ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES (A)

(UGC Autonomous)

Approved by AICTE, Affiliated to Andhra University, Accredited by
N.B.A. & NAAC with ‘A’ Grade

(Estd : 2001)

PRAGNANAM BRAHMA
ANITS

2021-22

Academic Regulations (R20-AI&ML)

Curriculum & Syllabi (II Year I&II Semesters)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

VISION

Our vision is to emerge as a world class Computer Science and Engineering department through excellent teaching and strong research environment that responds swiftly to the challenges of changing computer science technology and addresses technological needs of the stakeholders.

MISSION

To enable our students to master the fundamental principles of computing and to develop in them the skills needed to solve practical problems using contemporary computer-based technologies and practices to cultivate a community of professionals who will serve the public as resources on state-of-the-art computing science and information technology.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO-1	Employability	Work as Competent Computer Engineer either globally or locally by engaging in professional practice in a variety of roles with ability to serve as a team or individual.
PEO-2	Higher studies	Prepared to pursue masters or research programmes in computer science or other disciplines.
PEO-3	Entrepreneurship	Become successful Entrepreneurs who demonstrate strong technical and leadership skills to bring out innovative designs/products that also addresses social issues.
PEO-4	Lifelong learning and ethics	Adapt to rapidly changing technology in engineering domains through continuous learning and practice code of ethics.

PROGRAM SPECIFIC OUTCOMES (PSOs)

1	Programming and software Development skills: Ability to acquire programming efficiency to analyze, design and develop optimal solutions, apply standard practices in software project development to deliver quality software product.
2	Computer Science Specific Skills: Ability to formulate, simulate and use knowledge in various domains like data engineering, image processing and information and network security, artificial intelligence etc., and provide solutions to new ideas and innovations

PROGRAM OUTCOMES (POs)

Graduate Attribute1:	Engineering Knowledge
PO-1	Apply the knowledge of basic engineering sciences, humanities, core engineering and computing concept in modeling and designing computer based systems.
Graduate Attribute2:	Problem Analysis
PO-2	Identify, analyze the problems in different domains and define the requirements appropriate to the solution.
Graduate Attribute3:	Design/Development of Solution
PO-3	Design, implement & test a computer based system, component or process that meet functional constraints such as public health and safety, cultural, societal and environmental considerations.
Graduate Attribute4:	Conduct Investigations of Complex Problems
PO-4	Apply computing knowledge to conduct experiments and solve complex problems, to analyze and interpret the results obtained within specified timeframe and financial constraints consistently.
Graduate Attribute5:	Modern Tool Usage
PO-5	Apply or create modern techniques and tools to solve engineering problems that demonstrate cognition of limitations involved in design choices.
Graduate Attribute6:	The Engineer and Society
PO-6	Apply contextual reason and assess the local and global impact of professional engineering practices on individuals, organizations and society.
Graduate Attribute7:	Environment and Sustainability
PO-7	Assess the impact of engineering practices on societal and environmental sustainability.
Graduate Attribute8:	Ethics
PO-8	Apply professional ethical practices and transform into good responsible citizens with social concern.
Graduate Attribute9:	Individual and Team Work
PO-9	Acquire capacity to understand and solve problems pertaining to various fields of engineering and be able to function effectively as an individual and as a member or leader in a team.

Graduate Attribute10:	Communication
PO-10	Communicate effectively with range of audiences in both oral and written forms through technical papers, seminars, presentations, assignments, project reports etc.
Graduate Attribute11:	Project Management and Finance
PO-11	Apply the knowledge of engineering, management and financial principles to develop and critically assess projects and their outcomes in multidisciplinary areas.
Graduate Attribute12:	Life-long Learning
PO-12	Recognize the need and prepare oneself for lifelong self learning to be abreast with rapidly changing technology.

ANITS- B Tech CSE (AI & ML) CURRICULUM – REGULATIONS –R20

I Year Course structure – CSE(AI & ML)

Semester - I

Course Code	Title of the course	Category	Periods						Sessional Marks	Semester end Exam marks	Total Marks	Credits
			L	T	P	E	O	Total				
CSM111	EngineeringMathematics – I	BS	3	0	0	1	6	10	40	60	100	3
CSM112	Communicative English	HS	3	0	0	0	3	6	40	60	100	3
CSM113	BASIC ELECTRONICS	ES	3	0	0	1	3	7	40	60	100	3
CSM114	PROBLEM SOLVING WITH C	ES	3	0	0	0	3	6	40	60	100	3
CSM115	Digital Logic Design	ES	3	0	0	1	3	7	40	60	100	3
CSM116	English Language Lab	HS	0	0	3	0	3	6	50	50	100	1.5
CSM117	Problem solving with C – lab.	ES	0	0	3	0	3	6	50	50	100	1.5
CSM118	Environmental Science (Mandatory non-credit course)	BS	3	0	0	0	1	4	50	-	50	-
Total			**18**	**0**	**6**	**3**	**25**	**52**	**350**	**400**	**750**	**18**

I Year Course structure – CSE(AI & ML)

Semester - II

Course Code	Title of the course	Category	Periods						Sessional Marks	Semester end Exam marks	Total Marks	Credits
			L	T	P	E	O	Total				
CSM121	Engineering Mathematics – II	BS	3	0	0	1	6	10	40	60	100	3
CSM122	Engineering Physics	BS	3	0	0	1	4	8	40	60	100	3
CSM123	Engineering Chemistry	BS	3	0	0	1	4	8	40	60	100	3
CSM124	ELEMENTS OF ELECTRICAL ENGINEERING	ES	3	0	0	1	4	8	40	60	100	3
CSM125	Engineering Drawing	ES	2	0	3	1	3	9	40	60	100	3.5
CSM126	Engineering Physics Lab.	BS	0	0	3	0	1	4	50	50	100	1.5
CSM127	Engineering Chemistry Lab.	BS	0	0	3	0	1	4	50	50	100	1.5
CSM128	Engineering Workshop	ES	0	0	3	0	1	4	50	50	100	1.5
CSM129	Human Values and Professional Ethics (Mandatory non-credit course)	HS	3	0	0	0	1	4	50	-	50	-
Total			**17**	**0**	**12**	**5**	**25**	**59**	**400**	**450**	**850**	**20**

II Year Course structure – CSE(AI & ML)

Semester - I

CODE	SUBJECT NAME	Category	Periods						Sessional Marks	Semester end Exam marks	Total Marks	Credits
			L	T	P	E	O	Total				
CSM211	DATA STRUCTURES&ALGORITHMS	PC	2	1	0	1	4	8	40	60	100	3
CSM 212	COMPUTER ORGANIZATION AND MICROPROCESSORS	PC	3	0	0	1	4	8	40	60	100	3
CSM 213	JAVA PROGRAMMING	PC	3	0	0	1	4	8	40	60	100	3
CSM 214	DATA COMMUNICATION AN D COMPUTER NETWORKS	PC	3	0	0	1	4	8	40	60	100	3
CSM 215	DISCRETE MATHEMATICAL STRUCTURES	BS	3	0	0	1	4	8	40	60	100	3
CSM 216	DESIGN THINKING & PRODUCT INNOVATION	ES	2	0	2	1	3	8	40	60	100	3
CSM 217	JAVA PROGRAMMING LAB	PC	0	0	3	0	2	5	50	50	100	1.5
CSM 218	DATA STRUCTURES LAB USING C	PC	0	0	3	0	2	5	50	50	100	1.5
CSM 219	Constitution of India & - Intellectual Property Rights (Mandatory non-credit course)	HS	2	0	0	0	1	3	50	-	50	-
Total			**18**	**1**	**8**	**6**	**28**	**61**	**390**	**460**	**850**	**21**

II Year Course structure – CSE(AI & ML)

Semester - II

CODE	SUBJECT NAME	Category	Periods						Sessional Marks	Semester end Exam marks	Total Marks	Credits
			L	T	P	E	O	Total				
CSM 221	PROBABILITY , STATISTICS AND QUEUEING THEORY	BS	3	0	0	1	6	10	40	60	100	3
CSM 222	ARTIFCIAL INTELLIGENCE	PC	2	1	0	2	4	9	40	60	100	3
CSM 223	OPERATING SYSTEMS	PC	3	0	0	1	4	8	40	60	100	3
CSM 224	PYTHON PROGRAMMING	PC	3	0	0	1	4	8	40	60	100	3
CSM 225	THEORY OF COMPUTATION AND COMPILERS	PC	2	1	0	1	4	8	40	60	100	3
CSM 226	DESIGN AND ANALYSIS OF ALGORITHMS	PC	2	1	0	1	4	8	40	60	100	3
CSM 227	***PYTHON PROGRAMMING LAB***	PC	0	0	***3***	***0***	***1***	4	***50***	***50***	100	1.5
CSM 228	***COMPUTER ORGANIZATION & MICRO PROCESSOR INTERFACING LAB***	***PC***	0	0	***3***	***0***	***1***	4	***50***	***50***	***100***	1.5
CSM 229	***OPERATING SYSTEM LAB***	***PC***	0	0	***3***	***0***	***1***	4	***50***	***50***	***100***	1.5
Total			**15**	**3**	**9**	**7**	**29**	**63**	**390**	**510**	**900**	**22.5**

III Year Course structure – CSE(AI & ML) (Tentative)

Semester - I

CODE	SUBJECT NAME	Category	Periods						Sessional Marks	Semester end Exam marks	Total Marks	Credits
			L	T	P	E	O	Total				
CSM 311	OPEN ELECTIVE -I*	OE	3	0	0	1	2	6	40	60	100	3
CSM 312	PROFESSIONAL ELECTIVE -I	PE	3	0	0	1	2	6	40	60	100	3
CSM 313	MACHINE LEARNING	PC	3	0	0	1	2	6	40	60	100	3
CSM 314	COMPETITIVE PROGRAMMING	SOC	2	1	0	1	5	9	40	60	100	3
CSM 315	DATA BASE MANAGEMENT SYSTEMS	PC	3	0	0	1	4	8	40	60	100	3
CSM 316	***MACHINE LEARNING LAB***	***PC***	0	0	***3***	***0***	1	4	***50***	***50***	100	1.5
CSM 317	***DATA BASE MANAGEMENT SYSTEMS LAB***	***PC***	0	0	***3***	***0***	1	4	***50***	***50***	100	1.5
CSM 318	***COMPETITIVE PROGRAMMING LAB***	***SOC***	0	0	3	0	1	4	***100***	***0***	100	1.5
CSM 319	**QA&VA**	HS	0	0	3	0	1	4	***100***	***0***	100	1.5
CSM 31A	***SUMMER INTERNSHIP-INDUSTRY***	***PR***	0	0	***0***	***0***	***1***	1	***100***	***0***	***100***	2
Total			**14**	**1**	**12**	**5**	**20**	**52**	**600**	**400**	**1000**	**23**

III Year Course structure – CSE(AI & ML)

Semester - II

CODE	SUBJECT NAME	Category	Periods						Sessionals Marks	Semester end Exam marks	Total Marks	Credits
			L	T	P	E	O	Total				
CSM 321	OPEN ELECTIVE -II*	OE	3	0	0	1	2	6	40	60	100	3
CSM 322	PROFESSIONAL ELECTIVE -II	PE	3	0	0	1	2	6	40	60	100	3
CSM 323	PROFESSIONAL ELECTIVE -III	PE	2	1	0	1	5	9	40	60	100	3
CSM 324	OBJECT ORIENTED SOFTWARE ENGINEERING	PC	3	0	0	1	4	8	40	60	100	3
CSM 325	WEB TECHNOLOGIES	PC	2	1	0	1	4	8	40	60	100	3
CSM 326	DEEP LEARNING	PC	3	0	0	1	4	8	40	60	100	3
CSM 327	***WEB TECHNOLOGIES LAB***	***PC***	0	0	***3***	***0***	***1***	4	***50***	***50***	100	1.5
CSM 328	***DEEP LEARNING LAB***	***PC***	0	0	***3***	***0***	***1***	4	***50***	***50***	100	1.5
CSM 329	**QA-II& SoftSkills**	HS	0	0	3	0	1	4	***100***	***0***	100	1.5
Total			**16**	**2**	**9**	**6**	**24**	**57**	**440**	**460**	**900**	**22.5**

IV Year Course structure – CSM(AI & ML) (Tentative)

Semester - I

CODE	SUBJECT NAME	Category	Periods						Sessionals Marks	Semester end Exam marks	Total Marks	Credits
			L	T	P	E	O	Total				
CSM 411	OPEN ELECTIVE -III*	OE	3	0	0	1	2	6	40	60	100	3
CSM 412	PROFESSIONAL ELECTIVE -IV	PE	3	0	0	1	2	6	40	60	100	3
CSM 413	PROFESSIONAL ELECTIVE -V/MOOC	PE	3	0	0	1	3	7	40	60	100	3
CSM 414	INTRODUCTION TO INTELLIGENT SYSTEMS	HS	3	0	0	0	2	5	40	60	100	3
CSM 415	AI IN ROBOTICS	SOC	2	1	0	1	4	8	40	60	100	3
CSM 416	***OOSE LAB***	***PC***	0	0	***3***	***0***	***1***	4	***50***	***50***	***100***	1.5
CSM 417	***AI IN ROBOTICS- LAB***	***SOC***	0	0	***3***	***0***	***1***	4	***50***	***50***	***100***	1.5
CSM 418	***PROJECT PHASE -I***	***PR***	0	0	***3***	***0***	***1***	4	***100***	***0***	***100***	2
CSM 419	***SUMMER INTERNSHIP-INDUSTRY***	***PR***	0	0	***0***	***0***	***1***	1	***100***	***0***	***100***	2
Total			**14**	**1**	**9**	**4**	**17**	**45**	**500**	**400**	**900**	**22**

IV Year Course structure – CSM(AI & ML)

Semester - II

CODE	SUBJECT NAME	Category	Periods						Sessionals Marks	Semester end Exam marks	Total Marks	Credits
			L	T	P	E	O	Total				
CSM 421	OPEN ELECTIVE -IV*	OE	3	0	0	1	3	7	40	60	100	3
CSM 423	***PROJECT PHASE 2& INTERNSHIP IN INDUSTRY***	***PR***	0	0	9	0	2	11	***100***	***100***	***200***	8
Total			**3**	**0**	**9**	**1**	**5**	**18**	**140**	**160**	**300**	**11**

Total Credits **160**

*Open Elective can be Inter Department Disciplinary Course,Emerging Courses or MOOC. Final decision will be taken by the department.

Hours Tracks	2nd Yr-Sem -2	3rd Yr-Sem -1	3rd Yr-Sem	4th Yr Sem -1
Programming and Application Development	Mobile Application Development	Full Stack Web Development	Cloud Application Developme	Microservices
Security Engineering Track	Information Security &	System Secure Engineering	Cyber Security	Hardware Security
Soft Computing techniques	Introduction to SoftComputing	Evolutionary techniques	Optimization techniques	Fuzzy Computing
Computer Networks and Engineering	Mobile and Cellular Networks	4G/5G Networks	Wireless Sensor Networks	Network Management

Professional Electives	
PE1	•CSM 312(A)Smart Systems Design & Programming •CSM312(B)Advanced Data Structures •CSM312(C)Digital ImageProcessing .CSM 312(D)Big Data
PE2	•CSM322(A) Computer vision •CSM 322(B) Mobile Computing .CSM322(C)No SQL Data Bases .CSM322(D)Data warehousing and Data mining
PE3	• CSM323(A)Distributed Operating Systems • CSM323(B)Embedded Systems • CSM323(C)Human Computer Interaction • CSM323(D)Pattern Recognition
PE4	.CSM412(A)AugmentedReality • CSM412(B)Bioinformatics .CSM 412(C)High Performance Computing • CSM415(D)Principles Of Programming Languages
PE5	.CSM413(A)IOT • CSM413(B)Fuzzy Computing • CSM413(C)Social Network Analysis • CSM413(D) Cloud Computing

DATA STRUCTURES & ALGORITHMS	
CSE 211	**Credits : 3**
Instruction : 2 Periods & 1 Tut/Week	**Sessional Marks : 40**
End Exam : 3 Hours	**End Exam Marks : 60**

Prerequisites:

- Basic Knowledge of Programming Fundamentals
- Knowledge of Programming Languages (C)

Course Objectives:

The course should enable the students:

- To acquire knowledge on several linear and nonlinear data structures like stacks, queues, linked list, trees and graphs.
- To have better insight into to learn various sorting and searching techniques.
- To exercise the applications of data structures.
- To have a good understanding of problem solving using data structure tools and techniques.

Course Outcomes:

By the end of the course, the student will be able to:	
1.	Analyze the complexities of recursive and Non recursive algorithms and Implement linear, binary, interpolation, hashing searching techniques and sorting techniques namely bubble, insertion, selection, quick, merge sort.
2.	Apply ADT concepts such as stacks and queues for solving infix to post fix, postfix evaluation, priority queues.
3.	Apply the concepts of dynamic memory allocation for reducing the time and space complexity of algorithms.
4.	Design and implement the Nonlinear data structures (trees) to optimize the solution.
5.	Design and Implement Warshall's Algorithm, Shortest path Algorithm-Dijkstra's Algorithm, Minimum cost spanning trees (Prims and Kruskal's algorithms),Graph traversals (Breadth first search and Depth first Search algorithms.)

Mapping of Course Outcomes with Program Outcomes:

CO	PO												PSO	
	1	2	3	4	5	6	7	8	9	10	11	12	1	2
1	3	3	2	3	-	-	-	-	-	-	-	1	3	0
2	2	2	3	2	-	-	-	-	-	-	-	1	2	0
3	2	2	3	2	-	-	-	-	-	-	-	1	3	0
4	2	2	3	2	-	-	-	-	-	-	-	1	2	0
5	2	3	3	3	-	-	-	-	-	-	-	1	3	0

SYLLABUS

UNIT-I: **15 hours**

Introduction: Basic Terminology, Elementary Data Organization, Data Structure operations, Fundamentals of analysis of algorithms and efficiency – Asymptotic Notations and Basic Efficiency classes.

Arrays: Array Definition, Representation and Analysis, Single and Multidimensional Arrays, address calculation, application of arrays, Character String in C, Character string operation, Array as Parameters, Sparse Matrices.

Searching &Sorting: Sequential search, binary search, Interpolation Search, comparison and analysis, Hash Table, Hash Functions. Complexity of Search Algorithm, Insertion Sort, Bubble Sort, Selection Sort, Quick Sort, Merge Sort.

Learning Outcomes:

1: Analyze the complexity of Algorithms, Implement searching and soring algorithms.

2: Implement the searching and soring algorithms.

UNIT-II: **12 hours**

Stacks: Array Representation and Implementation of stack, Operations on Stacks: Push & Pop, Applications of stack: Conversion of Infix to prefix and Postfix Expressions, Evaluation of Postfix & Prefix expressions using stack, Recursion, Towers of Hanoi Problem.

Queues: Array representation and implementation of queues, Operations on Queue: Insert, Delete, Full and Empty. Circular queue, De-queue, and Priority Queue, Applications of Queues.

Learning Outcomes:

1: Implement stacks and queues using ADT and Implement the applications of Stacks and queues (solving infix to post fix, postfix evaluation, priority queues.)

2: Apply ADT and implement Stack and queue and applications of stack and queue.

UNIT-III: **12 hours**

Linked list: Representation and Implementation of Singly Linked Lists, Traversing and Searching of Linked List, Insertion and deletion to/from Linked Lists, Doubly linked list, Circular doubly linked list, implementing priority queue using Linked List, Polynomial Representation using Linked list & addition.

Learning Outcomes:

1: Implement singly linked list, Doubly Linked List, Circular doubly linked list and applications.

2: Implement Linked Lists and applications of Linked Lists.

UNIT-IV: **12 hours**

Trees: Basic terminology, Binary Trees, Binary tree representation, Almost Complete Binary Tree, Complete Binary Tree, Array and Linked Representation of Binary trees, Traversing Binary trees, Threaded Binary trees. Binary Search Tree (BST), Insertion and Deletion in BST, AVL Trees-Rotations in AVL trees, Insertion and Deletion in AVL.

Learning Outcomes:

1: Design and implement BST, AVL trees.

2: Implement BST, AVL tree along with various operations performed on BST and AVL tree.

UNIT-V: 12 hours

Graphs: Terminology & Representations- Graphs, Directed Graphs, Adjacency Matrices, Path OR Transitive Closure of a Graph, Warshall's Algorithm, Shortest path Algorithm-Dijkstra's Algorithm, Connected Component and Spanning Trees, Minimum Cost Spanning Trees, Graph Traversals.

Learning Outcomes:

1**:** Implement Graph Traversals algorithm and Minimum Cost Spanning Trees algorithms.

2: Implement Warshall's Algorithm, Shortest path Algorithm-Dijkstra's Algorithm and Minimum Cost Spanning Trees algorithm

TEXT BOOKS

1. Mark Allen Weiss, "Data Structures and Algorithm Analysis in C", Pearson Education, 2nd Edition, 1996

REFERENCE BOOKS

1. E.Horowitz and Sahani, "Fundamentals of Data Structures", W H Freeman & Co Publication, 1983.
2. S. Lipschutz, "Data Structures", McGraw Hill Publications, 1986.
3. P. Dey & M. Ghosh, "Programming in C" , Oxford Univ. Press, 2012
4. ISRD Group, "Data Structures through C++", McGraw Hill, 2011.

Web Resources:

1. https://nptel.ac.in/courses/106/102/106102064/
2. https://www.coursera.org/learn/data-structures?specialization=data-structures-algorithms
3. https://www.udacity.com/course/data-structures-and-algorithms-nanodegree--nd256

COMPUTER ORGANIZATION AND MICROPROCESSORS	
CSM 212	**Credits : 3**
Instruction **:** 3 Periods/Week	Sessional Marks : 40
End Exam : 3 Hours	End Exam Marks : 60

PREREQUISITES: Basic knowledge of Digital Logic Design

COURSE OBJECTIVES:

- The objective of the course is to understand the basic structure and operation of digital computer.
- To study the design of arithmetic and logic unit and control unit techniques.
- To understand the different ways of communicating with I/O devices and standard I/O interfaces.
- Familiar with the architecture and the instruction set of an Intel microprocessors 8085 and 8086.
- To learn the assembly language programming implementation of the 8085 and 8086 programs.

COURSE OUTCOMES:

By the end of the course, the student will be able to:	
1.	Identify the basic principles and apply to arithmetic for ALU implementation.
2.	Demonstrate the computer Arithmetic and hardware implementations of it.
3.	Compare and assess the working principles of hardwired and microprogrammed control and I/O interfacing.
4.	Analyze the architecture , instruction sets, subroutines and interrupts of 8085 and implementation of the programming using 8085
5.	Summarize the architecture , pin configuration and instruction sets of 8086 and implementation of the programming using 8086

Mapping of Course Outcomes with Program Outcomes:

CO	**PO**												**PSO**	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
1	**2**	**2**	**1**	**1**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**
2	**2**	**2**	**2**	**1**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**
3	**2**	**-**	**1**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**1**	**-**	**-**
4	**3**	**2**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**1**	**1**	**-**
5	**3**	**2**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**1**	**1**	**-**

SYLLABUS

UNIT- I

Basic computer Organization **12 Hours**

Register Transfer Language, Instruction Codes, Computer Registers, Computer Instructions, Timing and Control, Instruction Cycle, Memory-Reference Instructions

ALU

Arithmetic, Logic and Shift Micro operations, Arithmetic Logic Shift Unit.

Learning Outcomes:

1. Identify the basic principles of a computer , Computer Instructions & Memory Instructions
2. Identify the ALU circuit and its operations.

UNIT -II

Computer Arithmetic: **10 hours**

Introduction, Addition and Subtraction, Booth Multiplication, Division & Decimal Arithmetic Unit, Hardware Implementation & Algorithms.

Learning Outcomes:

1. Apply the Arithmetic operations for ALU Implementation.
2. Design the Hardware implementation of Arithmetic circuits.

UNIT – III **15hours**

Control Unit and I/O Interface:

Control Design: Hardwired & Micro Programmed (Control Unit), Control Memory, Address Sequencing, Micro program Example.

Input/output Interfaces: Peripheral Devices, Input-Output Interface, Asynchronous Data Transfer, Modes of Transfer, Priority Interrupt, Direct Memory Access, Parallel Programmable peripheral interface (8255).

Learning Outcomes:

1. Distinguish between Hardwired & Micro programmed control units.
2. Demonstrate the parallel I/O interfacing using 8255 PPI.

UNIT –IV **15 hours**

Introduction to 8085 Microprocessor

Introduction to Microprocessors and Microcomputers, Internal Architecture and Functional / Signal Description of typical 8-bit µP. 8085, Instruction Set, types of Instructions, Addressing modes of 8085 and Timing Diagrams of 8085 µP.

Assembly Language Programming Introduction to Assembly Language Programming Techniques: Looping, Counting, and Indexing, Counter and timing Delays, Stack and Subroutines, Code Conversions, BCD Arithmetic operations , 16-bit data Operations, Interrupts and Interrupt Service Routines.

Learning Outcome: At the end of this Unit the students will be able to

1. Draw and describe the basic architecture of 8085 and the functional description of 8085.
2. Develop the assembly language programs using various programming techniques

UNIT-V **15 hours**

Introduction to 8086 Microprocessor

Internal Architecture and Functional/Signal Description of 8086/8088, Segmented Memory, Maximum-Mode and Minimum-Mode Operation and Addressing Modes of 8086.

Assembly Language Programming
Instruction Set and Timing Diagrams, Interrupts and Interrupt Service Routines, Assembler Directives, Loops Procedures

Learning Outcomes: At the end of this unit, student will be able to

1. Describe the modes and functional block diagram of 8086 along with pins and their functions.
2. Develop the assembly language programs using various programming techniques.

Text Books:

1. M.Morris Mano, ―Computer System Architecture‖, Pearson Education Inc., 2003, Third Edition.
2. Ramesh S. Gaonkar, ―Microprocessor Architecture, Programming, and Applications with the 8085‖ Penram International, 6th Edition.
3. John E.Uffenbeck, ―The 80x86 Family, Design, Programming and Interfacing 3rdEdition, Pearson Education Inc.‖, 2002.

REFERENCE BOOKS:

1. William Stallings, Computer Organization and Architecture, 6th Edition, Pearson/PHI, 2007.
2. Andrew S. Tanenbaum, Structured Computer Organization, 5th Edition, PHI/Pearson, 2007.
3. BARRY B. BREY, ―The Intel Microprocessors 8086/8088, 80186/80188,80286,80386 and 80486, Pentium, Pentium Pro Processor, Pentium II, Pentium III, Pentium 4, Architecture, Programming and Interfacing‖, Pearson Education Inc., 2003,6thEdition.

ONLINE WEB RESOURCES:

1. https://www.tutorialspoint.com/microprocessor/index.htm
2. https://swayam.gov.in/nd1_noc20_ee11/preview

JAVA PROGRAMMING	
CSM 213	**Credits : 3**
Instruction : 3 Hours /Week	**Sessional Marks : 40**
End Exam : 3 Hours	**End Exam Marks : 60**

Pre requisites:

- Basic knowledge of computer fundamentals.
- Student must have knowledge of some programming languages (such as C, C++)

Course Objectives:

- To Understand Object Oriented Programming Concepts and Apply Them in Problem Solving.
- To Learn The Basics of Java Console and GUI Based Programming.

Course Outcomes:

At the end of the course the student will be able to

CO1	Design Classes for Real Time Applications.
CO2	Establish The Connectivity Among The Classes Using Inheritances And Interfaces.
CO3	Modularize The Application Using Packages and apply threads on classes to achieve parallelism through synchronization.
CO4	Develop Test Cases By Including The Runtime Errors Using Exceptions Handling Mechanism and multi Threading
CO5	Identify AWT components to Design the GUI Using Applet & AWT Frameworks.

CO-PO MAPPING:

CO	PO												PSO	
	1	2	3	4	5	6	7	8	9	10	11	12	1	2
1	3	3	2	3	-	-	-	-	-	-	-	1	3	-
2	2	2	3	2	-	-	-	-	-	-	-	1	2	-
3	2	2	3	2	-	-	-	-	-	-	-	1	3	-
4	2	2	3	2	-	-	-	-	-	-	-	1	2	-
5	2	3	3	3	-	-	-	-	-	-	-	1	3	-

Correlation Levels 1 2 3 Defined as Below

1 High: Strong Correlation

2 Medium: Moderate Correlation

3 Low: Slight

SYLLABUS

UNIT-I **10-12hours**

Fundamentals of Object Oriented Programming : Introduction, object oriented paradigm, object and classes, Data Abstraction and Encapsulation, Inheritance, Polymorphism, Dynamic Binding, Applications of OOP.

Java programming - History of Java, Java Buzzwords, Data types, variables, operators. Control structures, arrays, console input and output, Simple programs on java. Introduction to Classes, objects, constructors, methods, parameter passing, overloading constructors and methods, static fields and methods, this reference, final keyword, garbage collection, finalize method, inner class, String handling.

Learning Outcomes: At the end of this unit the Students will be able to

- Identify the object and understand object oriented principles
- Create class, constructor and can handle string operations

UNIT – II **10-12 hours**

Inheritance – Basics, using super keyword, multilevel hierarchy, Member access rules, preventing inheritance- using final, the Object class and its methods.

Polymorphism - dynamic binding, method overriding, abstract class and methods. Interfaces - Interfaces vs. Abstract class, defining an interface, implementing interfaces, accessing implementations through interface references, extending interfaces.

Packages - Defining, Creating and Accessing a Package, importing packages

Learning Outcomes: At the end of this unit the Students will be able to

- Derive a class from existing class or from interface
- Define a package and importing class from package

UNIT –III **10-12 hours**

I/O: I/O basics, byte and character streams, read/ write console input/output, reading and writing files.

Exception handling – Fundamentals, Exception types, use of try and catch, throw, throws, finally, multiple catches, built-in exceptions, user defined exceptions.

Multithreading – Thread Priorities, synchronization, messaging, reading a thread, creating multiple threads, use of alive and join, inter-thread communication- suspending resuming and stopping threads, producer-consumer problem with multithreading.

Learning Outcomes: At the end of this unit the Students will be able to

- Handle predefined Exceptions and can define custom exceptions
- Split a complex task into multiple threads.

UNIT-IV **10-12 hours**

Applets- Simple HTML tags, Difference between Application and Applet ,Applet class, Applet structure, An Example Applet Program, Applet Life Cycle, paint(),update() and repaint().

Swing-Introduction , JFrame, JApplet, JPanel, Components in swings, JList and JScroll Pane, Split Pane, JTabbedPane, Dialog Box.

Layout Managers: java. awt. Border Layout, Flow Layout, Grid Layout, Card Layout, GridBagLayout.

Learning Outcomes: At the end of this unit the Students will be able to

- Design Swing Applet class with html tag
- Arrange components in Layouts

UNIT-V **10-12 hours**

Abstract Window Toolkit

Why AWT?, java. Awt package, Components and Containers, Button, Label, Checkbox, Radio buttons, List boxes, Choice boxes, Text field and Text area, container classes, Layouts, Menu, Scroll bar.

Event Handling: The Delegation event model, Event classes, Event Listener interfaces, handling Action event, Item Event, Mouse Event, keyboard event and Window Events.

Learning Outcomes: At the end of this unit the Students will be able to

- Design GUI components using AWT.
- Define Event Handling on the components using Delegation event model.

TEXT BOOKS

1. Herbert Schildt, "JAVA The Complete Reference", TataMcGraw Hill,seventh edition.
2. E Balagurusamy, "Programming with JAVA - A Primer" – Third Edition.

REFERENCES BOOKS

1. 1. P.J. Deitel and H.M. Deitel, "Java for Programmers", Pearson education (OR) P.J. Deitel and H.M. Deitel, "Java: How to Program", PHI.
2. P. Radha Krishna, "Object Orientd Programming through Java", Universities Press.
3. Bruce Eckel, "Thinking in Java", Pearson Education
4. Bruce Eckel, "Programming in Java", Pearson Education
5. S. Malhotra and S. Choudhary, "Programming in Java", Oxford Univ. Press.

DATA COMMUNICATION AND NETWORKS	
CSM 214	CREDITS:3
INSTRUCTION: 3 Periods /Week	SESSIONAL MARKS: 40
FINAL EXAM: 3Hours	FINAL EXAM MARKS: 60

Pre requisites: Knowledge on computer basics, hardware and computer applications·

Course Objectives:

- Acquire knowledge of network terminology and layered communication architectures (OSI and TCP/IP).
- Gain knowledge of digital encoding techniques, multiplexing and spread spectrums.
- Illustrate the principles of error detection, error correction congestion control and trade-offs in fairness and efficiency.
- Explain the principles of routing and IP addressing.
- Interpret the usage of DNS server and working of application protocols.

Course Outcomes:
By the end of the course Students will be able to

1.	Describe the basics of data communications, categorize the network topologies and differentiate between the TCP/IP and OSI.
2.	Explain the design objectives of LLC and MAC and apply various error detection and correction methods.
3.	Implement the subnetting concept and analyze various routing and congestion control. algorithms.
4.	Evaluate the working of TCP - UDP operations and the performance metrics.
5.	Analyse the importance of DNS server and its usage.

CO-PO Mapping:

CO	PO												PSO	
	1	2	3	4	5	6	7	8	9	10	11	12	1	2
1	3	1	-	-	-	1	1	-	-	-	1	1	1	-
2	3	2	2	2	1	-	1	-	-	-	1	1	1	1
3	3	2	3	2	1	-	-	-	-	-	-	-	1	1
4	1	3	1	1	1	-	-	-	-	-	-	-	1	1
5	2	2	2	1	1	-	-	-	-	-	-	-	1	1

SYLLABUS

UNIT-I **12 Hours**

Introduction to Computer Networks: Introduction to Data Communications, Networking - Components, Categories and classification, Network Topologies, Reference Models-ISO-OSI, TCP/IP, Transmission Media, Introduction-Signal Representation, Modulation and Demodulation Techniques, Switching –Circuit, Packet, Message switching techniques .

Learning Outcomes: At the end of this unit the Students will be able to

- Compare various types of networks.
- Contrast TCP/IP and OSI.

UNIT- II **12 Hours**

Data Link Layer: LLC: Data Link Layer Design Issues, Error-Error Detection, Control and Correction Techniques- Parity, LRC,CRC, Hamming Code, Flow Control- Sliding Window Protocols-Go Back N, Selective Repeat, HDLC.

Medium Access Control Layer: Channel Allocation Problems- Static and Dynamic, MAC protocols: ALOHA, CSMA- CSMA/CD, CSMA/CA.

Learning Outcomes: At the end of this unit the Students will be able to

- Applying error correction and detection methods.
- Finding the best possible channel allocation protocol.

UNIT-III **12 Hours**

Network Layer: Datagram Approach, Protocols-ARP,DHCP, IPv4, IPv6, ICMP, Routing Algorithms: Distance vector Routing, Link State Routing, Hierarchical Routing, Multicast Routing and Routing for Mobile hosts. Subnetting Concept, Network Address Translation, Congestion Control Techniques: Approaches, Traffic aware Routing, admission control, Traffic throttling, Load Shedding.

Learning Outcomes: At the end of this unit the Students will be able to

- Finding the best suitable routing algorithm for the given network design.
- Implement the subnetting.

UNIT-IV **12 Hours**

Transport Layer: The Transport Service, Elements of Transport Protocols, Congestion Control, UDP: Introduction, Remote Procedure call, RTP, TCP: Introduction, Service model, TCP protocol, TCP segment Header, TCP connection Establishment, Connection Management, Connection release, TCP sliding Window.

Learning Outcomes: At the end of this unit the Students will be able to

- Examine services of transport layer protocols.
- Compare the UDP and TCP protocols.

UNIT-V **12 Hours**

Application Layer: DNS Name space, Domain Name records, Name servers, Electronic Mail: Architecture and services, The user agent, Message Formats, Message Transfer. The Worldwide Web: Architectural overview. HTTP, HTTPS, FTP, Mobile Web, Web search. Security: Introduction to Cryptography, Security Services, Message Confidentiality and Integrity.

Learning Outcomes: At the end of this unit the Students will be able to

- Examine Naming Services.
- Explain the importance of Application layer protocols.

Text Books:

1. Behrouz A Forouzan ―Data Communications and Networking‖ , 5th Edition, Tata McGraw-Hill.
2. Andrew S. Tanenbaum, ―Computer Networks‖, 5th edition, Prentice-Hall Publisher

Reference Books:

1. William Stallings, ―Data and Computer Communications‖. 7th edition, Pearson Education
2. J F Kurose, K W Ross, ―Computer Networking: A Top-Down Approach", 5th Edition,
   AddisonWesley.

Online Resources:

1. https://nptel.ac.in/courses/106/105/106105081/
2. https://swayam.gov.in/nd1_noc20_cs23/preview

DISCRETE MATHEMATICAL STRUCTURES	
CSM 215	**CREDITS:3**
INSRUCTION: 3 Periods /Week	**SESSIONAL MARKS: 40**
FINAL EXAM: 3Hours	**FINAL EXAM MARKS: 60**

Prerequisites:

- Elementary knowledge of Set theory, Matrices and Algebra.

Course Objective :

The main objectives of the course are to:

- Introduce concepts of mathematical logic for analyzing propositions and proving theorems.
- Use sets for solving applied problems binary relations and introduce concepts of algebraic
- structures
- Work with an ability to solve problems in Combinatorics
- Solve problems involving recurrence relations and generating functions.
- Introduce basic concepts of graphs, digraphs and trees

Course Outcomes: At the end of the course student should be able to

1	Apply mathematical logic, mathematical reasoning and to study about the validity of the arguments and also prove mathematical theorems using mathematical induction.
2	Determine properties of binary relations; identify equivalence and partial order relations, sketch relations and Familiarize with algebraic structures.
3	Apply counting techniques to solve combinatorial problems and identify, formulate, and solve computational problems in various fields.
4	Identify the Recurrence relation, generating functions and know the methods for solving problems involving recurrence equations.
5	Familiarize with the applications of graphs, trees and algorithms on minimal spanning tress and apply graph theory in solving computing problems

CO – PO MAPPING

CO	PO												PSO	
	1	2	3	4	5	6	7	8	9	10	11	12	1	2
1	3	2	1	-	-	-	-	-	-	-	-	1	-	-
2	3	2	1	-	-	-	-	-	-	-	-	1	-	-
3	3	2	1	-	-	-	-	-	-	-	-	1	-	-
4	3	2	1	-	-	-	-	-	-	-	-	1	-	-
5	3	2	1	-	-	-	-	-	-	-	-	1	-	-

SYLLABUS

UNIT - I: MATHEMATICAL LOGIC (12Periods)

Fundamentals of logic – Logical inferences – Methods of proof of implication – First order logic and other proof methods – Rules of inference for quantified propositions – Mathematical induction.

Sections: 1.5 to 1.10 of Text book [1].

Learning outcomes: At the end of this unit, student will be able to

1. Apply inference theory to verify the consistence of data.
2. Construct logical statements from informal language to propositional logic expressions.

UNIT - II: RELATIONS AND ALGEBRAIC SYSTEMS (12 Periods)

RELATIONS : Cartesian products of sets – Relations – Properties of binary relations in a set –Relation matrix and graph of a relation– Partition and covering of set – Equivalence relations– Composition of binary relations–Transitive closure of a relation–Partial ordering – Partially ordered set.

Sections: 2-1.9, 2-3.1 to 2-3.5, 2-3.7, 2-3.8, 2-3.9 of Text book [2].

ALGEBRAIC SYSTEMS: Definitions and simple examples on Semi groups – Monoids – Group – Ring and Fields.

Sections: 3-1.1, 3-2.1,3-2.2, 3-5.1,3-5.11and 3-5.12 of Text book [2].

Learning outcome: At the end of this unit, student will be able to

1. Determine properties of relations, identify equivalence and partial order relations, sketch relations.
2. Identify Semi group, Monoid, Group, Ring and Field for a given algebraic structure

UNIT - III: ELEMENTARY COMBINATORICS (10 Periods)

Basics of counting – Combinations and permutations – Their enumeration with and without repetition – Binomial coefficients – Binomial and multinomial theorems – The principle of inclusion and exclusion.

Sections: 2.1 to 2.8 of Text book [1].

Learning outcome: At the end of this unit, student will be able to

1. Solve problems on binomial and multinomial coefficients.
2. Solve counting problems by using principle of inclusion-exclusion.

UNIT - IV: RECURRENCE RELATIONS (10 Periods)

Generating functions of sequences – Calculating their coefficients – Recurrence relations – Solving recurrence relations – Method of characteristic roots – Non-homogeneous recurrence relations and their solutions.

Sections: 3.1to 3.6 of Text book [1].

Learning outcome: At the end of this unit, student will be able to

1. Formulate recurrence relations of the sequences and solve problems on generating functions.
2. Evaluate complementary function and particular integral for non- homogeneous linear recurrence relations.

UNIT – V: GRAPHS **(16 Periods)**

Introduction to graphs – Types of graphs – Graphs basic terminology and special types of simple graphs – Representation of graphs and graph isomorphism – Euler paths and circuits – Hamilton paths and circuits – Planar graphs – Euler's formula.

Introduction to trees and their properties – Spanning trees – Minimum spanning trees – Kruskal's algorithm .

Sections: 5.1 to 5.4, 5.7, 5.8, 5.9, and 5.10 of Text book [1].

Learning outcome: At the end of this unit, the student will be able to

1. Identify different graphs and their properties.
2. Build minimal spanning tree by using different algorithms.

TEXT BOOKS:

1. **Joe L. Mott, Abraham Kandel & T. P. Baker**, Discrete Mathematics for computer scientists &Mathematicians, Prentice Hall of India Ltd, New Delhi., 2008

2. **J. P. Tremblay, R. Manohar**, Discrete Mathematical Structures with Applications to Computer Science, Tata McGraw-Hill Publishing Company Limited,1997

REFERENCE BOOKS:

1. **Keneth. H. Rosen**, Discrete Mathematics and its Applications, 6/e, Tata McGraw-Hill, 2009.
2. **Richard Johnsonburg**, Discrete mathematics, 7/e, Pearson Education, 2008.

ANITS (A)
CSE - DEPARTMENT
II YEAR – I SEMESTER R20

DESIGN THINKING AND PRODUCT INNOVATION												
Code	**Category**	**Periods**						**Sessional Marks**	**End Exam Marks**	**Total Marks**	**Credits**
		L	**T**	**P**	**E**	**O**	**Tota l**				
CSM 216	**ES**	**2**	**0**	**2**	**1**	**3**	**8**	**40**	**60**	**100**	**3**

Prerequisite: NIL

Course Objectives:

The course titled Innovation, Business Models and Entrepreneurship are designed to give an in-depth Understanding on Various aspects of Innovation, Creativity, evolving business models, incubation and entrepreneurship. Come up with exposure to design thinking for designing innovative products. The course is a blend of theory and practice therefore this course does not require any prerequisite and will be useful to understand innovation and its applications in different spheres of development and growth. Driven by a vision to empower students with design thinking skills to be able to bring innovation and personal effectiveness to solve problems for the organization and society.

Course Outcomes:

The Student will be able to:

CO-1	Identify the Principles and Elements of Design; gain knowledge of the Need and charectestics of Design Thinking.
CO-2	Apply the Design Thinking process and use tools like Persona, Empathy Map for solving problems in user centric way.
CO-3	Develop skills in Brainstorming, prototype, testing and implementation for Product Design and Development.
CO-4	Create the Innovative Products by applying Lateral - Divergent and Convergent Thinking. Implementing design thinking for better process
CO-5	Apply the Design thinking Techniques for solving problems in various sectors like Education , I.T. , Finance and Management

Mapping of Course Outcomes with Programme Outcomes.
High-3, Medium-2, Low-1

COURSE OUTCOMES	**PO1**	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**
CO1	**3**	**3**	**2**	**-**	**1**	**-**	**1**	**-**	**-**	**1**	**1**	**1**
CO2	**3**	**3**	**1**	**-**	**1**	**-**	**1**	**-**	**-**	**1**	**1**	**1**
CO3	**3**	**3**	**2**	**-**	**1**	**-**	**1**	**-**	**-**	**1**	**1**	**1**
CO4	**3**	**2**	**1**	**-**	**1**	**-**	**1**	**-**	**-**	**1**	**1**	**1**
CO5	**3**	**3**	**2**	**-**	**1**	**-**	**1**	**-**	**-**	**1**	**1**	**1**

Course Outcomes	PSO1	PSO2
CO-1	2	1
CO-2	2	1
CO-3	2	1
CO-4	2	1
CO-5	2	1

SYLLABUS

UNIT-I
Introduction to Design Thinking: 6L+4P=10Periods
Introduction to elements and principles of Design, Introduction to design thinking, history and need of design thinking. 7 characteristics that define design thinking, comparison of design thinking to other ways of thinking, 5 characteristics of action plan. Problem statement.

Learning Outcomes

1. Explain the concept of design and its history
2. Describe the need of Design Thinking
3. Elucidate the charectestics of Design Thinking

UNIT-II 6L+8P=14Periods
Design Thinking process and Tools:
Design Thinking process empathize, analyze, ideate, prototype & Test. Implementing the process in driving inventions, design thinking in social innovations. Tools of Design Thinking - Ask 5x why, 5W+H questions, Empathy map, persona, costumer journey map for solving problems in user centric way.

Learning outcomes

1. Describe the ideas and tools required to solve a problem
2. Explain the design process
3. Solve a problem as a team
4. Identify the roles and responsibilities as a team member

UNIT – III 6L+4P=10Periods
Methods and Tools Implementation: -
Brain storming - How might we -question, Storytelling. Critical Function Prototype (CFP). Testing sheet, Feedback, Powerful questions in experience testing, Road map for implementation.

Product Design: problem formation, introduction to product design, Product strategies, Product value, Product planning, product specifications, Product development.

Learning Outcomes

1. Identify innovative problem solutions
2. Analyze the solution
3. Design Prototype and Testing methods
4. Design an innovative product

UNIT –IV **6L+6P=12Periods**

Product strategic Innovation: Innovation, Difference between innovation and creativity, role of creativity and innovation in organizations. Innovative Products by applying Lateral-Divergent and Convergent Thinking. Implementing design thinking for better process.

Learning Outcomes

1. Identify skills for innovation
2. Design with empathetic experience
3. Solve a problem innovatively

UNIT-V **6L+8P=14Periods**

Design thinking in various sectors: Design thinking for Startups. Double Dimond method - discover, define, develop and deliver. Case studies in Information Technology, Finance, Education and Management

Learning Outcomes

1. Apply Design thinking for Startups.
2. Apply Double Diamond method for various sectors
3. Perform case studies on various sectors.

Case study learning outcomes:

1. Make use of practical design thinking methods in every stage of problem with the help of method templates.
2. Apply design thinking to a problem in order to generate innovative and user-centric solutions.
3. Empathize with end user and initiate a new working culture based on user-centric approach.
4. Prototype and run usability tests for unbiased examination of the product in order to identify problem areas.

TEXT BOOKS:	
1.	Daniel Ling "*Complete Design Thinking Guide for Successful Professionals*", Emerge Creatives Group LLP, Print ISBN: 978-981-09-5564-9.
2.	Product Design and Manufacturing by A.K. Chitale and R.C. Gupta, Prentice Hall
3.	Michael Lewrick, Patrick Link, Larry Leifer, The Design Thinking Toolbox, John Wiley & Sons, 2020.

REFERENCE BOOKS:	
1.	Michael G. Luchs, Scott Swan, Abbie Griffin , *"Design Thinking: New Product Development Essentials from the PDMA"*, ISBN-13 : 978-1118971802
2.	Tim Brown, *Change by Design: How Design Thinking Transforms Organizations and Inspires Innovation*, HarperCollins e-books, 2009.
3.	Beverly Rudkin Ingle, *"Design Thinking for Entrepreneurs and Small Businesses"*, Apress, ISBN: 9781430261827
4.	Jose Betancur *"The Art of Design Thinking: Make More of Your Design Thinking Workshops"*, ISBN: 9781522095378
5.	Michael Lewrick, Patrick Link, Larry Leifer, The Design Thinking Playbook, John Wiley & Sons, 2018

6.	Jeanne Liedtka, Andrew King, And Kevin Bennett, "*Solving Problems with Design Thinking*" , Columbia University Press Publishers, E-ISBN 978-0-231-53605-9

WEB RESOURCES:	
1.	https://dschool.stanford.edu/resources/design-thinking-bootleg
2.	https://www.ideo.com/post/design-thinking-for-educators
3	https://nptel.ac.in/courses/110/106/110106124/#

JAVA LAB	
CSE 217	**CREDITS:1.5**
INSTRUCTION: 3 Periods/Week	**SESSIONAL MARKS: 50**
FINAL EXAM: 3Hours	**FINAL EXAM MARKS: 50**

Prerequisites:

- Basic knowledge of computer fundamentals.
- Student must have knowledge of some programming languages (such as C, C++)

Course Objectives:

- To understand object oriented programming concepts, and apply them in problem solving.
- To learn the basics of java Console, GUI based programming and networking programming.

Course Outcomes:

Student will be able to

CO1	Solve the given problem using basics of Java programming.
CO2	Develop the program using class and object.
CO3	Implement the solution using Inheritance and modularize the application using packages.
CO4	Apply multi threading, Exception handling, File Handling and Design GUI applications using java AWT and applets**.**

CO-PO Mapping:

CO	**PO**												**PSO**	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
1	2	2	1	2	1	-	-	-	-	-	1	2	2	2
2	2	3	3	3	1	-	-	-	-	-	1	2	2	2
3	1	2	3	3	1	-	-	-	-	-	1	2	2	2
4	2	1	3	1	1	-	-	-	-	-	1	2	2	2

JAVA LAB PROGRAMS

1. Write a java program which reads your name and other details through command line and print them. CO1
2. Arrays CO1

Write a to find the Valid program Parentheses

Given a string containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid. An input string is valid if:

1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.

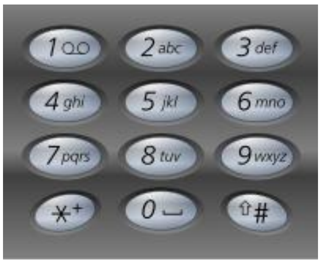
Note that an empty string is also considered valid. Input:( ) output:valid

Input: ({) } Output: Not valid

3. Letter Combinations of a Phone Number CO1

Given a string containing digits from 2-9 inclusive, return all possible letter combinations that the number could represent.

A mapping of digit to letters (just like on the telephone buttons) is given below. Note that 1 does not map to any letters.



Example: Input: "23"

Output: ["ad", "ae", "af", "bd", "be", "bf", "cd", "ce", "cf"].

4. Strings CO2

Write a program to find the longest Substring without Repeating Characters

Input: abcabcbb output:3 string: abc

Input: pwwkew output:3 string: wke Note: pwke is not a substring , it is a subsequence

5. Classes and Objects CO2

Design a “farm animals” java application with the details of animals like cow, pig, horse.Consider the following details like where they stay, what they eat,the sound they make by using classes and objects.

6. Constructor overloading CO2

An organization is maintaining the data of employee according to cadre of employee with following parameters name,id,designation,salary,promotion status. Apply the constructor overloading to implement it.

7. Method overriding CO3

All the banks operating in India are controlled by RBI. (e.g. minimum interest rate, minimum balance allowed, maximum withdrawal limit etc) which all banks must follow. For example, suppose RBI has set minimum interest rate applicable to a saving bank account to be 4% annually. however, banks are free to use 4% interest rate or to set any rates above it.

Write a JAVA program to implement bank functionality in the above scenario and demonstrate the dynamic polymorphism concept. Note: Create few classes namely Customer, Account, RBI (Base Class) and few derived classes (SBI, ICICI, PNB etc). Assume and implement required member variables and functions in each class.

Testcase1:
Enter the Bank name to find the rate of Interest : RBI
RBI rate of interest is : 4%

Testcase2:
Enter the Bank name to find the rate of Interest : SBI
RBI rate of interest is : 7%

8. Interfaces: CO3

Different categories of employees are working in a software company like Regular Employees, Contract Employees and Vendors. And their pay roll is different for regular and contract employees. For the regular employees Basic pay is 25000, HRA is 15000rs and TA is 5000. For the Contract employees Basic pay is 12000 TA is 3000rs and there is no HRA. Find the monthly salary details of Employee. If input is Regular Employee display the Regular employee salary details. If input is Contract based display the Contract salary details.

TestCase1: Input: Enter Employee Id: R101
Output: Salary Details:
Basic Pay: 25000 HRA: 15000 T.A: 5000 Total Amount: 45000

9. Packages CO3

Define a package **number** and in that define **Roman class** and implement **romanToInteger()** and import the method in another class.

Input: "LVIII" **Output:** 58
Explanation: L = 50, V= 5, III = 3.

10. File Handling CO4

Write the below text in the file called sample.txt and then find the frequency count of the patterns ‘pe’, and ‘pi’

Peter Piper picked a peck of pickled peppers
A peck of pickled peppers Peter Piper picked
If Peter Piper picked a peck of pickled peppers
Where’s the peck of pickled peppers Peter Piper picked?

Expected Output:
‘pe’ – no of occurrences - 20
‘pi’ – no of occurrences – 12

11. Exception Handling CO4

Input a mobile number and check the given number is valid mobile number or not.

- A valid mobile number is a combination of (0-9) digits of length exactly 10.
- If the given Number Exceeds length of 10 raise Invalid Mobile Number-ArrayIndexOutofBounds Exception
- If the given Number less than the length of 10 raise Invalid Mobile Number – LengthNotSufficientException
- If the given Number contain any character other than digit raise Invalid Mobile Number –NumberFormatException

Sample Input	Expected Output – 1
9885089465	Valid number
98567890121	Invalid Mobile Number-ArrayIndexOutofBounds Exception
88664433	Invalid Mobile Number – LengthNotSufficientException
98ab@123	Invalid Mobile Number –NumberFormatException

12. Multi Threading CO4

Implement a Reservation system which allows the persons to book seats. Define reserve method initially with 100 seats. Now create one or more person threads to book seats. At any time it should allow only one person thread to access the reserve method.

Expected Output:

```
Person-1 entered.
Available seats: 10 Requested seats: 5
Seat Available. Reserve now :-)
5 seats reserved.
Person-1 leaving.
----------------------------------------------
Person-2 entered.
Available seats: 5 Requested seats: 2
Seat Available. Reserve now :-)
2 seats reserved.
Person-2 leaving.
----------------------------------------------
Person-3 entered.
Available seats: 3 Requested seats: 4
Requested seats not available :-(
Person-3 leaving.
----------------------------------------------
```

13. Design a mini application using the java components. C01, CO2, CO3, CO4

Data Structures Lab Using C	
CSE 218	**CREDITS:1.5**
INSTRUCTION: 3 Periods/Week	**SESSIONAL MARKS: 50**
FINAL EXAM: 3 Hours	**FINAL EXAM MARKS: 50**

Prerequisite:

- Basic knowledge about problem solving
- Require programming knowledge through C language

Course Objectives:

- ➢ The course is designed to develop skills to design and analyse simple linear and non-linear data structures.
- ➢ It strengthens the ability of the students to identify and apply the suitable data structure for the given real-world problem.
- ➢ It enables them to gain knowledge in practical applications of data structures.

Course Outcomes of the Lab:

CO1	Implement the techniques for searching and sorting (quick and merge).
CO2	Implement of stack and queue and Linked list data structures and their applications.
CO3	Implement operations like insertion, deletion, search and traversing mechanism on binary search tree
CO4	Apply BFS and DFS algorithms to implement graph traversal.

CO-PO Mapping:

Mapping		PO												PSO	
		1	2	3	4	5	6	7	8	9	10	11	12	1	2
CO	1	2	2	2	1	-	-	-	-	-	-	-	1	1	-
	2	1	2	2	1	-	-	-	-	-	-	-	1	1	-
	3	2	2	2	1	-	-	-	-	-	-	-	1	2	1
	4	2	2	2	1	-	-	-	-	-	-	-	1	2	-

Experiments:

1. Write a program to sort the given array of N elements using divide and conquer method (merge sort and quick sort algorithms) **CO1**

Constraints: 1<N<1000
Sample Input array: 87, 36, 9, 12, 24, 5, 78, 567, 456, 34, 96, 45, 39, and 89,123
Sample Output array: 5, 9, 12, 24, 34, 36, 39, 45, 78, 87, 89, 96, 123, 456, and 567

2. Write a C Program to search whether an item K present in an array of N elements (Using Linear and binary Search algorithms) **CO1**

Constraints: 1<K<1000
1<N<1000
Sample Input array: 45, 78,123, 48, 34, 89, 67, 54, and 74,543

Search Item: 34	Search Item: 343
Output: Key Found	Output: Key Not Found

3. Write a C program to store k keys into an array of size n at the location computed using a hash function, loc = key % n, where k<=n and k takes values from [1 to m], m>n. **CO1**

4. Design, Develop and Implement a C program to handle the collisions using the following collision resolution Technique **CO1**

a) **Linear probing**: In linear probing, we linearly probe for next slot, let store k keys into an array of size S at the location computed using a hash function, hash(x) where k<=n and k takes values from [1 to m], m>n.

Constraints: If slot hash(x) % S is full, then we try (hash(x) + 1) % S

If (hash(x) + 1) % S is also full, then we try (hash(x) + 2) % S

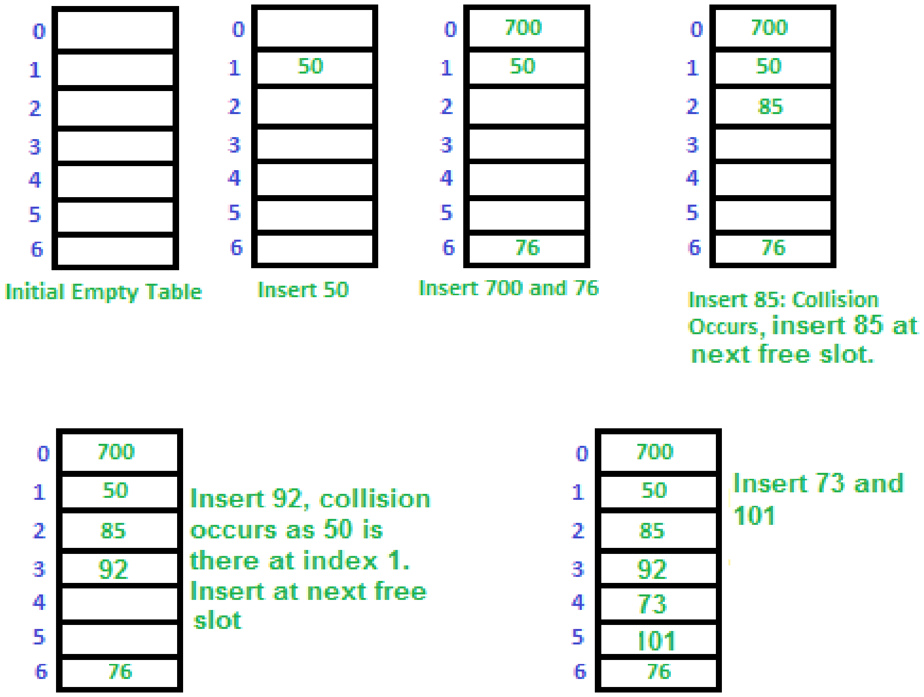
If (hash(x) + 2) % S is also full, then we try (hash(x) + 3) % S

....

....

Sample Test Case:

Let us consider a simple hash function as “key mod 7” and sequence of keys as 50, 700, 76, 85, 92, 73, 101.



0
1
2
3
4
5
6
Initial Empty Table
50
Insert 50
700
50
76
Insert 700 and 76
700
50
85
76
Insert 85: Collision Occurs, insert 85 at next free slot.
700
50
85
92
76
Insert 92, collision occurs as 50 is there at index 1. Insert at next free slot
700
50
85
92
73
101
76
Insert 73 and 101

b) Quadratic probing: Quadratic Probing we look for i2‘th slot in i’th iteration, let store k keys into an array of size S at the location computed using a hash function, hash(x) where k<=n and k takes values from [1 to m], m>n.

Constraints: let hash(x) be the slot index computed using hash function.

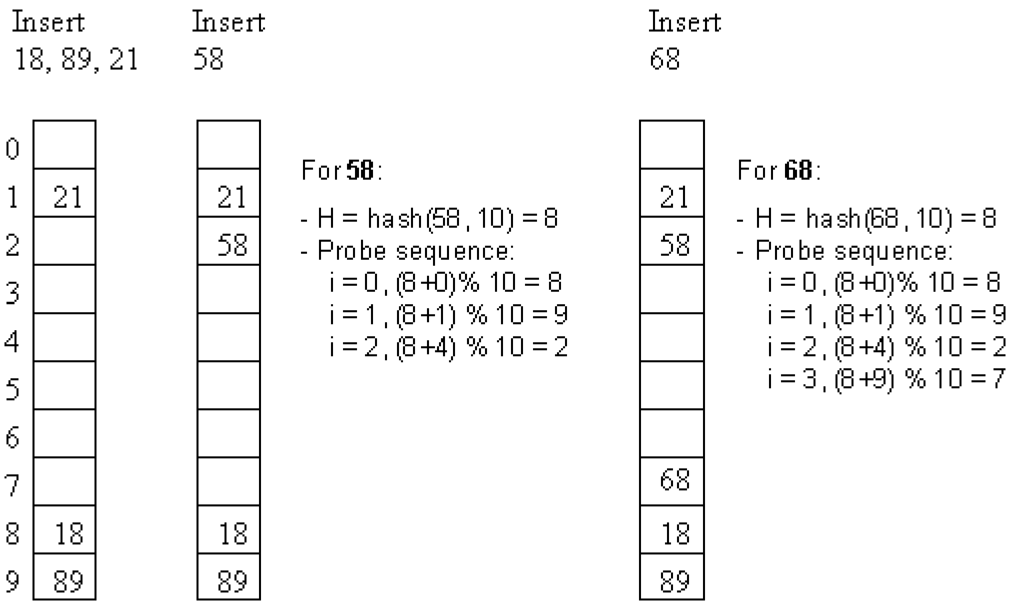
If slot hash(x) % S is full, then we try (hash(x) + 1*1) % S

If (hash(x) + 1*1) % S is also full, then we try (hash(x) + 2*2) % S

If (hash(x) + 2*2) % S is also full, then we try (hash(x) + 3*3) % S

....

Sample Test Case:

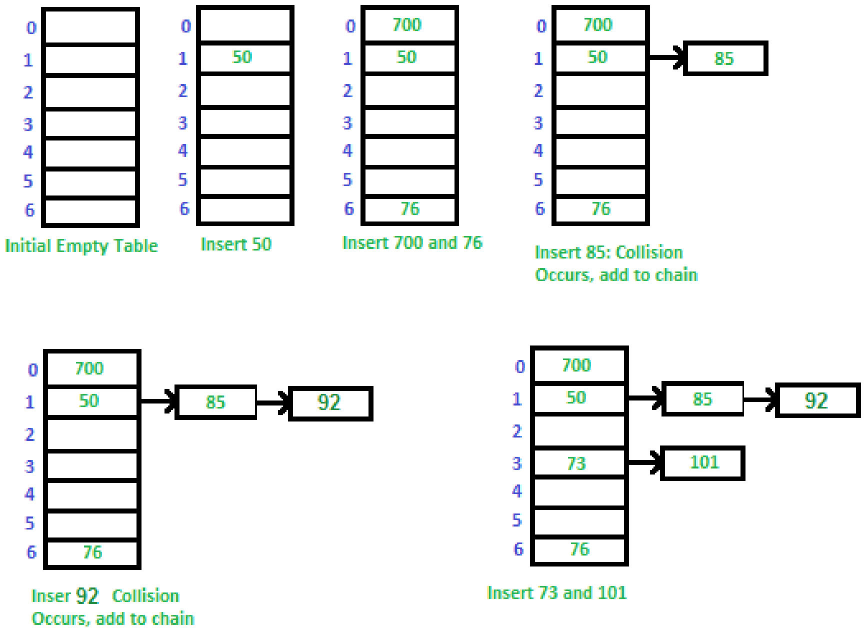


c) Separate Chaining: The idea is to make each cell of hash table points to a linked list of records that have same hash function value.

Let us store K keys into hash table of size S, where k<=n and k takes values from [1 to m], m>n.

Sample Test Case:

Let us consider a simple hash function as "key mod 7" and sequence of keys as 50, 700, 76, 85, 92, 73, 101.



5. Design, Develop and Implement a menu driven Program in C for the following. **CO2**

a) Operations on **STACK** of Integers (Array Implementation of Stack with maximum size **MAX**)

1. ***Push*** an Element on to Stack
2. ***Pop*** an Element from Stack

3. Demonstrate ***Overflow*** and ***Underflow*** situations on Stack
4. Display the status of Stack
5. Exit

b) Operations on **QUEUE** of Characters (Array Implementation of Queue with maximum size **MAX**)

1. Insert an Element on to QUEUE
2. Delete an Element from QUEUE
3. Demonstrate ***Overflow*** and ***Underflow*** situations on QUEUE
4. Display the status of QUEUE
5. Exit

Note: Support the program with appropriate functions for each of the above operations

6. Design, Develop and Implement a C program to do the following using a singly linked list. **CO2**

a) Stack- In single linked list store the information in the form of nodes .Create nodes using dynamic memory allocation method. All the single linked list operations perform based on Stack operations LIFO (last in first out).

A stack contains a top pointer. Which is "head" of the stack where pushing and popping items happens at the head of the list. first node have null in link field and second node link have first node address in link field and so on and last node address in "top" pointer.

Stack Operations:

1. push() : Insert the element into linked list nothing but which is the top node of Stack.
2. pop() : Return top element from the Stack and move the top pointer to the second node of linked list or Stack.
3. peek(): Return the top element.
4. display(): Print all element of Stack.

b) Queue- All the single linked list operations perform based on queue operations FIFO (First in first out).

In a Queue data structure, we maintain two pointers, *front* and *rear*. The *front* points the first item of queue and *rear* points to last item.

1. enQueue() This operation adds a new node after *rear* and moves *rear* to the next node.
2. deQueue() This operation removes the front node and moves *front* to the next node.
3. Display() Display all elements of the queue.

Note: Sample node information: Student Data with the fields: *USN, Name, Branch, Sem, PhNo.*

7. Design, Develop and Implement a Program in C for the following **CO2**

a) Converting an Infix Expression to Postfix Expression. Program should support for both parenthesized and free parenthesized expressions with the operators: +, -, *, /, % (Remainder), ^(Power) and alphanumeric operands.

b) Evaluation of postfix expression with single digit operands and operators: +,-, *, /, %, ^

8. Design, Develop and Implement a menu driven Program in C for the following: **CO2**

a) Circular Queue

1. Insert an Element on to Circular QUEUE
2. Delete an Element from Circular QUEUE

3. Demonstrate *Overflow* and *Underflow* situations on Circular QUEUE
4. Display the status of Circular QUEUE
5. Exit

b) Priority Queue

1. Insert an Element on to Priority QUEUE
2. Delete an Element with highest priority from Priority QUEUE
3. Demonstrate *Overflow* and *Underflow* situations on Priority QUEUE
4. Display the status of Priority QUEUE
5. Exit

Support the program with appropriate functions for each of the above operations

9. Design, Develop and Implement a menu driven C program to Perform Operations on dequeue (double ended queue) using circular array. **CO2**

a) insertFront(): Adds an item at the front of Deque.
b) insertRear(): Adds an item at the rear of Deque.
c) deleteFront(): Deletes an item from front of Deque
d) deleteRear(): Deletes an item from rear of Deque
e) getFront(): Gets the front item from queue
f) getRear(): Gets the last item from queue
g) isEmpty(): Checks whether Deque is empty or not
h) isFull(): Checks whether Deque is full or not

Support the program with appropriate functions for each of the above operations

10. Design, Develop and Implement a menu driven Program in C for the following operations on Binary Search Tree (BST) of Integers **CO3**

a. Create a BST of N Integers: 13, 3, 4, 12, 14, 10, 5, 1, 8, 2, 7, 9, 11, 6, 18
b. Traverse the BST(either inorder, predorder or postorder)
c. Search the BST for a given element (KEY) and report the appropriate message
d. Exit

11. Design, Develop and Implement a menu driven Program in C for the following operations on Binary Search Tree (BST) of Integers **CO3**

a. Create a BST of N Integers: 6, 9, 5, 2, 8, 15, 24, 14, 7, 8, 5, 2
b. Traverse the BST in Inorder, Preorder and Post Order using non-recursive functions
c. exit

12. Design, Develop and Implement a Program in C for the following operations on Graph(G) of Cities. **CO4**

a. Create a Graph of N cities using Adjacency Matrix.
b. Print all the nodes reachable from a given starting node in a digraph using DFS/BFS method

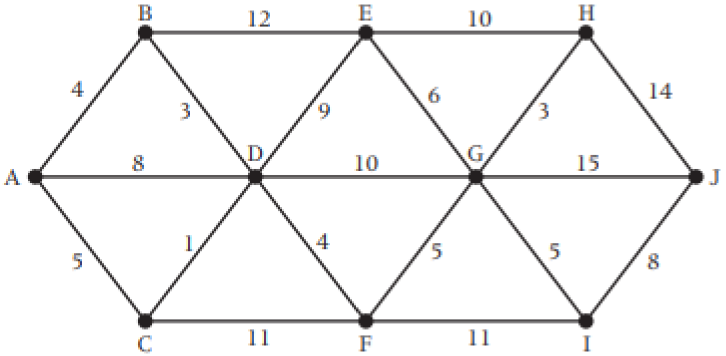
13. Design, Develop and Implement a C Program to the problem is to find shortest distances between every pair of vertices in a given edge weighted directed Graph using Warshall's Algorithm. The Graph is represented as Adjacency Matrix, and the Matrix denotes the weight of the edges (if it exists) else INF (1e7). **CO4**

Input:
The first line of input contains an integer **T** denoting the no of test cases. Then T test cases follow. The first line of each test case contains an integer V denoting the size of the adjacency matrix. The next V lines contain V space separated values of the matrix (graph). All input will be integer type.

Output:
For each test case output will be V*V space separated integers where the i-jth integer denote the shortest distance of ith vertex from jth vertex. For INT_MAX integers output INF.

Constraints:
1 <= T <= 20
1 <= V <= 100
1 <= graph[][] <= 500

14. Design, Develop and Implement a C Program to Find the shortest distance from A to J on the network below using Dijkstra's Algorithm **CO4**



Constitution of India & Intellectual Property	
CSM 219	**Credit:0**
INSTRUCTION: 2 Periods/Week	**SESSIONAL MARKS: 50**

Course Objectives

- To impart knowledge in basic concepts of Constitution of India
- To understand the fundamental principles of Intellectual Property Rights and its importance

At the end of this course, students will be able to:

	COURSE OUTCOMES	Bloom's Level
CO-1	Recognise basic knowledge about the Constitution of India	L1
CO-2	Comprehend the Fundamental Rights and Fundamental Duties of the Indian Citizen to implant morality, social values and their social responsibilities.	L2
CO-3	Familiarize with distribution of powers and functions of Local Self Government, state and central policies and amendment procedure	L2
CO-4	Recognise the fundamental principles of IPR	L1
CO-5	Appraise of IP rights like patents, industrial design, trademark, copyrights for effective protection and utilization of their innovations.	L3

SYLLABUS

Unit 1 - Introduction and Basic Information about Indian Constitution: 10hrs

Meaning of the constitution law and constitutionalism, Historical Background of the Constituent Assembly, Government of India Act of 1935 and Indian Independence Act of 1947, Enforcement of the Constitution, Indian Constitution and its Salient Features, Preamble of the Constitution.

Unit 2 - Fundamental Rights and Directive Principles 10hrs

Fundamental Rights, Fundamental Duties, Directive Principles of State Policy – Its importance and implementation, Scheme of the Fundamental Right to certain Freedom under Article 19, Scope of the Right to Life and Personal Liberty under Article 21

Unit 3 - Administrative organisation & Amendments 9hrs

Indian Federal System , Centre and State Relations , President's Rule , Constitutional Amendments , Parliamentary System in India

Unit 4 - Intellectual Property Rights 10hrs

Introduction to IPRs, Basic concepts and need for Intellectual Property – Patents, Copyrights, Geographical Indications, IPR in India and Abroad , Nature of Intellectual Property, Inventions and Innovations – Important examples of IPR

Unit 5 - Registration of IPR’s 9hrs

Meaning and practical aspects of registration of Copy Rights, Trademarks, Patents, Trade Secrets , Industrial Design registration in India and Abroad

TEXTBOOKS:

1.V. Scople Vinod, Managing Intellectual Property, Prentice Hall of India pvt Ltd, 2012

2. S. V. Satakar, ―Intellectual Property Rights and Copy Rights, Ess Publications, New Delhi,2002

3. Brij Kishore Sharma: *Introduction to the Indian Constitution*, 8th Edition, PHI Learning Pvt. Ltd.

4. Granville Austin: *The Indian Constitution: Cornerstone of a Nation (Classic Reissue),* Oxford University Press.

REFERENCES:

1. Deborah E. Bouchoux, ―Intellectual Property: The Law of Trademarks, Copyrights, Patents and Trade Secrets, Cengage Learning, Third Edition, 2012.

2.PrabuddhaGanguli,Intellectual Property Rights: Unleashing the Knowledge Economy, McGraw Hill Education, 2011.

3. Edited by Derek Bosworth and Elizabeth Webster, The Management of Intellectual Property,Edward Elgar Publishing Ltd., 2013.

4. Subhash C. Kashyap: *Our Constitution: An Introduction to India’s Constitution and constitutional Law*, NBT, 2018.

5. Madhav Khosla: *The Indian Constitution*, Oxford University Press.

6. PM Bakshi: *The Constitution of India*, Latest Edition, Universal Law Publishing

SEM-II

PROBABILITY , STATISTICS AND QUEUING THEORY	
CSM 221	**Credits : 3**
Instruction : 3 Periods /Week	**Sessional Marks : 40**
End Exam : 3 Hours	**End Exam Marks : 60**

Prerequisites:

- Elementary knowledge of Set theory, Combinations, Calculus and basic Statistics.

Course Objective:

- The objective of this course is to provide the required mathematical support in real life problems and develop probabilistic models which can be used in several areas of science and engineering.

Course Outcomes : At the end of the course student should be able to

CO - 1	Demonstrate basic principles of probability and understand a random variable that describe randomness or an uncertainty in certain realistic situation. It can be of either discrete or continuous type.
CO - 2	Comprehend concepts of discrete, continuous probability distributions and able to solve problems of probability using Binomial, Poisson, Uniform Distribution, Exponential Distribution, Normal distributions.
CO - 3	Compute simple correlation between the variables and fit straight line, parabola by the principle of least squares.
CO - 4	Analyze the statistical data and apply various small or large sample tests for testing the hypothesis.
CO - 5	Study basic characteristic features of a queuing system and acquire skills in analyzing queuing models.

CO – PO MAPPING

CO	**PO**											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1	3	2	1	-	-	-	-	-	-	-	-	1
2	3	2	1	-	-	-	-	-	-	-	-	1
3	3	2	1	-	-	-	-	-	-	-	-	1
4	3	2	1	-	-	-	-	-	-	-	-	1
5	3	2	1	-	-	-	-	-	-	-	-	1

SYLLABUS

UNIT - I : PROBABILITY & MATHEMATICAL EXPECTATIONS (12 Periods)

Introduction to Probability : Definition of random experiment– Events and sample space– Definition of probability – Addition and multiplication theorems– Conditional probability – Baye's theorem– Simple problems on Baye's theorem.

Introduction to Random variable: Discrete and continuous random variables– Distribution function of random variable– Properties, Probability mass function, Probability density function– Mathematical expectation– Properties of mathematical expectation– Moments– Moment generating function– Mean and variance.

Learning outcome: At the end of this unit, student will be able to

1. Calculate probabilities using conditional probability, rule of total probability and Bayes' theorem.
2. Explain the concept of a random variable, probability distributions and different measures of a random variable.

UNIT - II : PROBABILITY DISTRIBUTIONS (14 Periods)

Discrete Distributions : Binomial distribution – Poisson distribution – Mean, Variance, Moment Generating function and problems.

Continuous Probability Distributions: Uniform distribution – Exponential distribution, Memory less property – Normal distribution – Properties of normal distribution – Importance of normal distribution – Area properties of normal curve – MGF – Mean ,variance and simple problems.

Learning outcome: At the end of this unit, student will be able to

1. Recognize the importance of discrete probability distributions Binomial, Poisson and solve the problems about these distributions
2. Recognize the importance of continuous distributions Exponential ,Uniform and Normal and Exponential Distribution and solve the problems about these distributions.

UNIT - III: CURVE FITTING, CORRELATION AND REGRESSION (10 Periods)

Curve Fitting : Principle of least squares – Method of least squares –Fitting of straight lines – Fitting of second degree curves and exponential curves.

Correlation : Definition – Karl pearson's coefficient of correlation – Measures of correlation– Rank correlation coefficients.

Regression : Simple linear regression – Regression lines and properties.

Learning outcome: At the end of this unit, student will be able to

1. Understand the concept of principle of least squares for curve fitting of straight line ,second degree curve and exponential curve.
2. Calculate Pearson's correlation coefficient, Spearman's rank correlation coefficient and form the regression lines.

UNIT - IV : TESTING OF HYPOTHESIS (14 Periods)

Introduction– Null hypothesis – Alternative hypothesis – Type –I&II errors – Level of significance –

Critical region – Confidence interval – One sided test – Two sided test.

Small Sample Tests: Students t - distribution and its properties – Test of significance difference between sample mean and population mean – Difference between means of two small samples – F- Distribution– Test of equality of two population variances – Chi-square test of goodness of fit .

Large sample Tests: Test of significance of large samples – Tests of significance difference between sample proportion and population proportion & difference between two sample proportions – Tests of significance difference between sample mean and population mean & difference between two sample means.

Learning outcome: At the end of this unit, student will be able to

1. Define null hypothesis, alternative hypothesis, level of significance, test statistic, *p* value, and statistical significance.
2. Perform and analyze hypotheses tests of means, proportions and χ^2 −test using both one- and two-sample data sets.

UNIT - V : QUEUING THEORY (10 Periods)

Structure of a queuing system – Operating characteristics of queuing system – Transient and steady states– Terminology of Queuing systems – Arrival and service processes – Pure Birth-Death process deterministic queuing models – M/M/1 model of infinite queue – M/M/1model of finite queue.

Learning outcome: At the end of this unit, student will be able to

1. Analyze basic characteristic features of a queuing system and acquire skills in analyzing queuing models.
2. Solve problems on queuing models M/M/1 and M/M/n.

TEXT BOOK :

1. **T. Veerarajan**, *Probability, Statistics and Random Processes*, Tata McGraw Hill Publications.

REFERENCE BOOKS:

1. **Kishor S. Trivedi**, *Probability & Statistics with Reliability, Queuing and Computer Applications*, Prentice Hall of India .
2. **B. S. Grewal**, *Higher Engineering Mathematics*, 43rd edition, Khanna publishers, 2017.
3. **Sheldon M. Ross,** *Probability and Statistics for Engineers and Scientists,* Academic Press.
4. **S C Gupta and V.K.Kapoor**, *Fundamentals of Mathematical Statistics*.

ARTIFICIAL INTELLIGENCE	
Course Code: CSM222	**Credits : 3**
Instruction : 2 Periods & 1 Tut/Week	**Sessional Marks : 40**
End Exam : 3 Hours	**End Exam Marks : 60**

Prerequisites: Probability and Discrete Mathematics, Basic knowledge on algorithms and programming language

Course Objectives: The course should enable the students:

- To acquire knowledge on basic principles, techniques, and applications of artificial intelligence.
- To have better insight into knowledge representation, problem-solving, and learning methods in solving particular engineering problems.

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

1.	Interpret the fundamental concept of Artificial Intelligence and solve the problems by applying suitable searching methods.
2.	Illustrate the problems using logic based knowledge representation.
3.	Apply statistical reasoning models to solve the problems.
4.	Summarize the concept of planning and natural language processing.
5.	Classify the various approaches of the Expert system.

Mapping of Course Outcomes with Program Outcomes:

Mapping		**PO**												**PSO**	
		1	2	3	4	5	6	7	8	9	10	11	12	1	2
CO	1	2	2	1	-	-	-	-	-	-	1	-	2	-	2
	2	2	1	1	1	-	-	-	-	-	-	-	2	-	2
	3	2	1	-	-	-	-	-	-	-	-	-	2	-	2
	4	1	-	2	1	-	2	-	-	1	1	-	2	-	2
	5	1	-	2	1	-	2	-	-	1	1	-	2	-	2

SYLLABUS

Unit –I **(12 Hours)**

Introduction to Artificial Intelligence: History, Turing Test, AI problems, Artificial Intelligence Techniques, , Applications of AI; **Problems, Problem Spaces and Search:** Defining the problem as State space search, production systems, Problem characteristics, Production System characteristics, Issues in the Design of Search Programs; **Searching Techniques: Uninformed Search Strategies:** Breadth first search, Depth first search, **Heuristic Search Techniques:** Generate & Test, simple and Steepest-Ascent Hill Climbing, Best First search, A* Algorithm, Constraint satisfaction, Means-End Analysis.

Learning Outcomes: At the end of this unit, Student will be able to

1. Interpret the basics of Artificial Intelligence Techniques, Problem Characteristics, and production system.
2. Demonstrate searching strategies and identify the searching algorithms to solve the real world problems.

Unit –II **(10 Hours)**

Knowledge Representation: Issues-Representation and Mappings**,** approaches to knowledge representation, Issues in Knowledge representation.

Using Predicate Logic: Representing simple facts in logic, Representing Instance and ISA relationship, Computable Functions and Predicates, Resolution, Convert into clause forms, unification algorithm.

Representing Knowledge using Rules: Procedural vs Declarative Knowledge, Logic Programming, Forward vs Backward reasoning, Matching, Control Knowledge.

Learning Outcomes: At the end of this unit, Student will be able to

1. Describe the concept of knowledge representation approaches, rules and reasoning.
2. Determine the knowledge representations in the form of predicate logic

Unit-III **(10 Hours)**

Statistical Reasoning: Probability and Bayes' Theorem, Certainty Factor and Rule based System, Bayesian Networks, Dempster-Shafer Theory

Weak Slot Filler Structures: Semantic nets, Frames.

Strong Slot Filler Structures: Conceptual dependencies, Scripts

Learning Outcomes: At the end of this unit, Student will be able to

1. Demonstrate the concepts of statistical reasoning approaches.
2. Examine the knowledge on slot filler structures

Unit-IV **(10 Hours)**

Planning: Overview, An Example Domain: The Blocks World, Components of a planning system, Types of Planning: Goal Stack planning, Non-Linear Planning, Hierarchical Planning **Introduction to Natural Language Processing:** Introduction, Syntactic Processing, Semantic Analysis, Discourse and Pragmatic Processing, Statistical Natural Language Processing, Spell Checking, Applications of NLP.

Case study: Text Data Preprocessing.

Learning Outcomes: At the end of this unit, Student will be able to

1. Compare the planning approach for the real world scenario.
2. Describe the various steps involved in Natural language processing.

Unit-V **(8 Hours)**

Strong Method Problem Solving: Introduction, Overview of Expert System Technology, Rule Based Expert System, Model Based, Case Based and Hybrid Systems.

Learning Outcomes: At the end of this unit, Student will be able to

1. Describe the concept of the Expert system.
2. Outline the various approaches of the Expert System.

Text Books:

1. Rich E & Knight K, “Artificial Intelligence”, 4th Edition, Tata McGraw hill, 2010. (UNIT:I-IV)
2. George F Luger, “Artificial Intelligence: Structure and strategies of complex problem solving”, 6th Edition, Addison Wisley, 2009. (UNIT-V)

References:

1. Stuart Russell and Peter Norvig, “Artificial Intelligence: A Modern Approach”, 2nd Edition, Pearson Education, 2010.
2. DAN.W. Patterson, “Introduction to A.I and Expert Systems”, 1st Edition, PHI, 2007.

Web References:

1. https://nptel.ac.in/courses/106/105/106105077/
2. https://www.udemy.com/course/natural-language-processing-with-python-and-nltk/

OPERATING SYSTEMS	
CSM 223	**Credits : 3**
Instruction **:** 3 Periods /Week	Sessional Marks : 40
End Exam : 3 Hours	End Exam Marks : 60

Prerequisites: Basic programming language and Computer Organization.

Course Objectives:

- To understand the main components of an operating system and their functions.
- To understand the basic concept of shell programming.
- To learn the mechanism of an operating system as process manager, memory manager, device manager and file manager.
- To understand the concept of protection related to operating systems.

Course Outcomes:

At the end of the course the student will be able to

CO	At the end of the course the student will be able to
1.	Illustrate the structure of the operating system, functionality and services provided by the operating system and interpret the basic concept of shell programming, process state and state transitions.
2.	Implement the CPU Scheduling algorithms (Pre-emptive and Non Pre-emptive) and demonstrate the concept of Process synchronization.
3.	Demonstrate the concept of resource allocation. Apply and analyze various memory management mechanisms for contiguous and non-contiguous memory.
4	Demonstrate the structure and organization of the file systems and analyze the implementation of the file systems.
5	Analyze the secondary storage structure, protection of the system.

Mapping of Course Outcomes with Program Outcomes:

CO	PO												PSO	
	1	2	3	4	5	6	7	8	9	10	11	12	1	2
1	1	1	1	-	-	-	-	-	-	-	-	-	-	-
2	2	2	2	1	-	-	-	-	-	-	-	-	1	-
3	2	2	2	1	-	-	-	-	-	-	-	-	1	-
4	2	2	2	1	-	-	-	-	-	-	-	-	1	-
5	2	2	2	1	-	-	-	-	-	-	-	-	-	-

SYLLABUS

UNIT I **(12 Hours)**

Introduction to OS: Operating system Definition, Operating system Functionalities, Types of Operating system, operating system structures, system calls, system programs.

Introduction to Shell Programming: Commands and Shell script.

Processes: Process concept, Process scheduling, Operations on processes, Inter process communication, Communication in client-server systems.

Threads: Overview, Multithreading models.

Learning outcomes: At the end of this Unit, Students will be able to

1. Define the responsibilities of an operating system and implement the basic shell programs.
2. Demonstrate the different modes of communication among processes and multi threading models.

UNIT II **(10 Hours)**

CPU Scheduling: Scheduling criteria, Scheduling algorithms, Algorithm Evaluation.

Process Synchronization: The critical-section problem, Peterson's solution, Synchronization hardware, Mutex Locks, Semaphores, Classic problems of synchronization, Monitors.

Case Study: Linux operating system: Process Management.

Learning outcomes: At the end of this Unit, Students will be able to

1. Analyze the CPU scheduling algorithms and their performance evaluation.
2. Implement the different solutions for process synchronization.

UNIT III **(12 Hours)**

Deadlock: System model, Deadlock characterization, Methods for handling deadlocks, Deadlock prevention, Deadlock avoidance, Deadlock detection, Recovery from deadlock.

Memory Management: Background, Swapping, Contiguous memory allocation, Segmentation, Paging, Structure of the page table.

Virtual Memory: Background, Demand paging, Page replacement, Allocation of frames, Thrashing.

Case Study: Linux operating system: Memory Management.

Learning outcomes: At the end of this Unit, Students will be able to

1. Define the concept of deadlock and Identify the different ways to handle deadlock like prevention, detection, avoiding and recovery.
2. Distinguish between contiguous and non-contiguous memory allocation methods in memory management.

UNIT IV **(10 Hours)**

File Systems Interface: File concept, Access methods, Directory structure, File system mounting, File Sharing, Protection.

Implementing File-Systems: File system structure, File system implementation, Directory implementation, Allocation methods, Free-space management, Efficiency and performance, Recovery.

Learning outcomes: At the end of this unit, students will be able to

1. Demonstrate the concept of file system, various file access methods and Protection in files.
2. Identify and implement the file system and recovery.

UNIT V **(10 Hours)**

Secondary Storage Structure: Mass storage structures, Disk structure, Disk attachment, Disk scheduling, Disk management, Swap space management.

Protection: Goals of protection, Principles of protection, Domain of protection, Access matrix, Implementation of access matrix, Access control, Revocation of access rights.

Learning outcomes: At the end of this unit, students are will be to

1. Demonstrate the concept of mass storage structures and Analyze the various disk scheduling algorithms
2. State the goal and principles of protection and implement the access matrix.

TEXT BOOKS

1. Silberschatz, Galvin and Gagne, "Operating System Principles", 9th Edition, Wiley India Pvt Ltd, 2015.
2. Sumitabha Das, "Unix Concepts and Applications", 4th Edition. TMH, 2006.
3. Yashwanth Kanitkar, "Unix Shell programming", 1st Edition, BPB Publisher, 2010.

REFERENCES

1. Andrew S. Tanenbaum, "Modern Operating Systems", 4th Edition, Pearson Education, 2015.
2. William Stalling, "Operating Systems: Internals and Design Principles", 9th edition, PHI, 2018.

WEB REFERENCES:

1. https://nptel.ac.in/courses/106/106/106106144/
2. https://nptel.ac.in/content/storage2/courses/106108101/pdf/PPTs/Mod_13.pdf

PYTHON PROGRAMMING (R20- 2-2 AI-ML & DS)	
Code: CSM 224	**Credits : 3**
Instruction :3 Periods /Week	**Sessional Marks : 40**
End Exam : 3 Hours	**End Exam Marks : 60**

Prerequisites: Basic knowledge of any programming language concepts like conditional statements, iterative statements, functions, OOP's concepts. and basic mathematics.

Course Objectives:

- ➢ Describe the core syntax and semantics of Python programming language.
- ➢ Illustrate the process of structuring the data using lists, dictionaries, tuples, strings and sets.
- ➢ Discover the need for working with the functions, modules and packages.
- ➢ Infer the Object-oriented Programming concepts in Python.
- ➢ Familiarize the advanced concepts like regular expressions, date and time. Able to handle abnormal termination of the python scripts.

Course Outcomes:

By the end of the course, the student will be able to:	
1.	Interpret the fundamental Python syntax and semantics and able to solve, test and debug python programs
2.	Fluency in the use of Python control flow statements and determine the methods to create and manipulate Python programs by utilizing the data structures like lists, dictionaries, tuples, strings and sets.
3.	Express proficiency in the handling of functions, modules, packages and handle abnormal termination of the programs.
4.	Articulate the Object-Oriented Programming concepts such as encapsulation, inheritance and polymorphism as used in Python.
5.	List the usage and application of regular expressions, date and time. Handle abnormal termination.

Mapping of Course Outcomes with Program Outcomes:

CO	**PO**												**PSO**	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
1	2	2	3	-	-	-	-	-	-	-	-	-	1	1
2	2	2	2	-	-	-	-	-	-	-	-	-	1	1
3	2	1	2	-	-	-	-	-	-	-	-	-	-	-
4	2	2	3	3	-	-	-	-	-	-	-	-	1	1
5	3	2	3	-	-	-	-	-	-	-	-	-	1	1

Syllabus

Unit-I **(8 Hours)**

Introduction to Data Science - Why Python?, Introduction to Python: History and Features of Python, Variables, Reserved words, Data types, Operators, Input and output, Indentation, Comments and Documentation, Lists, Set, Dictionaries, Tuple. Strings: Creating strings and basic operations on strings, string slicing and indexing.

Learning Outcomes: At the end of this unit, Students are able to

1. Analyse fundamental advantages of python over the other programming languages and Solve, test and debug basic problems using python scrip
2. Manipulate python programs by using the python data structures like lists, dictionaries, tuples, strings and sets.

Unit-II **(10 Hours)**

Decision Control Statements, loop Control Statements, Functions: Defining a function, Calling a function, returning multiple values from a function, formal and actual arguments, positional arguments, default arguments, recursive functions. lambda functions, modules, Statistical functions like mean ( ),fmean( ),harmonic_mean ( ), median( ), median__low( ) , median__high(), median__grouped( ), mode( ), multimode( ), quantiles ( ), geometric_mean( )

Learning Outcome: At the end of this Unit the student will be able to

1. Implement Flow control statements required real world problems.
2. Resolve real world problems using python functions and statistical functions.

Unit-III **(12 Hours)**

NumPy packaage: Arrays and Vectorized Computation- The NumPy ndarray- Creating ndarrays- Data Types forndarrays- Arithmetic with NumPy Arrays- Basic Indexing and Slicing - Boolean Indexing-Transposing Arrays and Swapping Axes.

Pandas package: Introduction to pandas Data Structures, Data frame, Index objects, Re indexing, Dropping entries from axis, Indexing, selection, and filtering, Arithmetic and data alignment, Function application and mapping, Sorting and ranking, Axis indexes with duplicate values, Summarizing and Computing Descriptive Statistics, Correlation and Covariance, Unique Values, Value Counts, and Membership.

Learning Outcome: At the end of this Unit the student will be able to

1. Familiarize the usage of Modules and packages to enhance the problem solving
2. Resolve real world problems using NumPy and Pandas

Unit-IV **(10 Hours)**

Handling Missing Data-Filtering Out Missing Data, Filling in Missing Data, Hierarchical Indexing. Data Transformation: Removing Duplicates, Replacing Values, Renaming Axis Indexes. Discretization and Binning: Detecting and Filtering Outliers.

Visualizing Data: matplotlib package: Bar Charts, Line Charts, Scatter plots. Histograms and Density Plots, Colors, Markers, and Line Styles, Visualizing iris Data, Seaboran package, Ticks, Labels, and Legends.

Learning Outcome: At the end of this Unit the student will be able to

1. Process the real world data pre-processing
2. Analyse the real world data using visualizing tecniques

Unit-V **(10 Hours)**

Exception Handling in Python, What is an Exception?, Syntax for Exception Handing, Handling Single Exception, Handling Multiple Exceptions.

The regex package, Regular expression methods: findall, finditer, match, search, split, sub, subn. Date and Time Data Types and Tools: The datetime package, Types in datetime module, Datetime format specification, Locale-specific date formatting, Time Series Basics, Date Ranges, Frequencies, and Shifting, Base Time Series Frequencies, Time Zone Handling.

Learning Outcome: At the end of this Unit the student will be able to

1. Problem solving with the usage of exceptions
2. Resolve the problems like pattern matching and manipulation of time and date

Text Books

1. Wes McKinney, "Python for Data Analysis",O'REILLY,2012.
2. Reema, Thareja, "Python Programming: Using Problems Solving Approach", Oxford University Press, 2017
3. Gowrishankar S, Veena, "Introduction to Python Programming", CRC Press/Taylor & Francis, 2019.

Reference Books

1. Adnan Aziz, Luciano Ramalho, "Elements of Programming Interviews in Python: The Insiders' Guide, Fluent Python: Clear, Concise, and Effective Programming", Createspace Independent Pub, 2016.
2. Vamsi Kurama, Pearson, "Python Programming : A Modern Approach", Pearson India**,** 2017
3. Jake VanderPlas, "Python Data Science Handbook: Essential Tools for Working with Data", Kindle Edition, O'REILLY,2016.
4. Wesley J. Chun, "Core Python Programming", Prentice Hall,2006.

THEORY OF COMPUTATION AND COMPILERS	
CSM 225	**Credits: 3**
Instruction:2 Periods & 1 Tut/Week	**Sessional Marks : 40**
End Exam : 3 Hours	**End Exam Marks : 60**

Prerequisites:

- The students are expected to have a strong background in the fundamentals of discrete mathematics (symbolic logic, set, graph, basic proof techniques, etc.), algorithms and data structures.
- Some knowledge of programming languages and computer architecture will be helpful.
- Describes how a programming language works, how input is converted into output from the machine hardware level and various phases of compiler

Course Objectives:

- ➢ To introduce the fundamental concepts of formal languages and theory of computation
- ➢ Illustration of grammars and their role in compilers and various parsing techniques
- ➢ Introduce the major concept areas of language translation and compiler design.
- ➢ Enrich the knowledge in various phases of compiler and its use, intermediate code generation, optimization techniques, machine code generation, and use of symbol table.
- ➢ Focus on various storage allocation schemes
- ➢ Enforces various schemes for optimizing code

Course Outcomes:

By the end of the course, the student will be able to:	
1	Acquire the knowledge of understanding the fundamentals of the core concepts in automata theory and construct DFA and NFA.
2	Describe the Context free languages and grammars, and also Normalizing CFG
3	Analyze different phases of a compiler, Language Translators, importance of Lex tool and YACC and working with parser techniques
4	Describe the Intermediate code forms, various Storage Allocation, symbol table and run time storage is managed.
5	Analyze different code optimization techniques, Basic blocks, Machine dependent code generation techniques and importance of DAG.

Mapping of Course Outcomes with Program Outcomes:

CO	**PO**												**PSO**	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
1	1	1	1	-	2	-	-	1	2	1	-	2	1	1
2	2	2	2	2	2	2	2	1	2	1	2	2	1	1
3	2	2	2	2	2	2	2	1	2	1	2	2	1	1
4	2	2	2	1	2	1	1	1	2	1	1	2	1	1
5	2	2	2	1	2	1	1	1	2	1	1	2	1	1

SYLLABUS

UNIT I **15 Hours**

Finite Automata - Alphabets, Strings and Languages, Deterministic finite Automata (DFA)- Formal Definition, Simplified notation: State transition graph, Transition table, Language of DFA, Nondeterministic finite Automata (NFA), NFA with epsilon transition, Equivalence of NFA and DFA, Minimization of Finite Automata, Moore and Mealy machine, Equivalence of Moore and Mealy Machine, Applications and Limitation of FA.

Regular Expression (RE)- Regular expression (RE) Definition, Operators of regular expression and their precedence, Regular expression to FA, DFA to Regular expression, Arden Theorem, Pumping Lemma for regular Languages. Application of Pumping Lemma, Closure properties of Regular Languages

Learning Outcome: At the end of this Unit the students will be able to

1. Design Finite Automata's for different Regular Expressions and Languages.
2. Use the Pumping lemma for proving that languages are not regular.

UNIT II **10 Hours**

Context Free Grammar (CFG) - Definition, Examples, Derivation, Derivation trees, Ambiguity in Grammar, Ambiguous to Unambiguous CFG, Useless symbols, Simplification of CFGs, Normal forms for CFGs: CNF and GNF, Closure proper ties of CFLs.

Learning Outcome: At the end of this Unit the students will be able to

1. To construct context free grammar for various languages.
2. Describe the language accepted by automata or generated by a regular expression or a context-free grammar.

UNIT III **15Hours**

Overview of Compilers: Compilers, Analysis of the source program, Phases of a compiler, Cousins of the Compiler, Grouping of Phases and Compiler construction tools, Lexical Analysis, Role of Lexical Analyzer, Input Buffering – Specification of Tokens.

Syntax Analysis- Role of the parser, Parse trees and derivations. Left recursion and left factoring. Top-down and bottom-up parsing.

Learning Outcome: At the end of this Unit the students will be able to

1. A position to understand the different types of parsing techniques and should be in a position to solve the problem
2. Write the code by using YACC and lex tools.

UNIT IV **10 Hours**

Syntax directed translation: Semantic analysis, Syntax-directed translation,Various Intermediate code generation, Type checking.

Symbol Tables: Contents of a table, data structures for symbol tables, representing scope information.

Storage Organization: Storage Allocation strategies, Access to non-local names, Parameter Passing, Error detection and recovery.

Learning Outcome: At the end of this Unit the students will be able to

1. Perform the operations of semantic analysis and different notations of intermediate code forms
2. Analyze the run-time complexity of a few simple algorithms.

UNIT V **10 Hours**

Code Optimization: Issues in the design of code optimization, the principal source of optimization, loop optimization, DAG representation of basic blocks, Loop optimization and peephole optimization

Code Generation: Object programs, problems in code generation, machine model, simple code generator, register allocation and assignment, code generation from DAG"s, peephole optimization.

Learning Outcome: At the end of this Unit the students will be able to

1. Analyze the program and minimize the code by using optimization techniques which helps in reducing the no. of instructions in a program and also utilization of registers in an effective way.
2. Use different compiler optimization schemes.

Text Books:

1. AlfredAho, Ravi Sethi, Jeffrey D Ullman, "Compilers Principles, Techniques and Tools", Pearson Education Asia, 2003.
2. J.P. Bennet, "Introduction to Compiler Techniques", Second Edition, Tata McGraw-Hill, 2003

Reference Books:

1. Hopcroft H.E. and Ullman J. D. "Introduction to Automata Theory Languages and Computation". Pearson Education
2. Mishra, Chandra Shekaran "Theory of Computer Science, Automata languages and computation", 2/e, PHI
3. A.V. Aho . J.D.Ullman "Principles of Compiler Design", PEA
4. Kenneth C. Louden, "Compiler Construction: Principles and Practice", Thompson Learning, 2003

ONLINE WEB RESOURCES:

1. https://nptel.ac.in/courses/106/103/106103070/
2. https://nptel.ac.in/courses/106/104/106104123/

DESIGN AND ANALYSIS OF ALGORITHMS	
CSM 226	Credits : 3
Instruction : 2 Periods & 1 Tut/Week	Sessional Marks : 40
End Exam : 3 Hours	End Exam Marks : 60

Prerequisites:

- Some programming skills and a good back ground in discrete mathematics, data structures and probability will be very helpful.

Course Objectives:

- Student will understand the basic design concepts (e.g., pseudo code, specifications, top-down design).
- Student will learn the different algorithm design strategies (procedural knowledge).
- Student can acquire the knowledge to solve the complexities of different problems.
- Student will able to choose appropriate design strategies for different problems.

Course Outcomes:

By the end of the course, the student will be able to:	
1.	Demonstrate knowledge about basic design concepts (e.g., pseudo code, specifications, top-down design).
2.	Use and explain the algorithms for different design strategies.
3.	Apply the algorithms and design strategies to solve problems.
4.	Analyze the complexities of various problems in different domains.
5.	Categorize the notions of P and NP problems, NP complete and NP-hard problems.

Mapping of Course Outcomes with Program Outcomes:

Mapping		PO												PSO	
		1	2	3	4	5	6	7	8	9	10	11	12	1	2
CO	1	3	3	-	-	-	-	-	-	2	-	-	-	1	-
	2	3	3	-	-	-	-	-	-	2	-	-	-	-	-
	3	3	3	2	-	-	-	-	-	3	-	-	-	-	-
	4	3	3	2	3	-	-	-	-	1	-	-	-	1	-
	5	2	2	-	2	-	-	-	-	-	-	-	-	-	-

SYLLABUS

UNIT-I: **12 periods**

Introduction :

Introduction, Steps for algorithmic problem solving , Important Problem Types Analysis framework (Orders of growth, Cases), Asymptotic Notations and Efficiency Classes, Mathematical Analysis for recursive Algorithms and Non-recursive Algorithms, Empirical Analysis, Algorithm Visualization.

Case Study: Pseudo code Conventions, Time and Space Complexities

Learning Outcomes:

1. Argue the correctness of algorithms using inductive proofs and invariants.
2. Analyze worst-case running times of algorithms using asymptotic analysis.

UNIT-II: 14 periods

Brute Force:

Brute Force- Selection and Bubble sort, Sequential Search, String Matching, Closest- Pair, Convex Hull Problems, Exhaustive Search -Travelling Salesman problem, knapsack problem, Assignment Problem.

Decrease and Conquer:

Decrease by a constant: Insertion Sort, Algorithms for generating combinatorial problems, Decrease by constant factor algorithms, Variable size decrease.

Divide-and-Conquer :

Merge sort, Quick sort, Binary Search, Multiplication of large integers and Stassen's Matrix Multiplication, Closest- Pair, Convex Hull Problems.

Learning Outcomes:

1. Describe the divide-and-conquer paradigm and explain when an algorithmic design situation calls for it.
2. Recite algorithms that employ this paradigm.

UNIT-III: 12 periods

Transform and conquer:

Presorting, Gauss Elimination, Balanced Trees –2-3 Trees, Heap sort, Horner's rule and binary exponentiation, Problem reduction.

Dynamic Programming:

Computing a Binomial Coefficient, Warshall's and Floyd's Algorithm, Optimal Binary Search Trees, The Knapsack Problem and Memory Functions.

Learning Outcomes:

1. Describe the dynamic-programming paradigm and explain when an algorithmic design situation calls for it.
2. Recite algorithms that employ this paradigm. Synthesize dynamic-programming algorithms, and analyze them.

UNIT-IV: 12 periods

Greedy Technique: Prim's Algorithm, Kruskal's Algorithm, Dijkstra's Algorithm – Huffman Trees. **Space And Time Tradeoffs:** Sorting by computing, Input Enhancement in String Matching-Horspool's Algorithm, Boyer-Moore Algorithm, Hashing, B-Trees

Learning Outcomes:

1. Describe the greedy paradigm and explain when an algorithmic design situation calls for it.
2. Recite algorithms that employ this paradigm. Synthesize greedy algorithms, and analyze them.

UNIT-V: 14 periods

Limitations of Algorithm Power: Lower-Bound Arguments, Decision Trees, P, NP and NP complete problems, Challenges of Numerical Algorithms

Coping with the limitations of Algorithms Power – Backtracking, Branch-and-Bound
Case study for Backtracking: Graph Coloring

NP Problems - Approximation Algorithms for NP-hard Problems, Algorithms for solving Nonlinear Equations.

Learning Outcomes:

1. Explain what competitive analysis is and to which situations it applies.
2. Perform competitive analysis.

Text Books:

1. Anany Levitin,*"Introduction to Design & Analysis of Algorithms*", 2003, Pearson Education, New Delhi.

Reference Books :

1. Ellis Horowitz, S. Sahni et.al,"*Fundamentals of Computer Algorithms*",2001,Galgotia Pub.
2. Thomas H. Corman, Charles E. Leiserson, Ronald R. Rivest & Clifford Stein, "*Introduction to Algorithms*" Prentice Hall of India, New Delhi
3. Aho, Hopcroft & Ullman,"*The Design and Analysis of computer Algorithms*",2003 Pearson Education, New Delhi
4. Gilles Brassard & Paul Bratley,"*Fundamentals of Algorithmic*", Prentice Hall of India,
New Delhi

Web Resources:

1. http://nptel.ac.in/courses/106101060/
2. https://www.edx.org/course/subject/data-analysis-statistics
3. https://www.udacity.com/courses/data-science
4. https://www.coursera.org/specializations/algorithms

Python Programming Lab	
Code: CSM227	**Credits : 1.5**
Instruction :3 Periods /Week	**Sessional Marks : 50**
End Exam : 3 Hours	**End Exam Marks : 50**

Course Objectives:

- To train the students in solving computational problems using python programming.
- To understand the fundamentals of numpy, pandas and matplotlib packages for data analysis.
- Practical way of building supervised and unsupervised models. Evaluation of models.

Course Outcomes:

By the end of the course, the student will be able to:	
1.	Able to work with basic data structures, operators and conditional and control statements
2.	Develop application programs using functions (statistical), packages and modules
3.	Explore numpy package and automate tasks using numpy package
4.	Explore pandas, matplotlib packages. Write example programs to pre process data, visualize the data and apply supervised and unsupervised models on real world data sets

CO	PO												PSO	
	1	2	3	4	5	6	7	8	9	10	11	12	1	2
1	2	3	2	-	-	-	-	-	-	-	-	-	-	-
2	1	1	2	-	-	-	-	-	-	-	-	-	-	-
3	1	1	2	-	-	-	-	-	-	-	-	-	-	-
4	1	1	2	-	-	-	-	-	-	-	-	-	2	-

1. Write a python script to performdifferent Arithmetic Operations on numeric data types in Python (CO1)

   Sample Input:

   Sample input: 2 770
   Sample output:772

2. Write a python Program to Demonstrate a Function with and without Arguments (CO1)
3. Write a python program to create, append, and remove lists in Python. (CO1)
4. Write Python Program to Conduct a Linear Search for a Given Key Number in the List and Report Success or Failure. (CO1)

   Sample Input 1:

   Sample input [10,14,19,25,27,31,34,43,45,52]
   key = 27
   Element found at index: 4

   Sample Input 2:

   Sample input [10,14,19,25,27,31,34,43,45,52]
   key = 16

```
Element not found
```

5. Write a python program to demonstrate working with Tuple in Python. (CO1)
6. Write a python program to demonstrate working with dictionaries in Python. (CO1)
7. Program to Dynamically Build User Input as dictionaries. (CO1)
8. Write a python program to compute summary statistics such as mean, median, mode, standard deviation and variance of the given different types of data. (CO2)

   Sample Output:

   ```
   Original array:[0 1 2 3 4 5]

   Mean: 2.5
   std: 1
   variance: 2.9166666666666665
   ```

9. Write a python program to calculate the sum of every column in a NumPy array (CO3)

   Sample Output:

   ```
   [[ 1  2  3]
    [ 4  5  6]
    [ 7  8  9]]
   Sum of every column is
   [ 12 15 18 ]
   ```

10. Write a python program to calculate the sum of every row in a NumPy array (CO3)

    Sample Output:

    ```
    [[ 1  2  3]
     [ 4  5  6]
     [ 7  8  9]]

    Sum of every row is
    [ 6 15 24]
    ```

11. Write a NumPy program to compute the 80th percentile for all elements in a given array along the second axis. (CO3)

    Sample Output:

    ```
    Original array:
    [[ 0  1  2  3  4  5]
     [ 6  7  8  9 10 11]]

    80th percentile for all elements of the said array along the second axis:
    [ 4. 10.]
    ```

12. Write a NumPy program to compute the median of flattened given array (CO3)

    Sample Output:

    ```
    Original array:
    [[ 0  1  2  3  4  5]
     [ 6  7  8  9 10 11]]
    ```

```
Median of said array:
5.5
```

13. Write a NumPy program to compute the weighted of a given array (CO3)
    <u>Sample Output:</u>

```
Original array:
[0 1 2 3 4]

Weighted average of the said array:
2.6666666666666665
```

14. Write a NumPy program to compute the covariance matrix of two given arrays (CO3)
    <u>Sample Output:</u>

```
Original array1:
[3 6 8]

Original array2:
[2 4 7]

Covariance matrix of the said arrays:
 [[6.33333333 6.16666667]
 [6.16666667 6.33333333]]
```

15. Write a NumPy program to compute cross-correlation of two given arrays (CO3)
    <u>Sample Output:</u>

```
Original array1:
[4 7 9]

Original array2:
[2 6 8]

Cross-correlation of the said arrays:
[[6.33333333 7.66666667]
 [7.66666667 9.33333333]]
```

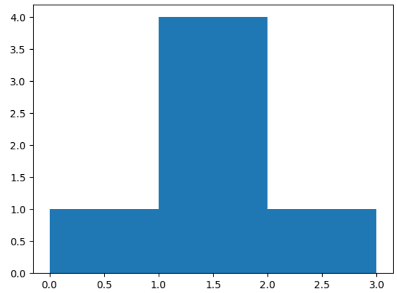
16. Write a Python NumPy program to compute the weighted average along the specified axis of a given flattened array (CO3)
    <u>Sample Output:</u>

```
Original flattened array:
[[0 1 2]
 [3 4 5]
 [6 7 8]]
Weighted average along the specified axis of the above flattened array:
[1.2 4.2 7.2]
```

17. Write a NumPy program to compute the histogram of nums against the bins (CO4)
Sample Output:

```
nums:  [1.5  0.7  1.   1.2  1.3  2.75]
bins:  [0 1 2 3]

Result: (array([1, 4, 1]), array([0, 1, 2, 3]))
```



18. Write a Pandas program to add, subtract, multiple and divide two Pandas Series. Sample Series: [2, 4, 6, 8, 10], [1, 3, 5, 7, 9] (CO4)
Sample Output:

```
Add two Series:
0     3
1     7
2    11
3    15
4    19
dtype: int64
Subtract two Series:
0    1
1    1
2    1
3    1
4    1
dtype: int64
Multiply two Series:
0     2
1    12
2    30
3    56
4    90
dtype: int64
Divide Series1 by Series2:
0    2.000000
1    1.333333
2    1.200000
3    1.142857
```

```
4    1.111111
dtype: float64
```

19. Write a Pandas program to join the two given dataframes along row and assign all data. (CO4)
    Sample Output:

```
Original DataFrames:
student_idname  marks
0        S1  Danniella Fenton    200
1        S2      Ryder Storey    210
2        S3      Bryce Jensen    190
3        S4         Ed Bernal    222
4        S5       Kwame Morin    199
------------------------------------
student_idname  marks
0        S4  Scarlette Fisher    201
1        S5  Carla Williamson    200
2        S6       Dante Morse    198
3        S7    Kaiser William    219
4        S8   Madeeha Preston    201



Join the said two dataframes along rows:
student_idname  marks
0        S1  Danniella Fenton    200
1        S2      Ryder Storey    210
2        S3      Bryce Jensen    190
3        S4         Ed Bernal    222
4        S5       Kwame Morin    199
0        S4  Scarlette Fisher    201
1        S5  Carla Williamson    200
2        S6       Dante Morse    198
3        S7    Kaiser William    219
4        S8   Madeeha Preston    201
```

20. Write a Pandas program to append rows to an existing DataFrame and display the combined data. (CO4)
    Sample Output:

```
Original DataFrames:
student_idname  marks
0        S1  Danniella Fenton    200
1        S2      Ryder Storey    210
2        S3      Bryce Jensen    190
3        S4         Ed Bernal    222
```

```
4        S5       Kwame Morin    199

New Row(s)
student_id                S6
name          Scarlette Fisher
marks                    205
dtype: object

Combined Data:
student_idname  marks
0        S1  Danniella Fenton    200
1        S2      Ryder Storey    210
2        S3      Bryce Jensen    190
3        S4         Ed Bernal    222
4        S5       Kwame Morin    199
5        S6  Scarlette Fisher    205
```

21. Write a Pandas program to identify the column(s) of a given DataFrame which have at least one missing value. (CO4)
    Sample Output:

```
Original Orders DataFrame:
ord_nopurch_amtord_datecustomer_idsalesman_id
0   70001.0     150.50  2012-10-05        3002       5002.0
1       NaN270.65  2012-09-10        3001       5003.0
2   70002.0      65.26         NaN        3001       5001.0
3   70004.0     110.50  2012-08-17        3003          NaN
4       NaN948.50  2012-09-10        3002       5002.0
5   70005.0    2400.60  2012-07-27        3001       5001.0
6       NaN5760.00  2012-09-10        3001       5001.0
7   70010.0    1983.43  2012-10-10        3004          NaN
8   70003.0    2480.40  2012-10-10        3003       5003.0
9   70012.0     250.45  2012-06-27        3002       5002.0
10      NaN75.29  2012-08-17        3001       5003.0
11  70013.0    3045.60  2012-04-25        3001          NaN

Identify the columns which have at least one missing value:
ord_no          True
purch_amt      False
ord_date        True
customer_id    False
salesman_id     True
dtype: bool
```

22. Write a Pandas program to count the number of missing values in each column of a given DataFrame (CO4)
    Sample Output:

```
Original Orders DataFrame:
ord_nopurch_amtord_datecustomer_idsalesman_id
0   70001.0     150.50  2012-10-05        3002       5002.0
1       NaN270.65  2012-09-10        3001       5003.0
2   70002.0      65.26         NaN        3001       5001.0
3   70004.0     110.50  2012-08-17        3003          NaN
4       NaN948.50  2012-09-10        3002       5002.0
5   70005.0    2400.60  2012-07-27        3001       5001.0
6       NaN5760.00  2012-09-10        3001       5001.0
7   70010.0    1983.43  2012-10-10        3004          NaN
8   70003.0    2480.40  2012-10-10        3003       5003.0
9   70012.0     250.45  2012-06-27        3002       5002.0
10      NaN75.29  2012-08-17        3001       5003.0
11  70013.0    3045.60  2012-04-25        3001          NaN

Number of missing values of the said dataframe:
ord_no         4
purch_amt      0
ord_date       1
customer_id    0
salesman_id    3
dtype: int64
```

23. Write a Pandas program to calculate the total number of missing values in a DataFrame. (CO4)

    Sample Output:

```
Original Orders DataFrame:
ord_nopurch_amtord_datecustomer_id
0       NaNNaNNaNNaN
1       NaN270.65  2012-09-10       3001.0
2   70002.0      65.26         NaN       3001.0
3       NaNNaNNaNNaN
4       NaN948.50  2012-09-10       3002.0
5   70005.0    2400.60  2012-07-27       3001.0
6       NaN5760.00  2012-09-10       3001.0
7   70010.0    1983.43  2012-10-10       3004.0
8   70003.0    2480.40  2012-10-10       3003.0
9   70012.0     250.45  2012-06-27       3002.0
10      NaN75.29  2012-08-17       3001.0
11      NaNNaNNaNNaN

Total number of missing values of the said DataFrame:
17
```

24. Write a Pandas program to find and replace the missing values in a given DataFrame which do not have any valuable information (CO4)

    Sample Output:

```
Original Orders DataFrame:
ord_nopurch_amtord_datecustomer_idsalesman_id
0   70001     150.5          ?       3002       5002
1     NaN270.65  2012-09-10       3001       5003
2   70002     65.26        NaN       3001          ?
3   70004     110.5  2012-08-17       3003       5001
4     NaN948.5  2012-09-10       3002        NaN
5   70005    2400.6  2012-07-27       3001       5002
6     --      5760  2012-09-10       3001       5001
7   70010         ?  2012-10-10       3004          ?
8   70003     12.43  2012-10-10         --       5003
9   70012    2480.4  2012-06-27       3002       5002
10    NaN250.45  2012-08-17       3001       5003
11  70013    3045.6  2012-04-25       3001         --

Replace the missing values with NaN:
ord_nopurch_amtord_datecustomer_idsalesman_id
0   70001.0     150.50        NaN      3002.0      5002.0
1       NaN270.65  2012-09-10      3001.0      5003.0
2   70002.0      65.26        NaN      3001.0         NaN
3   70004.0     110.50  2012-08-17      3003.0      5001.0
4       NaN948.50  2012-09-10      3002.0         NaN
5   70005.0    2400.60  2012-07-27      3001.0      5002.0
6       NaN5760.00  2012-09-10      3001.0      5001.0
7   70010.0        NaN  2012-10-10      3004.0         NaN
8   70003.0      12.43  2012-10-10         NaN      5003.0
9   70012.0    2480.40  2012-06-27      3002.0      5002.0
10      NaN250.45  2012-08-17      3001.0      5003.0
11  70013.0    3045.60  2012-04-25      3001.0         NaN
```

25. Write a Pandas program to replace the missing values with the most frequent values present in each column of a given DataFrame. (CO4)
Sample Output:

```
ord_nopurch_amtsale_amtord_datecustomer_idsalesman_id
0   70001.0  150.50     10.50  2012-10-05        3002       5002.0
1    NaNNaN20.65  2012-09-10        3001       5003.0
2   70002.0  65.26      NaNNaN        3001       5001.0
3  70004.0  110.50     11.50  2012-08-17        3003         NaN
4    NaN     948.50     98.50  2012-09-10        3002       5002.0
5  70005.0     NaNNaN  2012-07-27        3001       5001.0
6    NaN    5760.00     57.00  2012-09-10        3001       5001.0
7   70010.0 1983.43     19.43  2012-10-10        3004         NaN
8   70003.0     NaNNaN  2012-10-10        3003       5003.0
9   70012.0  250.45     25.45  2012-06-27        3002       5002.0
10   NaN      75.29     75.29  2012-08-17        3001       5003.0
11  70013.0 3045.60     35.60  2012-04-25        3001         NaN

ord_nopurch_amtsale_amtord_datecustomer_idsalesman_id
```

```
0   70001.0  150.50     10.50  2012-10-05        3002       5002.0
1   70001.0  NaN20.65  2012-09-10        3001       5003.0
2  70002.0  65.26  10.50  2012-10-10     3001       5001.0
3 70004.0110.5011.502012-08-17   3003       5001.0
4  70001.0  948.50     98.50  2012-09-10        3002       5002.0
5  70005.0 65.26     10.50  2012-07-27        3001       5001.0
6 70001.0 5760.00    57.00  2012-09-10        3001       5001.0
7 70010.01983.4319.43 2012-10-10      3004       5001.0
8 70003.0 65.26     10.50  2012-10-10        3003       5003.0
970012.0  250.45     25.45  2012-06-27        3002       5002.0
10  70001.075.29    75.29 2012-08-17        3001       5003.0
11 70013.0 3045.60  35.60  2012-04-25      3001       5001.0
```

26. Write a program to demonstrate Regression analysis with residual plots on a given data set. (CO4)
27. Write a program to demonstrate the working of the decision tree-based ID3 algorithm. Use anappropriate data set for building the decision tree and apply this knowledge to classify a new sample. (CO4)
28. Write a program to implement the Naïve Bayesian classifier for a sample training data set stored as a .CSV file. Compute the accuracy of the classifier, considering few test data sets. (CO4)
29. Write a program to implement k-Nearest Neighbor algorithm to classify the iris data set. Print both correct and wrong predictions using Java/Python ML library classes. (CO4)
30. Write a program to implement k-Means clustering algorithm to cluster the set of data stored in .CSV file. Compare the results of various “k” values for the quality of clustering. (CO4)

Text Books:

1. Allen B. Downe, Think Python, Second Edition, Green Tea Press, Needham, Massachus etts, 2014.

Reference Books:

1. Mark Lutz, Python Pocket Reference, Fifth edition, O'Reilly Media, Inc, 2014.

COMPUTER ORGANIZATION AND MICROPROCESSOR INTERFACING LAB	
CSM 228	**Credits : 1.5**
Instruction : 3 Periods /Week	**Sessional Marks : 50**
End Exam : 3 Hours	**End Exam Marks : 50**

Prerequisites:

- Basic knowledge of Digital logic design

Course Objectives:

- Able to design the simple logic circuits and test/verify the functionality of the logic circuits.
- Developing of assembly language programs and providing the basics of the processors.
- To provide solid foundation on interfacing the external devices to the processor according to the user requirements to create novel products and solutions for the real time problems
- To assist the students with sufficient knowledge on the interrupts and working with interrupt driven I/O for communication with external devices.

Course Outcomes:

By the end of the course, the student will be able to:	
1	Identify the basic principles and apply to arithmetic for ALU implementation.
2	Possessed a better command over the instruction of set of 8085 and 8086 microprocessor for programmatically deployment.
3	Demonstrate the interfacing of 8085 microprocessor with external I/O devices through 8255 PPI.
4	Students will possess the knowledge to design and develop a working prototype with various simulators and emulators that they have used throughout the lab sessions.

Mapping of Course Outcomes with Program Outcomes:

Mapping		**PO**												**PSO**	
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
CO	**1**	**3**	**1**	**3**	**-**	**-**	**-**	**-**	**-**	**2**	**1**	**-**	**-**	**-**	**-**
	2	**2**	**1**	**2**	**-**	**-**	**-**	**-**	**-**	**2**	**1**	**-**	**-**	**1**	**-**
	3	**1**	**1**	**2**	**-**	**-**	**-**	**-**	**-**	**2**	**1**	**-**	**-**	**1**	**-**
	4	**2**	**1**	**2**	**-**	**-**	**-**	**-**	**-**	**2**	**1**	**-**	**-**	**1**	**-**

S.NO	LIST OF EXPERIMENTS	No. of Weeks	CO’s
1	<u>**COMPUTER ORGANIZATION**</u> Verification of truth table of various logic gates. Study of Arithmetic Logic Unit (ALU)	4	1
2	**8085 ASSEMBLY LANGUAGE PROGRAMMING** According to theory course using the following: Keyboard Monitor of 8085µP Trainer.	4	2 &4
3	<u>**INTERFACING WITH 8085 TRAINER**</u> 8255 Study Card Scenarios (I/O and BSR Mode Operations) 8255 Modes Using Hardware Interrupts Traffic Light Controller Stepper Motor Controller Keyboard/Display Interface	4	3
4	**8086 ASSEMBLY LANGUAGE PROGRAMMING** According to theory course using the following: PC Assembler using TASM or MASM, TD or SYMDEB or CVD (Code View debugger)	2	2&4

TEXT BOOKS:

1. Ramesh S. Gaonkar, ―Microprocessor Architecture, Programming, and Applications with the 8085‖ Penram International, 6th Edition.
2. John E.Uffenbeck, ―The 80x86 Family, Design, Programming and Interfacing 3rdEdition, Pearson Education Inc.‖, 2002.

REFERENCE BOOKS:

1. BARRY B. BREY, ―The Intel Microprocessors 8086/8088, 80186/80188,80286,80386 and 80486, Pentium, Pentium Pro Processor, Pentium II, Pentium III, Pentium 4, Architecture, Programming and Interfacing‖, Pearson Education Inc., 2003,6thEdition.
2. Walter A. Tribel and Avtar Singh,‖ The 8088 and 8086 Microprocessors, Programming, interfacing, Software, Hardware, and Applications‖, Pearson Education Inc., 2003, 4thEdition.
3. Douglass V. Hall, ―Microprocessors and Interfacing, Programming and Hardware‖ , TMH Edition, 1999, 2ndEdition

ONLINE WEB RESOURCES:

1. https://swayam.gov.in/nd1_noc20_ee11/preview
2. https://www.coursera.org/lecture/cs-algorithms-theory-machines/digital-circuits-91A4N

OPERATING SYSTEM LAB	
CSM 229	**Credits : 1.5**
Instruction **:** 3 Periods /week	Sessional Marks : 50
End Exam : 3 Hours	End Exam Marks : 50

Prerequisites: Basic programming language

Course objective:

- To learn and execute the basic shell script, UNIX commands and system calls.
- To understand and implement the process, memory and file management.
- To solve the problems related to process synchronization.

Course Outcomes:

At the end of the course the student will be able to

CO1	Execute the Unix Shell programming on the given system configuration.
CO2	Acquire skill in the various services provided by the system calls
CO3	Simulate the process scheduling, process synchronization, deadlock avoidance and detection algorithms.
CO4	Simulate memory management techniques and file handling.

Mapping of Course Outcomes with Program Outcomes:

CO	PO												PSO	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
1	1	2	2	-	-	-	-	-	-	-	-	-	-	-
2	1	2	2	-	-	-	-	-	-	-	-	-	-	-
3	2	2	3	1	-	-	-	-	-	-	-	-	1	-
4	2	2	3	1	-	-	-	-	-	-	-	-	1	-

LIST OF SAMPLE PROGRAMS

Write a Program for the following

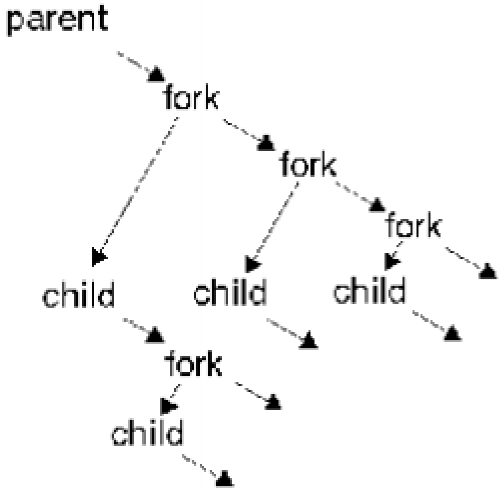
1. Implement basic shell commands. (CO1)

2. Shell programming: Simple logic programs. (CO1)

i) Write a menu driven script using the select statement to print calories for food items such as pizza, burger, Salad, Pasta etc.

ii) Write a shell script that, given a filename as the argument will count vowels, blank spaces, characters, number of line and symbols.

3. i) Analyze the below situation and develop a program for creating processes as required. Print the PID's of each process in a convenient way to understand. (CO2)



ii) Write a program to create two processes P1 and P2. P1 takes a string and passes it to P2. P2 concatenates the received string with another string without using string function and sends it back to P1 for printing. (CO2)

4. CPU Scheduling Algorithms (CO3)

i) A washing machine which requires the process to be executed sequentially. Consider the processes P1, P2, P3, P4 whose arrival times are 1, 5, 9, 10 and burst times are 4, 3, 5, 2 respectively. Implement an appropriate algorithm. Find the CPU idle time, so that the water can be supplied during that period of time.

ii) Implement shortest job first for the following data:

Consider the following set of processes, CPU burst time, Arrival time. Calculate the average waiting time, average response time and average turnaround time.

Process	Burst Time	Arrival Time
P1	3	0
P2	6	2

P3	4	4
P4	5	6
P5	2	8

iii) Implement Round Robin for the following data

Consider the following set of processes and length of the CPU burst time given in milliseconds.

Process	**Burst Time**
P1	10
P2	1
P3	2
P4	1
P5	5

The processes are assumed to have arrived in the order P1, P2, P3, P4, P5 all at time 0 and time quantum in RR=1.Calculate the average waiting time, response time and turnaround time.

5. Develop a program to provide synchronization among the 5 philosophers in Dining Philosophers problem using semaphore. (CO3)

6. Develop a program to provide synchronization among the producer and consumer processes in producer –consumer problem using a monitor. (CO3)

7. Consider the following data: (CO3)

Process	**Allocation**	**Max**	**Available**
	A B C D	**A B C D**	**A B C D**
P1	0 0 1 2	00 1 2	2 1 0 0
P2	2 0 0 0	2 7 5 0	
P3	0 0 3 4	6 6 5 6	
P4	2 3 5 4	4 3 5 6	
P5	0 3 3 2	0 6 5 2	

i) Calculate the need matrix
ii) Is this system currently in a safe or unsafe state?
iii) Is the system currently deadlock or not.
iv) Which process, if any, or may become deadlocked?

8. Consider the following scenario: A process has been allocated 3 page frames. Assume that none of the pages of the process are available in the memory initially. The process makes the following sequence of page references (reference string): 1, 2, 1, 3, 7, 4, 5, 6, 3, 1, 2, 4, 6, 3, 1. Find a page replacement policy which gives the least number of page faults. (CO4)

9. Simulate the Virtual Memory concept. (CO4)

10. Implement the first fit and best fit algorithm in memory management. (CO4)

11. Simulate the Contiguous file allocation method. (CO4)

12. Implement a bit map for the following scenario. (CO4)
For a memory of size 32 blocks ,the allocated blocks are 2,3,4,5,8,9,10,11,12 and display the bitmap pattern.

REFERENCES:

1. Sumitabha Das, “Unix Concepts and Applications”, 4th Edition. TMH, 2006.
2. Abraham Silberschatz, Peter Baer Galvin, Greg Gagne, “Operating System Concepts”, 9th Edition, John Wiley & Sons, 2015.
3. William Stalling, “Operating Systems: Internals and Design Principles”, 9th edition, PHI, 2018.

WEB REFERENCES:

1. https://nptel.ac.in/content/storage2/courses/106108101/pdf/PPTs/Mod_13.pdf
2. https://nptel.ac.in/courses/117106113/