scope of control between Provider and Consumer, IBM's Cloud Computing Reference Architecture (CCRA 2.0).

UNIT-IV	RESOURCE MANAGEMENT	8 hours

Managed and Unmanaged resources in cloud, **Instance Management**- EC2, Azure Virtual Machine, Google Compute Engine. **Storage Services**: Block Storage, Elastic File Storage, Object Storage- S3, RDS, Dynamo DB, Backup, disaster recovery and storage migration. **Network Services**: VPC, Subnets, Routing, Security Groups, DNS, Direct Connect, VPC Endpoints,

UNIT-V	CLOUD SECURITY, MONITORING AND AUDITING	8 hours

Challenges and Objectives; Cloud data life cycle; Common Attacks in Cloud; Security Standard: Confidentiality, Integrity, and Availability (CIA), Authentication and Authorization, Access controls: Role based access controls, multi-factor authentication; Security policy management, IAM; Security Governance and Open Security Architecture; Monitoring and Auditing.

Course outcome: At the end of course, the student will be able to

CO 1	Understand the fundamentals of cloud computing and computing techniques.	K2
CO 2	Understand the concepts of virtualization and its role in cloud service delivery.	K2
CO 3	Discuss various services and architecture of cloud	K4
CO 4	Understand and analyze the management of various cloud resources like instances, storage and network.	K2
CO 5	Analyze the importance of cloud security solutions with monitoring and auditing.	K4

Text books :

1. Ritting house, John W., And James F. Ransome, ―Cloud Computing: Implementation, Management And Security, CRC Press, 2017.
2. Kai Hwang, Geoffrey C. Fox, Jack G. Dongarra, "Distributed And Cloud Computing, From Parallel

Processing To The Internet Of Things", Morgan Kaufmann Publishers, 2013.
3. Raj kumar Buyya, Christian Vecchiola, S. Thamaraiselvi, ―Mastering Cloud Computing, Tata Mcgraw Hill, 2013.

Reference Books:

1. Toby Velte, Anthony Velte, Robert Elsenpeter, "Cloud Computing – A Practical Approach, Tata Mcgraw Hill, 2009.
2. George Reese, "Cloud Application Architectures: Building Applications and Infrastructure In The Cloud: Transactional Systems For EC2 And Beyond (Theory In Practice), O'Reilly, 2009.

Links:

1. https://docs.aws.amazon.com/EC2
2. https://docs.aws.amazon.com/vpc
3. https://docs.aws.amazon.com/vpcEndpoint
4. https://docs.aws.amazon.com/S3
5. https://docs.aws.amazon.com/Security

Course code	AMICA0402	L T P	Credit
Course title	Database System	3 1 0	4
Course objective:			
	The objective of the course is to introduce about database management systems, with an emphasis on how to organize, maintain and retrieve - efficiently, and effectively - information in relational & non-relational databases.		

Pre-requisites:

Fundamental computer knowledge that includes concepts of computer architecture, storage and hardware.

Course Content / Syllabus

UNIT-I	Introduction of Database & Conceptual Designing	7 Hours

Introduction about the DBMS: Introduction of SDLC, Data, Information, Database, DBMS, History of Database, Database system Vs File system, Data Design & Implement the ER Diagram Relational Database term: - Relation, Tuple, Attribute and Domain, Codd Rules Introduction on SQL, Implements the DDL, DML, DCL & TCL: Basic Concept: - Introduction of SDLC, Data, Information, Database, DBMS, History of Database, Database system Vs File system, Data models & Types of Data Models

Relational Database term: - Relation, Tuple, Attribute and Domain, Codd Rules Data Modelling using the Entity Relationship Model: ER model concepts, Degree of relationship, Notation for ER diagram, mapping constraints reduction of an ER diagrams to tables. Extended Entity Relationship Diagram & reduction of EER Introduction on SQL& Types of SQL commands: -DDL, DML, DCL, TCL

Basic of Relation Algebra and Relational calculus.

UNIT-II	Basic of SQL & Normalization	10 Hours

Keys & Types of Keys: - Super key, Candidate Key, Primary Key, Alternative Key Composite Primary key, Foreign Key, unique and Composite Unique key Data Constraint: - Null, Not Null, Default and check Constraint Use of Aggregate Function: -Min (), Max (), Count (), AVG (), Sum ().Uses of String Functions in SQL Uses of mathematical functions in SQL, Uses of Advanced Functions in SQL, Use of Clause: Where, Group by, Having and Order by Functional Dependencies, Normalization & Types of Normalization, Candidate Key, Minimal Cover of FD's.

UNIT-III	Introduction of Complex Queries	5 Hours

Binary Operator: - Cartesian Product, join: -Inner Join: - Natural Join, Equi Join & Non Equi Join

Outer Join: - Left Outer Join, Right Outer Join and Full Outer Join, Division Operator Nested Query or Sub Query: - IN, NOT IN, Exists, Not Exists, All and Any Database connectivity with Java/Python and other Programming Languages.

UNIT-IV	Introduction of PL/SQL and Transaction & Concurrency control concept	8 Hours

Managing Indexes, Synonyms and Sequences, Managing Views, Managing Data in Different Time Zones, Array Function & Operators, Introduction of PL/SQL Implementation of PL/SQL Function, Procedure, Trigger, Cursor Transaction system: - Life cycle of transaction, ACID Properties Schedule & Types of Schedules Control Concurrency Techniques: Concurrency Control, Locking Techniques for concurrency control, 2-phase Locking protocol Transaction & Data Control: - Grant, Revoke, commit & Rollback.

UNIT-V	Introduction of NoSql With MongoDB	10 Hours

Introduction of NoSQL Data Models, Overview of NoSQL Databases With their Types, Uses& Features of NoSQL Document Databases, CAP theorem, BASE Vs ACID Introduction and Features of MongoDB, MongoDB Operators, MongoDB Collection & Document, CRUD operations, MongoDB Shell & their commands, MongoDB Compass, MongoDB Cursor & Methods, Relations in MongoDB, Aggregation in MongoDB, **Introduction of Cloud Database. MongoDB Cloud: -**Stitch, Atlas, Cloud Manager.

Course outcome:

At the end of the course students will be able to

CO 1	Understand ER and EER diagram to design the database for solving the real-world problems.	K3
CO 2	Apply and analyze the Structured Query Language (SQL) to solve the complex queries and implement normalization.	K4
CO 3	Implement the operators in complex queries and apply database connectivity for different applications.	K4
CO 4	Implement PL/SQL, analyze transaction and concurrency control in transaction management.	K4
CO 5	Design and implement relational and non-relational database for the need of the real-world project.	K5

Textbooks

1. Korth, Silbertz, Sudarshan," Database System Concepts", Seventh Edition, McGraw - Hill.
2. Elmasri, Navathe, "Fundamentals of Database Systems", Seventh Edition, Addison Wesley.
3. Ivan Bayross "SQL, PL/SQL The programming language Oracle, Fourth Edition, BPB Publication. (December 1-2010)
4. Brad Dayley "NoSQL with MongoDB in 24 Hours" Sams Publishing; 1st edition (September 8, 2014).

Reference Books

1. Thomas Cannolly and Carolyn Begg, "Database Systems: A Practical Approach to Design, Implementation and Management", Third Edition, Pearson Education, 2007.
2. Raghu Ramakrishan and Johannes Gehrke "Database Management Systems" Third Edition, McGraw-Hill.
3. NoSQL and SQL Data Modeling: Bringing Together Data, Semantics, and Software First Edition by Ted Hills.

Links:

Unit-1

NPTEL Video Course : NOC:Data Base Management System

https://www.youtube.com/watch?v=OWX4RvijwLw

https://www.youtube.com/watch?v=OQanW4NVksY

https://www.youtube.com/watch?v=pm_Tr3eZAac

https://www.youtube.com/watch?v=pBGJYwR5rlM

https://www.youtube.com/watch?v=H6iFrMYZFhU

https://www.youtube.com/watch?v=c5HAwKX-suM

https://www.youtube.com/watch?v=7S_tz1z_5bA

Unit-2

https://www.youtube.com/watch?v=_UZLrD_R0T4

https://www.youtube.com/watch?v=kr4iTckAVUs

https://www.youtube.com/watch?v=FToHXp-IX0g

https://www.youtube.com/watch?v=cwVegKAZO1k

https://www.youtube.com/watch?v=xHB4PeqLK8o

https://www.youtube.com/watch?v=7S_tz1z_5bA

Unit-3

https://www.youtube.com/watch?v=xxBEPiUWGCg

https://www.youtube.com/watch?v=bLL5NbBEg2I
https://www.youtube.com/watch?v=FNYdBLwZ6cE
https://www.youtube.com/watch?v=oRW3PyZi3GE
https://www.youtube.com/watch?v=3aCErW7gMPU
https://www.youtube.com/watch?v=y_YxwyYRJek

Unit-4

https://www.youtube.com/watch?v=X-1viE7QFtQ
https://www.youtube.com/watch?v=5ammL5KU4mo
https://www.youtube.com/watch?v=8yfEl0Yvxto
https://www.youtube.com/watch?v=abLIS6BX964
https://www.youtube.com/watch?v=uuRf-VEFbco
https://www.youtube.com/watch?v=7S_tz1z_5bA

Unit-5

https://www.youtube.com/watch?v=2yQ9TGFpDuM
https://www.youtube.com/watch?v=fbYExfeFsI0
https://www.youtube.com/watch?v=-68k-JS_Y88
https://www.youtube.com/watch?v=c2M-rlkkT5o

Course Code	AMICA0403	L	T	P	Credit
Course Title	Software Engineering and Design	3	0	0	3

Course objective:

Students will be able to apply the principles of analysis, design, development, test, and maintenance in systematic way to create and build cost effective software solutions and become a successful professional with good fundamental knowledge of software engineering.

Pre-requisites:

- The student must understand basic computer terminology.
- The student must have knowledge of some programming language..

Course Contents / Syllabus

UNIT-I	Introduction	8 hours

Evolving role of software, Software Characteristics, Software crisis, silver bullet, Software myths, Software Engineering Phases, Team Software Process (TSP), emergence of software engineering, Software process, project and product. Development models: Software Process Models: Waterfall Model, Prototype Model, Spiral Model, Iterative Model, Incremental Model, Agile Methodology: Scrum Artifacts, Scrum Roles and Scrum Events, Kanban framework.

UNIT-II	Software Requirement Specifications	8 hours

Software Requirement Specifications (SRS): Requirement Engineering Process: Elicitation, Analysis, Documentation, Review and Management of User Needs, Feasibility Study, Information Modelling, Use Case Diagram, Data Flow Diagrams, Entity Relationship Diagrams, Decision Tables, SRS Document, IEEE Standards for SRS. Software Quality Assurance (SQA): Quality concepts, SQA activities, Formal approaches to SQA; Statistical software quality assurance; CMM, The ISO standard.

UNIT-III	Software Design	8 hours

Software Design: Design principles, the design process; Design concepts: refinement, modularity: Cohesion, Coupling, Effective modular design: Functional independence, Design Heuristics for effective modularity, Software architecture: Function Oriented Design, Object Oriented Design: OOPs concepts-Abstraction, object, classification, inheritance, encapsulation, UML Diagrams-Class Diagram, Interaction diagram, Activity Diagram, control hierarchy: Top-Down and Bottom-Up Design, structural partitioning, software procedure.

UNIT-IV	Software Testing	8 hours

Testing Objectives, 7 Principals of Testing, Levels of Testing: Unit Testing, System Testing, Integration Testing, User Acceptance Testing, Regression Testing, testing for Functionality and Testing for Performance, Top Down and Bottom-Up Testing Strategies: Test Drivers and Test Stubs, Accessibility Testing, Structural Testing (White Box Testing), Functional Testing (Black Box Testing), Test Data Suit Preparation, Alpha and Beta Testing of Products. Functional Testing (DAO, BO) Static Testing Strategies: Formal Technical. Reviews (Peer Reviews), Walk Through, Code Inspection, Compliance with Design and Coding Standards, Test Management, Test Planning and Estimation, Test Monitoring and Control, Configuration Management, Risks and Testing, Defect Management, Tool Support for Testing, Effective Use of Tools.

UNIT-V	Project Maintenance and management concepts	8 hours

Project maintenance and management concepts: Project management concepts, Planning the software project, Estimation: Software Measurement and Metrics, Various Size Oriented Measures-LOC based, FP based, Halestead's Software Science, Cyclomatic Complexity Measures: Control Flow Graphs, Use-case based, empirical estimation COCOMO- A Heuristic estimation techniques, staffing level estimation, team structures, risk analysis and management. Configuration Management, Software reengineering reverse engineering, restructuring forward engineering, Clean Room software engineering. Case Tools, Software Maintenance: Preventive, Corrective and Perfective Maintenance, Cost of Maintenance, Need of Maintenance.

Course Outcome

CO 1	Understand various software characteristics and analyze different software Development Models.	K2
CO 2	Demonstrate the contents of an SRS and ensure that analysis, design and development meet applicable standards.	K2
CO 3	Compare and contrast various methods for software design and create various object-oriented diagrams.	K4
CO 4	Apply testing strategies for software systems, apply various testing techniques such as unit testing, test driven development and functional testing.	K3
CO 5	Apply the project management concepts and calculate various metrics related to software project.	K3

Reference:

3. Pankaj Jalote, Software Engineering, Wiley. (1 January 2010)
4. Ghezzi, M. Jarayeri, D. Manodrioli, Fundamentals of Software Engineering, PHI Publication. 2nd Edition. (1 January 2007)
5. Kassem Saleh, "Software Engineering", Cengage Learning. (2009)
4. Ian Summerville, Software Engineering, Addison Wesley. 9th Edition. (29 October 2017)

Textbooks:

1. KK Aggarwal and Yogesh Singh, Software Engineering, New Age International Publishers 3RD Edition (December 11, 2008)
2. RS Pressman, Software Engineering: A Practitioners Approach, McGraw Hill. 7th Edition (14-Jan-2022)
3. Rajib Mall, Fundamentals of Software Engineering, PHI Publication.4th Edition. (1 January 2014)

Link:

- https://www.mlsu.ac.in/econtents/16_EBOOK-7th_ed_software_engineering_a_practitioners_approach_by_roger_s._pressman_.pdf
- https://davcollegetitilagarh.org/wp-content/uploads/2020/09/fundamentals-of-software-engineering-fourth-edition-rajib-mall.pdf
- https://handoutset.com/wp-content/uploads/2022/05/An-Integrated-Approach-to-Software-Engineering-Pankaj-Jalote.pdf
- https://nptel.ac.in/courses/106105182

Course Code	AMICA0404	L	T	P	Credit
Course Title	Design Thinking-II	2	1	0	3
Course Objective: Objective of this course is to:		Duration: 32 Hours			
	The objective of this course is to upgrade Design Thinking skills by learning & applying advanced and contextual Design Thinking Tools. It aims to solve a Real-Life Problem by applying Design Thinking to create an impact for all the stakeholders.				

Pre-requisites: Student must complete Design Thinking-I course.

Course Contents / Syllabus

UNIT-I	INTRODUCTION	10 Hours

Design thinking & Innovation, Design Thinking Mindset and Principles, recap of 5-Step Process of Design Thinking, Design Approaches, additional in-depth examples of each design approaches. Simon Sinek's – Start with Why, The Golden Circle, Asking the "Why" behind each example (an in-class activity of asking 5-WHYS),

The Higher Purpose, in-class activity for LDO & sharing insights Visualization and its importance in design thinking, reflections on wheel of life (in-class activity for visualization & Wheel of Life), Linking it with Balancing Priorities (in class activity), DBS Singapore and Bank of Americas' Keep the Change Campaign. Litter of Light & Arvind Eye Care Examples, understanding practical application of design thinking tools and concepts, case study on McDonald's Milkshake / Amazon India's Rural Ecommerce & Gillette.

Working on 1-hour Design problem, Applying RCA and Brainstorm on innovative solutions.

Main project allocation and expectations from the project.

UNIT-II	REFINEMENT AND PROTOTYPING	8 Hours

Refine and narrow down to the best idea, 10-100-1000gm, QBL, Design Tools for Convergence – SWOT Analysis for 1000gm discussion. In-class activity for 10-100-1000gm & QBL.

Prototyping (Convergence): Prototyping mindset, tools for prototyping – Sketching, paper models, pseudo-codes, physical mockups, Interaction flows, storyboards, acting/role-playing etc, importance of garnering user feedback for revisiting Brainstormed ideas, Napkin Pitch, Usability, Minimum Viable Prototype, Connecting Prototype with 3 Laws, A/B Testing, Learning Launch.

Decision Making Tools and Approaches – Vroom Yetton Matrix, Shift-Left, Up, Right, Value Proposition, Case study: Career buddy, You-Me-Health Story & IBM Learning Launch. In-class activities on prototyping- paper-pen / physical prototype/ digital prototype of project's 1000gm idea

UNIT-III	STORYTELLING, TESTING AND ASSESSMENT	8 Hours

Storytelling: Elements of storytelling, Mapping personas with storytelling, Art of influencing, Elevator Pitch, Successful Campaigns of well-known examples, in-class activity on storytelling. Testing of design with people, conducting usability test, testing as hypothesis, testing as empathy, observation and shadowing methods, Guerrilla Interviews, validation workshops, user feedback, record results, enhance, retest, and refine design, Software validation tools, design parameters, alpha & beta testing, Taguchi, defect classification, random sampling.

Final Project Presentation and assessing the impact of using design thinking

UNIT-IV	INNOVATION, QUALITY AND LEADERSHIP	6 Hours

Innovation: Need & Importance, Principles of innovations, Asking the Right Questions for innovation, Rationale for innovation, Quality: Principles & Philosophies, Customer perception on quality, Kaizen, 6 Sigma. FinTech case study of Design Thinking application – CANVAS

Leadership, types, qualities and traits of leaders and leadership styles, Leaders vs Manager, Personas of Leaders & Managers, Connecting Leaders-Managers with 13 Musical Notes, Trait theory, LSM (Leadership Situational Model), Team Building Models: Tuckman's and Belbin's. Importance of Spatial elements for innovation

UNIT-V	**UNDERSTANDING HUMAN DESIRABILITY**	**8 Hours**

Comprehensive human goal: the five dimensions of human endeavour (Manaviya - Vyavstha) are: Education-Right living (Sikhsa- Sanskar), Health – Self-regulation (Swasthya - Sanyam), Justice – Preservation (Nyaya-Suraksha), Production – Work (Utpadan – Karya), Exchange – Storage (Vinimya – Kosh), Darshan-Gyan-Charitra (Shifting the Thinking)

Interconnectedness and mutual fulfilment among the four orders of nature recyclability and self-regulation in nature, Thinking expansion for harmony: Self-exploration (Johari's window), group behaviour, interpersonal behaviour and skills, Myers-Briggs personality types (MBTI), FIRO-B test to repair relationships.

Course outcome: At the end of the course, the student will be able to:

CO 1	Learn sophisticated design tools to sharpen their problem-solving skills	K2
CO 2	Construct innovate ideas using design thinking tools and converge to feasible idea for breakthrough solution	K6
CO 3	Implement storytelling for persuasive articulation	K3
CO 4	Understanding the nature of leadership empowerment	K2
CO 5	Understand the role of a human being in ensuring harmony in society and nature	K2

Textbooks

1. Arun Jain, UnMukt: Science & Art of Design Thinking, 2020, Polaris "Operating System: A Design-oriented Approach", Charles Patrick Crowley, 9th Edition, 2017
2. Gavin Ambrose and Paul Harris, Basics Design 08: Design Thinking, 2010, AVA Publishing SA "Design of the UNIX Operating Systems", Maurice J. Bach., 1st Edition, 2015
3. R R Gaur, R Sangal, G P Bagaria, A Foundation Course in Human Values and Professional Ethics, First Edition, 2009, Excel Books: New Delhi.

Reference Books

1. Jeanne Liedta, Andrew King and Kevin Benett, Solving Problems with Design Thinking – Ten Stories of What Works, 2013, Columbia Business School Publishing
2. Dr Ritu Soryan, Universal Human Values and Professional Ethics, 2022, Katson Books
3. Vijay Kumar, 101 Design Methods: A Structured Approach for Driving Innovation in Your Organization, 2013, John Wiley and Sons Inc, New Jersey
4. Roger L. Martin, Design of Business: Why Design Thinking is the Next Competitive Advantage, 2009, Harvard Business Press, Boston MA
5. Tim Brown, Change by Design, 2009, Harper Collins
6. Pavan Soni, Design your Thinking: The Mindsets,Tool sets and Skill Sets for Creative Problem-Solving, 2020, Penguin Books.

Links:

Unit I

- https://www.youtube.com/watch?v=6_mHCOAAEI8
- https://nptel.ac.in/courses/110106124
- https://designthinking.ideo.com/
- https://blog.experiencepoint.com/how-mcdonalds-evolved-with-design-thinking

Unit II

- https://www.coursera.org/lecture/uva-darden-design-thinking-innovation/the-ibm-story-iq0kE
- https://www.coursera.org/lecture/uva-darden-design-thinking-innovation/the-meyouhealth-story-part-i-what-is-W6tTs
- https://onlinecourses.nptel.ac.in/noc19_mg60/preview

Unit III

- https://nptel.ac.in/courses/109/104/109104109/
- https://www.d-thinking.com/2021/07/01/how-to-use-storytelling-in-design-thinking/

Unit IV

- https://www.worldofinsights.co/2020/10/infographic-8-design-thinking-skills-for-leadership-development/

Unit V

- https://www.youtube.com/watch?v=hFGVcx1Us5Y

Course Code	**AMICA0452**	**L**	**T**	**P**	**Credit**
Course Title	**Database Management System Lab**	**0**	**0**	**4**	**2**

Suggested list of Experiment

Course outcome: At the end of course, the student will be able to

CO 1	Understand ER and EER diagram to design the database for solving the real-world problems.	K_2, K_3
CO 2	Apply and analyze the Structured Query Language (SQL) to solve the complex queries and implement normalization.	K_3, K_6
CO 3	Implement the operators in complex queries and apply database connectivity for different applications.	K_2
CO 4	Implement PL/SQL and analyze transaction and concurrency control in transaction management.	K_3
CO 5	Design and implement relational and non-relational database for the need of the real-world project.	K_3, K_4

List of Experiment:

List of Fundamental Programs

S.N.	Program Title	Category
1	Understand and implement the different ER diagram notation with their relationship and Cardinalities.	ER Diagram Notation
2	Write a program to implement default constructor, parameterized constructor, and destructor.	Create ER Diagram-1
3	Design an ER diagram for a travel agency that includes entities such as travellers, bookings, destinations, and itineraries. also implement the relationship and cardinalities between the entities with their relevant attribute.	Create ER Diagram-2
4	Converting Company & Travel Agency ER Model to Relational Model (Represent entities and relationships in tabular form, represent attributes as columns, identifying keys).	Reduction of ER Diagram1 & 2
5	Each students create at least one ER & EER diagram from real world problem and convert in tabular from with all needed constraint.	Exercise -1
6	Implement DDL and DML commands	DDL, DML Commands
7	Implement DCL &TCL commands	DDL, DML Commands
8	1. Create Database, Rename Database, Delete Database in relational database tool. 2. Create table employee with attributes Emp_no<datatype><size> E_name<datatype><size> JOB <datatype><size> Address <datatype><size> Salary<datatype><Size> 3. Insert data into the table 4. Implementation of select command 5. Implementation of update command	Exercise-2

	6. Implementation of alter command 7. Implementation of delete command 8. Implementation of rename command. 9.Implementation of rollback command 10.Implementation of commit Command 11.Implemenation of Truncate Command 12. Implementation of Drop Command	
9	Implementation of I/O Constraint: Primary Key, composite primary key, Foreign Key with on delete set null and on delete set null constraint	Key Constraints
10	Implementation of constraint: Unique Key and Composite unique key and uses Unique key as foreign key.	Key Constraints
11	Implementation of Business Constraint: Null, Not Null, Default, Check	Business Constraints
12	Implement and apply the different form of normalization approach on company /Travel Agency Database.	Normalization
13	Reduction & Implementation in SQL for ER Diagram of Company Database: - Create table for EMPLOYEE, DEPARTMET, PROJECT, DEPENDENTS and WORK_ON with all needed keys and other constraints. Populated all table with atleast Ten records in each table as per as applied constraints.	Case Study-1
14	Practicing Queries using Like, Between, Aliases, distinct Operator & Predicate.	Predicate & Operators
15	Implementation of Aggregate Functions.	Aggregate Functions
16	Implementation of Scalar, Mathematical and Advanced functions	String and Advanced Functions
17	Implementation of Queries using Where, Group by, Having and Order by Clause.	Clause
18	Implementation and uses of clause and operators on Company/ Travel Agency or Other database. A. Find the name of employee whose name start with A. B. Find the name of employee where 'hi' in any position. C. Find the name of employee whose 'r' have in second position. D. Find the details of employee whose salary is less than 70000. E. Find the name of employee whose name start with V and end with l. F. Find the average salary of each department G. Find the max salary of each department H. Find the sum of salary of department that have more than three employees in ascending order. I. Find the empid of Employee who work in more than 3 project. J. Find the empid who have more than one dependent. K. Implement the concept of rollback and commit on Employee Table	Exercise: -3
19	Create a table EMPLOYEE with following schema:-(Emp_no, E_name, E_address, E_ph_no, Dept_no, Dept_name,Job_id, Designation, Salary) Write SQL statements for the following query. 1. List the E_no, E_name, Salary of all employees working for MANAGER. 2. Display all the details of the employee whose salary is more than the Salary of any IT Professor.	Execrise-4

	3. List the employees in the ascending order of Designations of those joined after 1981. 4. List the employees along with their Experience and Daily Salary. 5. List the employees who are either 'CLERK' or 'ANALYST' 6. List the employees who joined on 1-MAY-81, 3-DEC-81, 17-DEC-81,19-JAN-80. 7. List the employees who are working for the Deptno 10 or 20. 8. List the Enames those are starting with 'S'. 9. Display the name as well as the first five characters of name(s) starting with 'H' 10. List all the emps except 'PRESIDENT' & 'MGR" in ASC order of Salaries. 11. Display total salary spent for each job category. 12. Display lowest paid employee details under each manager. 13. Display number of employees working in each department and their department name. 14. Display the details of employees sorting the salary in increasing order. 15. Show the record of employee earning salary greater than 16000 in each department. 16.Add constraints to check, while entering the empno value (i.e) empno > 100. 17. Define the field DEPTNO as unique. 18. Create a primary key constraint for the column (EMPNO).	
20	Implementation of Queries using set theory operators UNION, INTERSECT, MINUS.	Set Theory Operators
21	Implementation of Queries using Inner Join: - Natural Join, Equi Join & Non Equi Join	Join Operators
22	Implementation of Queries using Outer Join: - Left Outer Join, Right Outer Join and Full Outer Join	Join Operators
23	Implementation of Queries nested Queries or Sub Queries: - IN, NOT IN, Exists, Not Exists, All and Any.	Nested Queries
24	Apply the set theory operators, join's and nested queries on company database (Case Study-1) Write the SQL Queries for the following statement (a)Retrieve the names of employees in department 5 who work more than 10 hours per week on the 'ProductX' project. (b) List the names of employees who have a dependent with the same first name as themselves. (c) Find the names of employees that are directly supervised by 'Franklin Wong'. (d) For each project, list the project name and the total hours per week (by all employees) spent on that project. (e) Retrieve the names of all employees who work on every project controlled by department 5. (f) Retrieve the names of all employees who do not work on any project. (f') Retrieve the names of all employees who do not work on every project (g) For each department, retrieve the department name, and the average salary of employees working in that department. (h) Retrieve the average salary of all female employees. (i) Find the names and addresses of all employees who work on at least one project located in Houston but whose department has no location in Houston. (j) List the last names of department managers who have no dependents. (k)Retrieve the names of all employees who work in the department that has the employee with the highest salary among all employees. (l) Retrieve the names of all employees whose supervisor's supervisor has	Exercise -5

	'888665555' for Ssn. (m) For each department that has more than 5 employees retrieve the dno and no. of its employees who are making more than 6,00,000 (n)Find the sum of salaries of all employees of 'ACCOUNTS' department as well as the MAX(SAL), MIN(SAL), AVG(SAL) in this department (o)Show the resulting salary for employee working on IOT project is given a 10% raise	
25	**Requirement: -** A college consists of number of employees working in different departments. In this context, create two tables' employee and department. Employee consists of columns empno, empname, basic, hra, da, deductions, gross, net, date-of-birth. The calculation of hra,da are as per the rules of the college. Initially only empno, empname, basic have valid values. Other values are to be computed and updated later. Department containsdeptno, deptname, and description columns. Deptno is the primary key in department table and referential integrity constraint exists between employee and department tables. Perform the following operations on the database: 1. Create tables department and employee with required constraints. 2. Initially only the few columns (essential) are to be added. Add the remaining columns separately by using appropriate SQL command 3. Basic column should not be null. 4. The default value for date-of-birth is 1 jan, 1990. 5. When the employees called daily wagers are to be added the constraint that salary should be greater than or equal to 5000 should be dropped. 6. Display the information of the employees and departments with description of the fields. 7. Display the average salary of all the departments. 8. Display the average salary department wise. 9. Display the maximum salary of each department and also all departments put together. 9. Commit the changes whenever required and rollback if necessary. 10. Find the employees whose salary is between 5000 and 10000 but not exactly 7500. 11. Find the employees whose name contains 'en'. 12.Create alias for columns and use them in queries. 13. List the employees according to ascending order of salary. 14. List the employees according to ascending order of salary in each department. 15. Find the employees who are born on Feb 29. 16. Find the departments where the salary of at-least one employee is more than 20000. 17. Find the departments where the salary of all the employees is less than 20000. 18. Add the column dept_location in department table.	Exercise -6
26	Understand & implement the Database Connectivity with Java/Python etc programming language	Database Connectivity
27	1. Implementation and apply all the set theory operators, join and nested queries concept on Case study -1. A. Make a list of all project members for projects that involve an employee whose name is SCOTT either as a worker or as a manager of the department that controls the project. B. To retrieve the Social Security numbers of all employees who either work in department 5 or directly supervise an employee who works in department 5. C. To retrieve the SSN of all employees who work as a supervisor not a manager.	Exercise -7

	D. To retrieve the SSN of all employees who work as a supervisor and also manage the department. E. We want to retrieve a list of names of each female employee's dependents F. We want a list of all employee names as well as the name of the departments they manage if they happen to manage a department; if they do not manage one, we can indicate it with a NULL value. G. Retrieve the names of employees who have no dependents. H. List the names of all employees with two or more dependents. I. List the names of managers who have at least one dependent. J. Retrieve the names of all employees who do not have supervisors. K. Retrieve the name of each employee who has a dependent with the same first name and is the same sex as the employee. 2. Create Standalone Application/Web application with populated the data by any database.	
28	Implementation of Array Function	Array Functions
29	Implementation of Array Operators	Array Functions
30	Implementation of Indexing, Views and sequence	Index, Views
31	Write a PL/SQL Program to Add Two Numbers Write PL/SQL Program for Fibonacci Series Write PL/SQL Program to Find Greatest of Three Numbers	PL/SQL Basic
32	Write a Pl/SQL code block to calculate the area of a circle for a value of radius varying from 3 to 7. Store the radius and the corresponding values of calculated area in an empty table named Areas, consisting of two columns Radius and Area.	PL/SQL Procedure
33	Write a PL/SQL code block that will accept an account number from the user, check if the users balance is less than the minimum balance, only then deduct Rs.100/- from the balance.	PL/SQL Procedure
34	Create a row level trigger for the customers table that would fire for INSERT or UPDATE or DELETE operations performed on the CUSTOMERS table. This trigger will display the salary difference between the old values and new values:	PL/SQL Trigger
35	Implementation of commit and rollback statement with amount transfer example.	Transaction
36	Implementation array, indexing, transaction concept on Case study 1. 1.Implementation of Array Functions & Operators 2. Implementation of Sequence -Creating Sequences -Modifying a Sequence Definition -Removing Sequences 3. Implementation of Views -Creating Simple and Complex Views -Modifying Views -Removing Views 4. Implementation of Indexes -Manual and Automatic Indexes -Creating Indexes - Removing Indexes	Execrise-8
37	A. Write a PL/SQL block to calculate the incentive of an employee whose ID is 110. B. Grant and revoke DCL command used on Employee table -GRANT SELECT ON Employee TO emp_name;	Execrise-09

	-Granting multiple privileges on Employee table -Granting all privileges on Employee table; -Granting privilege to a role in Employee table -Granting the WITH GRANT OPTION on Employee table. -Revoke all the permission applied on Employee table. C Create the CUSTOMERS table having the following attributes: - - (ID, NAME, AGE, ADDRESS, SALARY) - Insert ten records in customer table. -In Customer table delete those records which have age = 25 and then COMMIT the changes in the database. -In Customer table delete those records which have age = 30 and then Rollback the changes in the database. - Create three save point for customer table in that the three deletions have taken place. - Apply the save point 2 with rollback on customer table and display the table record. - Apply the SET Transnation command.	
38	Study of Open Source NOSQL Database and installation of MongoDB	Installation of MongoDB
39	Create, drop, rename the database in MongoDB	MongoDB Database
40	Implementation the MongoDB Operators.	MongoDB Operators
41	Implementation the CRUD Operation in MongoDB	MongoDB CRUD Operations
42	Implementation of the MongoDB Shell commands	MongoDB Shell Commands
43	Implementation of MongoDB Cursor and their methods	MongoDB Cloud Commands
44	Implementation of relation in MongoDB	Relation in MongoDB
45	Implementation of Aggregate in MongoDB	Aggregate in MongoDB
46	Implementation of all CRUD operation, Cursor and aggregate etc on real world problem. Connect to MongoDB (by using mongo shell) A. Create database with name (ems) - use ems; B. Create collection with following fields: - {"name,age",gender","exp",subjects,"type""qualification"}, C. Insert the Ten documents into "faculty" collection (Use insertMany()) Write the following queries: - 1. Get the details of all the faculty. 2. Get the count of all faculty members. 3. Get all the faculty members whose qualification is “Ph.D”.	Exercise - 10

	4. Get all the faculty members whose experience is between 8 to 12 years. 5. Get all the faculty members who teach “MATHS” or “NETWORKING”. 6. Get all the faculty members who teach “MATHS” and whose age is more than 30 years and qualification must be “Ph.D”. 7. Get all the faculty members who are working part-time or who teach “JAVA”. 8. Add the following new faculty members: {"name": "Ankita ", "age":34,"gender":"F","exp":25, subjects: ["MATHS","DE"],"type":"Full Time", "qualification": "Ph.D"} 9. Update the data of all faculty members by incrementing their age and exp by one year. 10. Update the faculty “Sivani” with the following data: update qualification to “Ph.D” and type to “Full Time”. 11. Update all faculty members who are teaching “DBMS” such that they should now also teach “Java Programming”. 12. Delete all faculty members whose age is more than 55 years. 13. Get only the name and qualification of all faculty members. 14. Get the name, qualification and exp of all faculty members and display the same in ascending order of exp. 15. Sort the faculty details by their age (descending order) and get the details of the first five faculty members only.	
47	Mini project (Design & Development of Data and Application) for following: - 1. Analyzing Sales Data 2. Customer Segmentation 3. International Debt Statistics Analysis 4. Analyze the World Population 5. House Property Sales Analysis 6. Sentiment Analysis 7. Health care organization database 8. Blood donation system database 9. Art gallery management database 10. ATM management system database 11. Face detection 12. Evaluation of academic performance 13. Mobile wallet with merchant payment 14. Public news droid 15. Crime rate prediction 16. Twitter Sentiment Analysis 17. Election Analysis 18. Smart Farming used whether forecasting 19. Speech to Text 20. Automated Patient and Doctor Handling System 21. Web Scraping Using Beautiful Soup 22. Movie recommendation system 23. Online examination and evaluation system 24. Book Publishing Company	Mini Project & applications
48	Implementation of case Study on different domain 1. E-commerce Platform 2. Inventory Management	Case Study on domain wise

	3. Railway System 4. Hospital Data Management 5. Voice-based Transport Enquiry System 6. SMS-based Remote Server Monitor system 7. Banking System	

Course Code	AMICA0455	L	T	P	Credit
Course Title	Object Oriented Techniques using JAVA Lab	0	0	8	4

Course objective:

- The objective of this course is to understand the object-oriented methodology, and its techniques to design stand alone and GUI applications using hands-on engaging activities.

Pre-requisites:

- The student must understand basic computer terminology.
- The student must have knowledge of some programming language.

Course Contents / Syllabus

UNIT-I	Basics of Java Programming	8 hours

Introduction and Pillars of OOP with real life example, JVM architecture and its components. Modelling Concepts: Introduction, Class Diagram and Object Diagram, UML concepts: Association, Composition, aggregation, realization, and Generalization Control Statements: Decision Making, Looping and Branching, Argument Passing Mechanism: Command Line Argument, Console Input Class and Object: Object Oriented Concept in Java ,Object Reference, Constructor, Abstraction: Abstract Class, Interface and its uses, Defining Methods, Use of "this" and "super" keyword, Garbage Collection and finalize () Method etc.

UNIT-II	OOPs features, arrays and lambda expressions	8 hours

Inheritance: Introduction and Types of Inheritance in Java, Implementing Multiple Inheritance, Access Modifiers, Constructors and super constructor in Inheritance Polymorphism: Introduction and Types, Overloading and Overriding Lambda expression: Introduction and Working with Lambda Variables Arrays: Introduction and its Types.

UNIT-III	Packages, Exception Handling and String Handling	8 hours

Packages: Introduction and Types, Access Protection in Packages, Import and Execution of Packages Exception Handling, Assertions and Localizations Introduction and Types, Exceptions vs. Errors, Handling of Exception Finally, Throws and Throw keyword, Multiple Catch Block, Nested Try and Finally Block, Tokenizer Assertions and Localizations Concepts and its working String Handling: Introduction and Types, Operations, Immutable String, Method of String class, String Buffer and String Builder class.

UNIT-IV	Concurrency in Java and I/O Stream	8 hours

Threads: Introduction and Types, Creating Threads, Thread Life-Cycle, Thread Priorities, Daemon Thread, Runnable Class, Synchronizing Threads etc. I/O Stream: Introduction and Types, Common I/O Stream Operations, Interaction with I/O Streams Classes Annotations: Introduction, Custom Annotations and Applying Annotations.

UNIT-V	GUI Programming, Generics and Collections	8 hours

GUI Programming: Introduction and Types, Swing, Components and Containers, Layout Managers and User- Defined Layout and Event Handling Generics: Introduction to Generic Classes, Initializing a Generic Object, Generic Cell Driver Class, Generic Methods, Use enumerated type Collections: Introduction, Using Method References, Using Wrapper Class, Using Lists, Sets, Maps and Queues, Collection using Generics, Iterators.

Course outcome:	At the end of the course, the student will be able to	
CO 1	Understand the concepts of object-oriented programming and relationships among them needed in modelling.	K_2, K_3
CO 2	Demonstrate the Java programs using OOP principles and implement the concepts of lambda expressions.	K_3, K_6

CO 3	Analyse packages with different protection level resolving namespace collision and implement the error handling concepts for uninterrupted execution of Java program.	K_2
CO 4	Implement Concurrency control, I/O Streams and Annotations concepts by using Java program.	K_3
CO 5	Design and develop the GUI based application, Generics and Collections in Java programming language to solve the real-world problem.	K_3, K_4

Reference:

1. Cay S. Horstmann, "Core Java Volume I – Fundamentals", Prentice Hall AK Sharma, "Data Structure Using C", Pearson Education India.
2. Joshua Bloch," Effective Java", Addison Wesley.
3. Herbert Schildt," Java - The Complete Reference", McGraw Hill Education 12th edition.

Text books :

1. Herbert Schildt," Java: A Beginner's Guide", McGraw-Hill Education 2nd edition
2. E Balagurusamy, "Programming with Java A Primer", TMH, 4th edition

Link:

- https://www.youtube.com/watch?v=r59xYe3Vyks&list=PLS1QulWo1RIbfTjQvTdj8Y6yyq4R7g-Al
- https://www.youtube.com/watch?v=ZHLdVRXIuC8&list=PLS1QulWo1RIbfTjQvTdj8Y6yyq4R7g-Al&index=18
- https://www.youtube.com/watch?v=hBh_CC5y8-s
- https://www.youtube.com/watch?v=qQVqfvs3p48
- https://www.youtube.com/watch?v=2qWPpgALJyw

List of Experiment:

List of Fundamental Programs

S.N.	Program Title	Category
1	Understanding Text Editors to Write Programs Compile and run first java file Byte Code and class file	Setting class path variables, Compilation of java file and execute its byte code.
2	Sketch a class and object diagram describing the sales order system of restaurant	Designing object and class diagram with UML concepts.
3	Sketch a class diagram describing the circle and rectangle class	Designing object and class diagram with UML concepts.
4	Sketch a class diagram for a college platform including, classroom, playground, chair, table, smart board, teaching staff etc.	Designing object and class diagram with UML concepts.
5	Sketch a class diagram containing class called Employee, which models an employee with an ID, name and salary. Add method raise Salary(percent) that increases the salary by the given percentage.	Designing object and class diagram with UML concepts.

6	Program to display default value of all Primitive data types	Data Types
7	Implement the code using main() method to calculate and print the Total and Average marks scored by a student from the input given through the command line arguments. Assume that four command line arguments name, marks1, marks2, marks3 will be passed to the main() method in the below class with name Total And AvgMarks.	Command Line Arguments
8	Write code which uses if-then-else statement to check if a given account balance is greater or lesser than the minimum balance. Write a class Balance Check with public method check Balance that takes one parameter balance of type double. Use if-then-else statement and print Balance is low if balance is less than 1000. Otherwise, print Sufficient balance.	Conditional Statement
9	A class Number Palindrome with a public method is Number Palindrome that takes one parameter number of type int. Write a code to check whether the given number is palindrome or not. For example, Cmd Args : 333 333 is a palindrome	Conditional Statement and Loops
10	Write a class Fibonacci Series with a main method. The method receives one command line argument. Write a program to display Fibonacci series i.e. 0 1 1 2 3 5 8 13 21	Conditional Statement and Loops
11	Write a Java Program to find the Factorial of a given number.	Conditional Statement and Loops
12	Java Program to create a class, methods and invoke them inside main method.	Class and Object
13	Write a Java program to illustrate the abstract class concept. Create an abstract class Shape, which contains an empty method number Of Sides(). Define three classes named Trapezoid, Triangle and Hexagon extends the class Shape, such that each one of the classes contains only the method number Of Sides(), that contains the number of sides in the given geometrical figure. Write a class Abstract Example with the main() method, declare an object to the class Shape, create instances of each class and call numberOfSides() methods of each class.	abstract class
14	Java program to illustrate the static field in the class.	‘static’ keyword
15	Java Program to illustrate static class.	‘static’ keyword
16	Write a java program to access the class members using super keyword	‘super’ keyword
17	Java program to access the class members using this keyword	‘this’ keyword
18	Implement an interface named MountainParts that has a constant named TERRAIN that will store the String value “off_road”. The interface will define two methods that accept a String argument name newValue and two that will return the current value of an instance field. The methods are to be named: getSuspension, setSuspension, getType , setType.	Java interface
19	Java program to demonstrate nested interface inside a interface.	Java Interface
20	Java program to demonstrate nested interface inside a class.	Java Interface
21	Java program to explicit implementation of garbage collection by using finalize() method	Garbage Collection and

		finalize() method
22	JAVA program to implement Single Inheritance	Concepts of inheritance
23	JAVA program to implement multi-level Inheritance	Concepts of inheritance
24	JAVA program to implement constructor and constructor overloading.	Constructor and Inheritance
25	JAVA program implement method overloading.	Overloading and Overriding
26	JAVA program to implement method overriding.	Overloading and Overriding
27	Java program to implement lambda expression without parameter.	Lambda Expression
28	Java program to implement lambda expression with single parameter.	Lambda Expression
29	Java program to implement lambda expression with multi parameter.	Lambda Expression
30	Java program to implement lambda expression that iterate list of objects	Lambda Expression
31	Java program to define lambda expressions as method parameters	Lambda Expression
32	Write a class Count Of Two Numbers with a **public** method compare Count Of that takes three parameters one is arr of type int[] and other two are arg1 and arg2 are of type int and returns true if count of arg1 is greater than arg2 in arr. The return type of compare Count Of should be boolean. Assummptions: • arr is never null • arg1 and arg2 may be same	Arrays
33	JAVA program to show the multiplication of two matrices using arrays.	Arrays
34	Java Program to search an element using Linear Search	Array Searching
35	Java program to search an element using Binary Search	Array Searching
36	Java Program to sort element using Insertion Sort	Array Sorting
37	Java Program to sort element using Selection Sort – Largest element Method	Array Sorting
38	Java program to Sort elements using Bubble Sort	Array Sorting
39	Java program to create user defined package.	Java Package
40	Java Program to create a sub- classing of package	Java Package
41	Implement the following: 1. Import package*; 2. import package.classname; 3. Using fully qualified name.	Java Package

Lab Code	AMICA0453	L	T	P	Credit
Lab Title	Project Based on Software Engineering & Design Lab	0	0	4	2

Course outcome: At the end of the course, the student will be able to

CO 1	Identify ambiguities, inconsistencies, and incompleteness from a requirements specification and state functional and non-functional requirement	K_2, K_3
CO 2	Graphically represent various UML diagrams, and associations among them.	K_3, K_6
CO 3	Able to use modern engineering tools for specification, design, implementation and testing	K_2

List of Experiment:

List of Fundamental Programs

S.No.	Program Title	Category
1.	Find the real-world problem and create the requirement statements.	Requirement Gathering
2.	Draw the use case diagram for assigned project.	Requirement Engineering
3.	Draw the Data Flow Diagram (DFD): All levels.	Requirement analysis
4.	Design an ER diagram for with multiplicity.	Requirement analysis
5.	Prepare SRS document in line with the IEEE recommended standards.	Requirement analysis
6.	Create Flowchart diagram for the assigned project	Design
7.	Create Object diagram for the assigned project	Object oriented design
8.	Create Class diagram for the assigned project.	Object oriented design
9.	Create State chart diagram assigned project.	Software design
10.	Create Interaction diagram: sequence diagram.	Software design
11.	Create Interaction diagram: collaboration diagram.	Software design
12.	Create Activity diagram for the assigned project.	Software design
13.	Create Timing diagram for the assigned project	Software design
14.	Create Component diagram for the assigned project.	Software design
15.	Create Deployment diagram for the assigned project.	Software design
16.	Estimation of Test Coverage Metrics and Structural Complexity.	Software testing
17.	Design and develop a program in a language of your choice to solve the triangle problem defined as follows: Accept three integers which are supposed to be the three sides of a triangle and determine if the three values represent an equilateral triangle, isosceles triangle, scalene triangle, or they do not form a triangle at all.	Test cases

	Assume that the upper limit for the size of any side is 10. Derive test cases for your program based on boundary-value analysis, execute the test cases, and discuss the results.	
18.	Design, develop, code, and run the program in any suitable language to solve the commission problem. Analyz it from the perspective of boundary value testing, derive different test cases, execute these test cases, and discuss the test results.	Black box Testing
19	Design and develop a program in a language of your choice to solve the triangle problem defined as follows: Accept three integers which are supposed to be the three sides of a triangle and determine if the three values represent an equilateral triangle, isosceles triangle, scalene triangle, or they do not form a triangle at all. Assume that the upper limit for the size of any side is 10. Derive test cases for your program based on equivalence class partitioning, execute the test cases, and discuss the results.	equivalence class partitioning
20	Design and develop a program in a language of your choice to solve the triangle problem defined as follows: Accept three integers which are supposed to be the three sides of a triangle and determine if the three values represent an equilateral triangle, isosceles triangle, scalene triangle, or they do not form a triangle at all. Derive test cases for your program based on decision-table approach, execute the test cases, and discuss the results.	decision-table based testing
21	Create test cases for a program which determine whether an integer is prime or not by using path testing.	Path testing
22	Create test cases for a program which determine whether an integer is prime or not by using Cyclomatic complexity.	White box testing
23	Consider a program to input two numbers and print them in ascending order. Find all du paths and identify those du-paths that are not feasible. Also find all dc paths and generate the test cases for all paths (dc paths and non dc paths).	DC path testing
24	Consider the code to arrange the nos. in ascending order. Generate the test cases for loop coverage and path testing. Check the adequacy of the test cases through mutation testing and compute the mutation score for each.	White box testing
25	Write Test cases for any Known Application (e.g., Banking Application)	Test case preparation
26	Create a test plan document for any application (e.g., Library Management System)	Test Plan
27	Study of any testing tool (e.g., Win Runner)	Testing Tools
28	Study of any bug tracking tool (e.g., Bugzilla, Bug bit)	Testing Tools
29	Study of any test management tool (e.g., Test Director)	Testing Tools
30	Study of any open source-Testing tool (e.g., Test link, Test Rail)	Testing Tools
31	Study of any web testing tool (e.g., Selenium)	Testing Tools
32	Mini Project with CASE tools.	Mini Project
33	Case Study Provided by Industry.	Case study

Course Code	AMICA0451	L T P	Credit
Course Title	Cloud Computing Lab	0 0 4	2

List of Experiments:

Sr. No	Name of Experiment	CO
1	Navigate the AWS Management Console.	CO1
2	Create and manipulate Elastic Compute Cloud instances.	CO1
3	Create AWS EC2 Virtual Machine Using AWS Console.	CO1
4	Monitoring Virtual Resources in AWS.	CO2
5	Getting Started with S3 in Cloud.	CO3
6	Working with EBS in AWS	.CO3
7	Build a relational database server.	CO3
8	Create private cloud - Designing a Custom VPC (Virtual Private Cloud).	CO4
9	Create an IAM Group in Cloud.	CO4
10	Built a RESTful serverless API on AWS.	CO5

ACTIVITIES

1. AWS Management Console Scavenger Hunt.
2. Estimate the cost of launching 2 EC2 Instances he AWS Pricing Calculator and TCO Calculator.
3. Select and research use cases for a specific database type and prepare a 10 min presentation.
4. Aurora Database.

Lab Course Outcomes: After completion of the course, students will be able to

CO 1	To know about the use AWS management console, create and manipulate Amazon instances.
CO 2	Access the encrypting and controlling of S3.
CO 3	Describe how to create private and virtual private cloud.
CO 4	How to create IAM group in cloud.
CO5	To understand the steps of Installation of Open Stack.