UNIVERSITY SCHOOL

OF

INFORMATION AND COMMUNICATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAMME STRUCTURE

MASTER OF COMPUTER APPLICATIONS (MCA)

WITH SPECIALIZATION IN ARTIFICIAL INTELLIGENCE 2 YEAR PROGRAMME

2021-23

GAUTAM BUDDHA UNIVERSITY

GAUTAM BUDH NAGAR, GREATER NOIDA, UP, INDIA

Passed B.Sc/ B.Com/ B.A with Mathematics at 10+2 level or at Graduation level with having at least 50% marks (45% for SC/ST candidates of Haryana only) in aggregate, along with the students admitted with this eligibility will have to simultaneously undertake additional *bridge course as prescribed by the University during the first semester.

Note: * It is compulsory for each student to pass out bridge course (three additional theory papers and one practical as prescribed in scheme of examination of bridge course) as per University norms during the 1st year of MAI-2 year course and the degree will be awarded after the completion of bridge course. However, these papers under bridge course will be taught only in the 1st semester of the course.

OR

Passed BCA/B.Sc.(Hons.)Computer Science/ B.E. or B.Tech.(CSE/IT)/ B.Voc.(Software Development/IT) or an equivalent degree with having at least 50% marks (45% for SC/ST candidates of Haryana only) in aggregate.

BRIDGE COURSE

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	MCB001	Computer Fundamental and Programming	3	1	0	4	BC1
2	MCB003	Introduction to Internet Technology	3	0	0	3	BC2
3	MCB005	Fundamental of Operating Systems	3	0	0	3	BC3
4	MCB081	Operating System Lab	0	0	3	2	BC-L1
5	MCB083	Internet Technology Lab	0	0	3	2	BC-L2
		Total Hours and Credits	**9**	**1**	**6**	**14**	

SEMESTER I

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	MAI101	Computer Fundamental and C Programming	3	0	0	3	CC1 / FC
2	MAI103	Software Engineering	3	0	0	3	CC2
3	MAI105	Artificial Intelligence	3	0	0	3	CC3 / FC
4	MAI107	Discrete Mathematics	3	0	0	3	CC4
5	MAI109	Theory of Automata	3	0	0	3	CC5
6	MAI111	Operating System	3	0	0	3	CC6
7	MAI113	Data Base Management System	3	0	0	3	CC7
8	ES401	Fundamental of Environmental Science	3	0	0	3	OE1 / AECC
9	MAI181	C Programming Lab	0	0	3	2	CC-L1 / SEC
10	MAI183	Data Base Management System Lab	0	0	3	2	CC-L2 / SEC
11	GP	General Proficiency	Non Credit				
		Total Hours and Credits	**24**	**0**	**6**	**28**	

SEMESTER II

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	MAI102	JAVA Programming	3	0	0	3	CC8 / SEC
2	MAI104	Data Structure and Algorithms	3	0	0	3	CC9 / SEC
3	MAI106	Compiler Design	3	0	0	3	CC10
4	MAI108	Expert System Design	3	0	0	3	CC11
5	MAI110	Natural Language Processing	3	0	0	3	CC12
6		Elective-1	3	0	0	3	E1 / DSE
7		Elective-2	3	0	0	3	E2 / DSE
8	EN532	Fundamental of Language Science	3	0	0	3	OE2 / AECC
9	MAI182	Data Structure and Algorithms Lab	0	0	3	2	CC-L3 / SEC
10	MAI184	JAVA Programming Lab	0	0	3	2	CC-L4 / SEC
11	GP	General Proficiency	Non Credit				
		Total Hours and Credits	**24**	**0**	**6**	**28**	

SEMESTER III

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	MAI201	Analysis and Design of Algorithm	3	0	0	3	CC13
2	MAI203	Machine Learning	2	0	0	2	CC14
3	MAI205	Soft Computing Techniques	3	0	0	3	CC15
4	MAI207	Python Programming	2	0	0	2	CC16 / SEC
5		Elective-3 (MOOC)	3	0	0	3	E3 / DSE
6		Elective-4	3	0	0	3	E4 / DSE
7	MA209	Numerical Algorithms for Machine Learning	3	0	0	3	GE1
8	MAI291	Minor Project	0	0	10	5	MP1 / E
9	MAI281	Analysis and Design of Algorithm Lab	0	0	3	2	CC-L5 / SEC
10	MAI283	Machine Learning using Python Lab	0	0	3	2	CC-L6 / SEC
11	GP	General Proficiency	Non Credit				
		Total Hours and Credits	**19**	**0**	**16**	**28**	

SEMESTER IV

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	MAI290	Seminar	0	0	3	2	S / E
2	MAI292	Major Project	0	0	20	10	MP2 / E
3	MAI294	Internship	0	0	32	16	I / E
4	GP	General Proficiency	Non Credit				
		Total Hours and Credits	**0**	**0**	**55**	**28**	

GRAND TOTAL OF CREDITS = 112

Swayam/NPTEL courses will also be offered as **Discipline Specific Elective** during the II and III semester and Swayam/NPTEL courses list will be provided by the deptt. OR if USICT will offer the course as an Discipline Specific Elective, students will complete it through regular class. It will be evaluated as per University Examination Rules.

In the **Seminar**, student need to study and present individually, on latest research paper of their specialized area and It will be evaluated as per University Examination Rules.

The **Internship** in Industry will be done by candidate individually during the 4th semester and it will be for a minimum of 4 (-6) months. It will be evaluated as per University Examination Rules.

Minor and Major Project will be in a group and It will be evaluated as per University Examination Rules.

USICT will provide a **mentor/supervisor** for seminar, internship, minor and major projects.

ELECTIVES FROM USICT

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	MAI112	Pattern Matching	3	0	0	3	E1
2	MAI114	Robot Kinematics	3	0	0	3	E1
3	MAI116	Knowledge Engineering	3	0	0	3	E1
4	MAI118	Computer Vision	3	0	0	3	E1
5	MAI120	Fuzzy Logic	3	0	0	3	E2
6	MAI122	Evolutionary Computing	3	0	0	3	E2
7	MAI124	Speech Processing	3	0	0	3	E2
8	MAI126	Convex Optimization	3	0	0	3	E2
9	MAI209	Computational Intelligence	3	0	0	3	E3
10	MAI211	Parallel Distributed Systems	3	0	0	3	E3
11	MAI213	AI Enabled Cyber Security	3	0	0	3	E3
12	MAI215	Deep Learning and Reinforcement Learning	3	0	0	3	E3
13	MAI217	Pattern Matching	3	0	0	3	E4
14	MAI219	Neural Network	3	0	0	3	E4
15	MAI221	Brain and Neuroscience	3	0	0	3	E4
16	MAI223	Speech Analysis and Systems	3	0	0	3	E4
17	MA033	Numerical Algorithms for Machine Learning	3	0	0	3	GE1
18	ES401	Fundamentals of Environmental Science	3	0	0	3	OE1
19	ES507	Energy and Environment	3	0	0	3	OE1
20	EN532	Fundamental of Language Science	3	0	0	3	OE2

MAI MCA

Master of Computer Applications and **A**rtificial **I**ntelligence for Course Code

Master of Computer Applications

SEC Skill Enhancement Course

CC	Core Course from USICT for Course Type
FC	Foundation Course
CC-L	Core Course Lab from USICT
GE	General Elective from related discipline of other Deptt./School
AECC	Ability Enhancement Compulsary Course
OE	Open Elective from other discipline of other Deptt./School
DSE	Discipline Specific Course
BC	Bridge Course
BC-L	Bridge Course Lab
MP	Minor / Major Project
S	Seminar
I	Industrial Training / Internship

SEMESTER I

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	MAI101	Computer Fundamental and C Programming	3	0	0	3	CC1 / FC
2	MAI103	Software Engineering	3	0	0	3	CC2
3	MAI105	Artificial Intelligence	3	0	0	3	CC3 / FC
4	MAI107	Discrete Mathematics	3	0	0	3	CC4
5	MAI109	Theory of Automata	3	0	0	3	CC5
6	MAI111	Operating System	3	0	0	3	CC6
7	MAI113	Data Base Management System	3	0	0	3	CC7
8	ES401	Fundamental of Environmental Science	3	0	0	3	OE1 / AECC
9	MAI181	C Programming Lab	0	0	3	2	CC-L1 / SEC
10	MAI183	Data Base Management System Lab	0	0	3	2	CC-L2 / SEC
11	GP	General Proficiency	Non Credit				
		Total Hours and Credits	**24**	**0**	**6**	**28**	

COMPUTER FUNDAMENTAL AND C PROGRAMMING			
Course Code:	**MAI101**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs./Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

- To develop the fundamental understanding of computers, its components and programming environment.
- To create programming logics and learn C language programming concepts and techniques.
- To design and develop algorithms and programs with different data declarations, initialization and related operations.
- To develop the ability to define and manage functions, array, structures, pointers etc. based on program objective.
- To understand and develop C programs to handle computer files, their usage and perform various operations on files.

COURSE OUTCOMES

At the end of the course the students should be able to:

- Understand the Computer fundamentals.
- Use of various problem-solving techniques.
- Understand the C programming fundamentals.
- Understand C by using arrays, functions, structures and union.
- Develop the Programs in C using its advance features.

UNIT I INTRODUCTION TO COMPUTER AND PROGRAMMING CONCEPTS

Definition, characteristic, generation of computers, basic components of a computer system, memory, input, output and storage units, high level language and low level language, Soft-ware: system software, application software, hardware, firmware, Operating System, compiler, interpreter and assembler, linker, loader, debugger, IDE. Introduction to algorithm and flow chart; representation of algorithm using flow chart symbol, pseudo code, basic algorithm de-sign, characteristics of good algorithm, development of algorithm.

UNIT II INTRODUCTION TO C PROGRAMMING LANGUAGE

Introduction to C programming language , Declaring variables, preprocessor statements, arithmetic operators, programming style, keyboard input , relational operators, introduction, feature of C language, concepts, uses, basic program structure, simple data types, variables, constants, operators, comments, control flow statement :if, while, for, do-while, switch.

UNIT III DATA TYPES AND STRUCTURES

Bitwise operators, Pre-defined and User defined data types, arrays, declaration and operations on arrays, searching and sorting on arrays, types of sorting, 2D arrays, passing 2D arrays to functions, structure, member accessing, structure and union, array of structures, functions, declaration and use of functions, parameter passing, and recursion.

UNIT IV FUNDAMENTALS OF POINTERS

Introduction to pointers, pointer notations in C, Declaration and usages of pointers, operations that can be performed on computers, use of pointers in programming exercises, parameter passing in pointers, call by value, call by references, array and characters using pointers, dynamic memory allocation

UNIT V FILE HANDLING IN C AND ENUM

Introduction to file handling, file operations in C, defining and opening in file, reading a file, closing a file, input output operations on file, counting: characters, tabs , spaces, file opening modes, error handling in input/output operations, sEnumerated data types, use of Enum, declaration of Enum.

Text Books:

[1] C Programming, Herbert Shield
[2] C Programming Language 2nd Edition by Brian, W Kernighan Pearson Education.

Reference Books:

[3] Programming in ANSI C by E. Balagurusamy, Tata Mgraw Hill
[4] C Puzzle Book: Puzzles For The C. Programming Language by Alan R Feuer Prentice Hall-Gale
[5] Expert C Programming: Deep C Secrets (s) by Peter Van Der Linden Dorling Kindersley India.
[6] Introduction To UNIX System by Morgan Rachel Tata Mcgraw Hill Education.
[7] C: A Reference Manual (5th Edition) by Samuel P. Harbison&Samuel P. Harbison.
[8] Programming Using the C Language by Hutchison,R.C, Mcgraw Hill Book Company, New York.
[9] Fundamentals of computers and programming with C, A.K. SHARMA

SOFTWARE ENGINEERING			
Course Code:	**MAI103**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs./Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

- Knowledge of basic SW engineering methods and practices and application.
- A general understanding of software process models.
- Understanding of software requirements and the SRS documents.
- Understanding of software design process.
- Understanding of software coding, testing and maintenance.

COURSE OUTCOMES

At the end of the course the students should be able to:

- Basic knowledge and understanding of the analysis and design of complex systems.
- Ability to apply software engineering principles and techniques.
- Ability to develop, maintain and evaluate large-scale software systems.
- To produce efficient, reliable, robust and cost-effective software solutions.
- Ability to perform independent research and analysis.

UNIT I SOFTWARE ENGINEERING

Introduction to software engineering: definitions, role of software engineering, planning a software project, defining the problem, developing a solution strategy, planning the development process, software engineering process paradigms, principles of software engineering, software engineering activities.

UNIT II SOFTWARE LIFE CYCLE MODELS

Software Development Life Cycle (SDLC), SDLC models, waterfall model and its variations, prototype model, iterative enhancement model, spiral model, RAD model, comparison of these models, software development teams, software development environments, validation and traceability, maintenance, prototyping requirements, Software project management.

UNIT III REQUIREMENT ANALYSIS AND DESIGN

Software Requirement Specification (SRS): Introduction, need of SRS, significance, characteristics of SRS, Structure of SRS, IEEE standards for SRS design, functional and non-functional requirements, Requirement gathering and analysis, requirement engineering and management.

UNIT IV SOFTWARE DESIGN PROCESS

Software Design: Introduction, design process activities: architectural design, Abstract specification, Interface design, component design, data structure design, algorithm design modular approach, top-down design, bottom-up design, design methods: data-flow model: data flow diagram, entity-relation-attribute model: E-R diagram, structural model: structure charts, context diagrams, object models: use case modeling, use case diagrams, sequence diagrams, cohesion and coupling.

UNIT V SOFTWARE CODING, TESTING AND MAINTENANCE

Coding, Testing Methods: unit testing, integration testing, system testing, acceptance testing, testing techniques: white box testing, black box testing, thread testing, regression testing, alpha testing, beta testing, static testing, dynamic testing, Evolution of software products, economics of maintenance, category of software maintenance, Role of product development life cycle, deployment model, adaptive maintenance, corrective maintenance, perfective maintenance, enhancement request, proactive defect prevention, problem reporting, problem resolution, software maintenance from customers' perspective, maintenance standard: IEEE-1219, ISO-12207.

Text Books:

1. Pankaj Jalote, An Integrated Approach to Software Engineering, Narosa Publishing House, New Delhi 1997.
2. Ian Sommerville, Software Engineering, Pearson Education, 2009.
3. Pressman Roger S., Software Engineering: Practitioner's Approach, McGraw-Hill Inc., 2004.
4. Software Engineering: Software Reliability, Testing and Quality Assurance, Nasib S. Gill, Khanna Book Publishing Co (P) Ltd., New Delhi, 2002.

ARTIFICIAL INTELLIGENCE			
Course Code:	**MAI105**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs./Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

- Gain a historical perspective of AI and its foundations.
- Become familiar with basic principles of AI toward problem-solving, inference, perception, knowledge representation, and learning.
- Investigate applications of AI techniques in intelligent agents, expert systems, and machine learning models.
- Experience AI development tools such as an 'AI language', expert system shell, and/or data mining tool.
- Explore the current scope, potential, limitations, and implications of intelligent systems.

COURSE OUTCOMES

At the end of the course the students should be able to:

- Demonstrate knowledge of the building blocks of AI as presented in terms of intelligent agents
- Analyze and formalize the problem as a state space, graph, design heuristics, and select different search or game-based techniques to solve them.
- Develop intelligent algorithms for constraint satisfaction problems and also design intelligent .systems for Game Playing
- Attain the capability to represent various real-life problem domains using logic-based techniques and use this to perform inference or planning.
- Solve reasoning problems with Expert Systems.

UNIT I INTRODUCTION TO ARTIFICIAL INTELLIGENCE

Basic concept of artificial intelligence (AI), history of AI, AI and consciousness, weak and strong AI, physical symbol system hypothesis, comparison of computer and human skills, practical systems based on AI, development of logic, components of AI, Turing Test in AI, Advantages and Disadvantages of AI, Intelligence, Intelligent System, Role of IS, Comparison of various IS, Mind-Body Problem in AI, Chinese Room Experiment in AI, Parallel and Distributed AI.

UNIT II PROBLEM SOLVING THROUGH AI

Defining the problem as state-space search, analyzing the problem, representing the problems from AI viewpoint, production system, developing production rules, characteristics of the production system, algorithm for problem-solving using AI technique.

UNIT III SEARCH TECHNIQUES

Use of search in AI problem solution, blind search techniques, heuristic search techniques, concept of heuristic knowledge, designing of the heuristic function, types of heuristic search techniques: generate and test, best first search, problem reduction using AND-OR graph, local search technique, branch and bound search, memory bounded search technique, local beam search, properties

of heuristic search techniques, overestimation and underestimation of heuristic function hill climbing search, simulated annealing search, constraint satisfaction means ends analysis, Tic-Tac Toe Problem, Water Jug problem, Chess Problem, Tower of Hanoi problem, Travelling Salesman problem, Monkey and Banana Problem, Magic Square.

UNIT IV INTRODUCTION TO LOGIC

Introduction, proposition calculus, syntax o propositional calculus, semantics of propositional calculus, well-formed formula, properties of statements, inferencing of propositional logic, predicate logic, syntax of predicate logic, semantics of predicate logic, concept of resolution, resolution algorithm, skolemization, types of resolution unit resolution, binary resolution.

UNIT V AI TECHNIQUES AND APPLICATIONS

Introduction to Machine Learning, Introduction to Deep Learning, Introduction to Expert system: Introduction phases in building expert systems, Expert system versus traditional systems, rule-based expert systems, blackboard systems, application of expert systems, list of shells and tools, Introduction to Natural Language Processing, AI in future, AI in social Media, AI in Entertainment and education, AI in drones, AI in Automated Computer support, AI in personalized shopping experience, AI in Finance, AI in smart Cars, AI in travel and navigation, AI in smart home devices, AI in security and surveillance, Ai in education, AI in health care, AI in E commerce.

Text Books:

1. Artificial Intelligence, Elanie Reich: Tata mcgraw Hill publishing house, 2008.
2. Artificial Intelligence, Peterson, TataMcGraw Hill, 2008.
3. Artificial Intelligence, Russel and Norvig, Pearson Printice Hall Publication, 2006.
4. Artificial Intelligence, Winston, PHI publication, 2006

DISCRETE MATHEMATICS			
Course Code:	**MAI107**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs./Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

- Simplify and evaluate basic logic statements including compound statements, implications, inverses, converses, and contrapositives using truth tables and the properties of logic.
- Express a logic sentence in terms of predicates, quantifiers, and logical connectives.
- Apply the operations of sets and use Venn diagrams to solve applied problems; solve problems using the principle of inclusion-exclusion.
- Determine the domain and range of a discrete or non-discrete function, graph functions, identify one-to-one functions, perform the composition of functions, find and/or graph the inverse of a function, and apply the properties of functions to application problems.
- Apply rules of inference, tests for validity, proof by contradiction, proof by cases, and mathematical induction and write proofs using symbolic logic and Boolean Algebra.

COURSE OUTCOMES

At the end of the course the students should be able to:

- To express a logic sentence in terms of predicates, quantifiers, and logical connectives.
- Apply the rules of inference, proof by contradiction, and mathematical induction.
- Students will be able to evaluate Boolean functions and simplify expressions using the properties of Boolean algebra.
- Students will be able to learn about predicates, quantifiers, and logical connectives.
- Student will be able to use tree and graph algorithms to solve problems.

UNIT 1 MATHEMATICAL LOGIC

Statements and notations, connectives, well formed formulas, truth tables, tautology, equivalence implication, normal forms, predicates: predicative logic, free & bound variables, rules of inference, consistency, proof of contradiction, automatic theorem proving.

UNIT II SET THEORY

Set Theory: Introduction, Combination of sets, Multi sets, ordered pairs, Set Identities, Properties of binary relations, equivalence, compatibility and partial ordering relations, Hasse diagram. functions: Operations on functions, inverse function Classification of functions, recursive functions, lattice and its properties, algebraic structures: algebraic systems examples and general properties, semi groups and monads, groups sub groups‟ homomorphism, isomorphism.

UNIT III ELEMENTARY COMBINATORICS

Basis of counting, combinations & permutations, with repetitions, constrained repetitions, binomial coefficients, binomial multinomial theorems, the principles of inclusion – exclusion, pigeon hole principles and it's application.

UNIT IV RECURRENCE RELATION

Generating functions, function of sequences calculating coefficient of generating function, recurrence relations, solving recurrence relation by substitution and generating funds, characteristics roots solution of in homogeneous recurrence relation.

UNIT V GRAPH THEORY

Representation of graph, Trees: Definition, Binary tree, Binary tree traversal, Binary search tree. DFS, BFS, spanning trees, planar graphs. graph theory and applications, basic concepts isomorphism and sub graphs, multi graphs and euler circuits, hamiltonian graphs, chromatic numbers.

Text Books:

1. Discrete and Combinational Mathematics- An Applied Introduction-5th Edition – Ralph. P.Grimaldi, Pearson Education
2. Discrete Mathematical Structures with applications to computer science Trembly J.P. & Manohar.P, TMH
3. Discrete Mathematics and its Applications, Kenneth H. Rosen, Fifth Edition.TMH.
4.Discrete Mathematical structures Theory and application-Malik & Sen
5.Discrete Mathematics for Computer science, Garry Haggard and others, Thomson.
6.Logic and Discrete Mathematics, Grass Man & Trembley, Person Education

THEORY OF AUTOMATA			
Course Code:	**MAI109**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs./Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

- Determine the various categories of automata (deterministic and nondeterministic finite state automata, and variants of Turing machines)
- Understand the various categories of languages and grammars in the Chomsky hierarchy
- Define the notions of computability and decidability
- Recognize to which class in the Chomsky hierarchy the language described (by a grammar or machine)
- Discover the problems reducible to/from well-known decidable/undecidable problems

COURSE OUTCOMES

- Acquire a fundamental understanding of the core concepts in automata theory and formal languages.
- An ability to design grammars and automata (recognizers) for different language classes.
- An ability to identify formal language classes and prove language membership properties.
- An ability to prove and disprove theorems establishing key properties of formal languages and automata.
- Acquire a fundamental understanding of core concepts relating to the theory of computation and computational models including decidability and intractability.

UNIT I INTRODUCTION

Introduction; alphabets, strings and languages; automata and grammars, deterministic finite automata (DFA)-formal definition, simplified notation: state transition graph, transition table, language of DFA, Nondeterministic finite Automata (NFA), NFA with epsilon transition, language of NFA, equivalence of NFA and DFA, minimization of finite automata, distinguishing one string from other, Myhill-Nerode Theorem.

UNIT II REGULAR EXPRESSIONS

Regular expression (RE), definition, operators of regular expression and their precedence, algebraic laws for regular expressions, Kleen‟s theorem, regular expression to FA, DFA to regular expression, arden theorem, non regular languages, pumping lemma for regular languages. application of pumping lemma, closure properties of regular languages, decision properties of regular languages, FA with output: moore and mealy machine, equivalence of moore and mealy machine, applications and limitation of FA.

UNIT III CFG

Context Free Grammar (CFG) and Context Free Languages (CFL): definition, examples, derivation, derivation trees, ambiguity in grammar, inherent ambiguity, ambiguous to unambiguous CFG, useless symbols, simplification of CFGs, normal forms for CFGs: CNF and GNF, closure properties of CFLs, decision properties of CFLs: emptiness, finiteness and membership, pumping lemma for CFLs.

UNIT IV PUSH DOWN AUTOMATA

Push Down Automata (PDA): description and definition, instantaneous description, language of PDA, acceptance by final state, acceptance by empty stack, deterministic PDA, equivalence of PDA and CFG, CFG to PDA and PDA to CFG, two stack PDA.

UNIT V TURING MACHINES (TM)

Basic model, definition and representation, instantaneous description, language acceptance by TM, variants of turing machine, TM as computer of integer functions, universal TM, church‟s thesis recursive and recursively enumerable languages, halting problem, introduction to undecidability, undecidable problems about TMs. Post Correspondence Problem (PCP), modified PCP, introduction to recursive function theory.

Text Books:

1. Hopcroft, Ullman, “Introduction to Automata Theory, Languages and Computation”, Pearson Education
2. K.L.P. Mishra and N.Chandrasekaran, “Theory of Computer Science : Automata, Languages and Computation”, PHI
3. Martin J. C., “Introduction to Languages and Theory of Computations”, TMH
4. Papadimitrou, C. and Lewis, C.L., “Elements of the Theory of Computation”, PHI

OPERATING SYSTEM			
Course Code:	**MAI111**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs./Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
● To introduce the different types of operating system.
● To introduce about the synchronization algorithms and semaphores.
● To introduce about conditional critical regions and continuous allocation.
● To introduce about the need for file system organization and disk scheduling.
● To understand the security and protection in operating system
● To know virtual memory concepts and security secondary management
COURSE OUTCOMES
At the end of the course the students should be able to:
● Understand the different types of Operating systems and scheduling algorithms.
● Understand the synchronization algorithms and semaphores.
● Critically evaluate the different memory allocation techniques.
● Appreciate the importance of file system organization, I/O management and disk scheduling.
● Appreciate the inter process communication and deadlock handling.

UNIT I

Types of operating systems, Different views of the operating system, Principles of Design and Implementation. The process and threads, System programmer's view of processes, Operating system's views of processes, Operating system services for process management, Process scheduling, Schedulers, Scheduling algorithms, Overview of Linux operating system.

UNIT II

Interprocess synchronization, Mutual exclusion algorithms, Hardware support, Semaphores, Concurrent programming using semaphores.

UNIT III

Conditional critical regions, Monitors, Interprocess communication, Messages, Pipes. Deadlocks: Characterization. Prevention, Avoidance. detection and recovery, Combined approach to deadlock handling.

UNIT IV

Contiguous allocation. Static and dynamic partitioned memory allocation, Segmentation, Non- contiguous allocation, Paging, Hardware support, Virtual Memory.

UNIT V

Need for files, File abstraction, File naming, File system organization, File system optimization, Reliability, Security and protection, I/O management and disk scheduling, Recent trends and developments.

Text Books:

1. Gary: Operating Systems- A modern Perspective, (2/e), Addison Wesley,2000.
2. M.Milenkovic: Operating systems, Concepts and Design, McGraw Hill,1992. Reference Books 1. C. Crowley: Operating Systems, Irwin,1997.
2. J.l. Peterson & A.S. Chatz: Operating System Concepts, Addison Wesley,1985.

3. W. Stallings: Operating Systems, (2/e), Prentice Hall,1995.
4. Mattuck,A., Introduction to Analysis,Prentice-Hall,1998. 5. Recent literature in Operating Systems.

DATA BASE MANAGEMENT SYSTEM			
Course Code:	**MAI113**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials (Hrs./Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

- Understand database concepts and structures and query language
- Understand the E R model and relational model.
- To design and build a simple database system and demonstrate competence with the fundamental tasks involved with modeling, designing, and implementing a DBMS.
- Understand Functional Dependency and Functional Decomposition.
- Apply various Normalization techniques.
- Execute various SQL queries. Understand query processing and techniques involved in query optimization.
- Understand the principles of storage structure and recovery management.

COURSE OUTCOMES

At the end of the course the students should be able to:

- Basic knowledge and understanding of the analysis and design of complex systems.
- Ability to apply software engineering principles and techniques.
- Ability to develop, maintain and evaluate large-scale software systems.
- To produce efficient, reliable, robust and cost-effective software solutions.
- Ability to perform independent research and analysis.

UNIT I DATA BASE SYSTEM

Data base system vs. file system, view of data, data abstraction, instances and schemas, data models, ER model, relational model, database languages, DDL, DML, database access for applications programs, data base users and administrator, transaction management, data base system structure, storage manager, query processor, history of data base systems, data base design and ER diagrams, beyond ER design entities, attributes and entity sets, relationships and relationship sets, additional features of ER model, concept design with the ER model, and conceptual design for large enterprises.

UNIT II RELATIONAL DATA BASE MODEL

Introduction to the relational model, integrity constraint over relations, enforcing integrity constraints, querying relational data, and logical data base design, destroying /altering tables and views. relational algebra and calculus: relational algebra, selection and

projection set operations, renaming, joins, division, relational calculus, tuple relational calculus, domain relational calculus, expressive power of algebra and calculus.

UNIT III SQL QUERY

Examples of basic SQL queries, nested queries, correlated nested queries set, comparison operators, aggregative operators, NULL values, comparison using null values, logical connectivity's, AND, OR and NOTR, impact on SQL constructs, outer joins, disallowing NULL values, complex integrity constraints in SQL triggers and active data bases.

UNIT IV NORMAL FORM

Problems caused by redundancy, decompositions, problem related to decomposition, reasoning about FDS, FIRST, SECOND, THIRD normal form, BCNF, forth normal form, fifth normal form, lossless join decomposition, dependency preserving decomposition, schema refinement in data base design, multi valued dependencies.

UNIT V TRANSACTION MANAGEMENT

ACID properties, transactions and schedules, concurrent execution of transaction, lock based concurrency control, performance locking, and transaction support in SQL, crash recovery, concurrency control, Serializability and recoverability, lock management, lock conversions, dealing with dead locks, specialized locking techniques, concurrency without locking, crash recovery: ARIES, log, other recovery related structures, the write, ahead log protocol, check pointing, recovering from a system crash, media recovery, other approaches and interaction with concurrency control.

Test Books:

1. Elmasri Navrate, Data Base Management System, Pearson Education, 2008.
2. Raghurama Krishnan, Johannes Gehrke, Data Base Management Systems, TMH, 3rd edition, 2008.
3. C. J. Date, Introduction to Database Systems, Pearson Education, 2009.
4. Silberschatz, Korth, Database System Concepts, McGraw hill, 5th edition, 2005.
5. Rob, Coronel & Thomson, Database Systems Design: Implementation and Management, 2009.

C PROGRAMMING LAB			
Course Code:	**MAI181**	**Course Credits:**	**2**
Course Category:	**CC-L**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials + Lab (Hrs./Week):	**0 + 0+3**	**Mid Sem. Exam Hours:**	**-**
Total No. of Lectures (L + T+P):	**0+ 0+10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1 To develops fundamental understanding C programming environment.
2 To create programming logics and learn C language programming concepts.
3 To design and develop algorithms and programs with different data declarations, initialization and related operations.
4 To develop the ability to define and manage functions, array, structures, pointers etc. based on program objective.
5 To understand and develop C programs to handle computer files, their usage and perform various operations on files.

COURSE OUTCOMES
At the end of the course the students should be able to:
1 Understand the C programming fundamentals.
2 Understand the use of various programming concepts and techniques.
3 Understand the C data types and operators with their applications.
4 Understand C by using arrays, functions, structures and union.
5 Develop the Programs in C using its advance features.

LIST OF EXPERIMENTS:

1. Write a program to find the sum of the digits of a number.
2. Write a program to calculate factorial of a number using recursion.
3. Write a program to find the reverse of a given number.
4. Write a program to check whether the year is leap or not.
5. Write a program to take marks of a student of 5 subjects as an input and print the grade.

marks<40 = FAIL
marks>=40 and <=59 GOOD
marks>=59 and <80 EXCELLENT marks>=80 = OUTSTANDING

6. Perform program number 5 using switch case statement.
7. Write a program to compute the roots of a quadratic equations.
8. Write a program to compute the length of a string using While Loop.

9. Write a program to print the following pattern: -

A)
```
*
**
***
****
```

B)
```
    *
   * *
  *  * *
 *   * * *
```

C)
```
    0
   1 2
  3 4 5
 6 7 8 9
```

10. Write a program to compute and display the product of two matrices.
11. Write a program to illustrate the difference between call by value and call by reference.
12. Write a program to check whether a given string is palindrome or not.
13. Create a structure called STUDENT having name, reg no., and class as its field. Compute the size of structure STUDENT.
14. Write a program to compute the length of a string using pointers.
15. Write a program to create a file, input data and display its content.

DATA BASE MANAGEMENT SYSTEM LAB			
Course Code:	**MAI183**	**Course Credits:**	**2**
Course Category:	**CC-L**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Lectures + Tutorials + Lab (Hrs./Week):	**0 + 0+3**	**Mid Sem. Exam Hours:**	**-**
Total No. of Lectures (L + T+P):	**0+ 0+10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

- Understand the basics and functions of MS excel.
- Clear understanding and use of data validations and templates.
- Purpose of sorting and filtering features.
- Use of reports in business organizations.
- Purpose and advantage of charts for top management in any work place.

COURSE OUTCOMES

At the end of the course the students should be able to:

- Learn to understand the functions in Excel.
- Understand the validations.
- Make reports in excel.
- Learn to work with pivot tables.
- Learn how to make charts in MS excel.

LIST OF EXPERIMENTS:

1 Write the queries for Data Manipulation and Data Definition Language.
2 Write SQL queries using logical operations and operators.
3 Write SQL query using group by function.
4 Write SQL queries for sub queries, nested queries
5 Write SQL queries to create views.
6 Write an SQL query to implement JOINS.
7 Write a query for extracting data from more than one table.
8. Write a query to understand the concepts for ROLL BACK, COMMIT & CHECK POINTS.
9. Create tables according to the following definition.
Create table deposit (actno varchar2(5) ,cname varchar2(18) , bname varchar2(18) , amount number(8,2) ,adate date);
Create table branch(bname varchar2(18),city varchar2(18));
Create table customers(cname varchar2(19) ,city varchar2(18));
Create table borrow(loanno varchar2(5), cname varchar2(18), bname varchar2(18), amount number (8,2));
10. Retrieve all data from employee, jobs and deposit.
(1) Give details of account no. and deposited rupees of customers having account opened between dates 01-01-06 and 25-07-06.
(2) Display all jobs with minimum salary is greater than 4000.
(3) Display name and salary of employee whose department no is 20. Give alias name to name of employee.
(4) Display employee no,name and department details of those employee whose department lies in(10,20)

SEMESTER II

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	MAI102	JAVA Programming	3	0	0	3	CC8 / SEC
2	MAI104	Data Structure and Algorithms	3	0	0	3	CC9 / SEC
3	MAI106	Compiler Design	3	0	0	3	CC10
4	MAI108	Expert System Design	3	0	0	3	CC11
5	MAI110	Natural Language Processing	3	0	0	3	CC12
6		Elective-1	3	0	0	3	E1 / DSE
7		Elective-2	3	0	0	3	E2 / DSE
8	EN532	Fundamental of Language Science	3	0	0	3	OE2 / AECC
9	MAI182	Data Structure and Algorithms Lab	0	0	3	2	CC-L3 / SEC
10	MAI184	JAVA Programming Lab	0	0	3	2	CC-L4 / SEC
11	GP	General Proficiency	Non Credit				
		Total Hours and Credits	**24**	**0**	**6**	**28**	

JAVA PROGRAMMING			
Course Code:	**MAI102**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs./Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

- To teach principles of object-oriented programming paradigm including abstraction, encapsulation, inheritance, and polymorphism.
- To impart fundamentals of object-oriented programming in Java, including defining classes, invoking methods, using class libraries, etc.
- To inculcate concepts of inheritance to create new classes from existing one & design the classes needed given a problem specification;
- To familiarize the concepts of packages and interfaces.
- To demonstrate the concept of event handling used in GUI.

COURSE OUTCOMES

At the end of the course the students should be able to:

- Demonstrate an introductory understanding of graphical user interfaces, multithreaded programming, and event-driven programming
- Analyze the necessity for Object Oriented Programming paradigm over structured programming and become familiar with the fundamental concepts in OOP like encapsulation, Inheritance and Polymorphism
- Design and develop java programs, analyze, and interpret object-oriented data and report results.
- Design an object-oriented system, AWT components and multithreaded processes as per needs and specifications.
- Prepare UML diagrams for software system

UNIT I OBJECT-ORIENTED PROGRAMMING

Concept of object-oriented programming (OOP), benefits of OOP, application of OOP, Java history, Java features, Java streaming, Java and Internet, Java contribution to Internet: Java applets, security, portability; Java environment, Java library, Java program structure, Java program, Java Virtual Machine (JVM) architecture, Just In Time compiler (JIT), data type, variables and arrays, operators, control statements, object-oriented paradigms; abstraction, encapsulation, inheritance, polymorphism, Java class and OOP implementation

UNIT II DATA TYPE, OPERATORS AND CONTROL STATEMENT

Data types, Java key words, identifiers, constants, variables, declaration and scope of the variable, symbolic constant, type casting, arithmetic operator, relational operator, logical operator, assignment operator, increment and decrement operator, conditional operator, bitwise operator, ?: operator, arithmetic expressions, expressions, type conversions in expressions, mathematical functions, more data types: arrays, strings, vectors, wrappers classes, program control statements: decision making and branching: if, if….else, else….if, else if ladder,

switch, decision making and looping: while, do….while, for.

UNIT III CLASSES, OBJECTS AND METHODS

Java class libraries, class fundamentals, object, methods, adding variables, add methods, creating objects, accessing class members, constructors, methods overloading, static members, nesting of methods, inheritance: extending a class, overriding methods, final variables and methods, final classes, finalizer methods, abstract methods and classes, visibility control, exception handling fundamental.

UNIT IV INTERFACES AND PACKAGES

Interfaces, extending interfaces, implementing interfaces, interfaces references, accessing interface variable, creating queue interface, variable in interfaces, packages, finding a packages and classpath, package and member access, Java API package, system package, naming conventions, creating package, accessing a package, adding a class to a package, hiding classes,

UNIT V MULTITHREADING AND APPLET PROGRAMMING

Multithreading programming: creating threads, thread class and runnable interface extending the thread class, stopping and blocking a thread, life cycle of a thread, thread methods, thread exceptions, thread priority, synchronization, thread communication using notify(), wait(), and notify all(), applet programming : applet basic, applets architecture, a complete applet skeleton, building applets code, applets life cycle, creating a executable applet, designing a web page, applets tag, passing parameters to applets, applets and HTML.

Textbook:

1. Programming with JAVA, E. Balagurusawamy, Tata McGraw Hill, 1998.
2. JAVA Beginner's guide, Herbert Schildt, Tata McGraw Hill, 2007.
3. Java How to Program, Deitel & Deitel, Prentice-Hall, 1999.
4. The Complete Reference JAVA 2, Herbert Schildt, 5th Edition, Tata McGraw Hill, 2002.
5. The Complete Reference JAVA 2, Herbert Schildt, 7th Edition, Tata McGraw Hill, 2009.

Reference Books:

1. The Java Programming Language, Ken Arnold, James Gosling, Addison-Wesley, 1996.
2. How to Program Java, Peter Coffee, Ziff-Davis Press, 1996.

DATA STRUCTURE AND ALGORITHMS			
Course Code:	**MAI104**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

- To emphasize the importance of appropriate data structure in developing and implementing efficient algorithms
- Understand basic data structures such as arrays, stacks, queues, hash tables and linked list
- To analyze the asymptotic performance of various algorithms
- Solve problems using graphs, trees and heaps
- Apply important algorithmic design paradigms and methods of analysis

COURSE OUTCOMES

At the end of the course the students should be able to:

- Define basic static and dynamic data structures and relevant standard algorithms for them.
- Select basic data structures and algorithms for autonomous realization of simple programs or program parts.
- Determine and demonstrate bugs in program, recognise needed basic operations with data structures
- Formulate new solutions for programming problems or improve existing code using learned algorithms and data structures
- Evaluate algorithms and data structures in terms of time and memory complexity of basic operations.

UNIT I INTRODUCTION TO DATA STRUCTURES

Abstract data types, sequences as value definitions, data types in C, pointers in C, data structures and C, arrays in C, array as ADT, one dimensional array, Implementing one dimensional array, array as parameters, two dimensional array, structures in C, implementing structures, Unions in C, implementation of unions, structure parameters, allocation of storage and scope of variables, recursive definition and processes: factorial function, fibonacci sequence, recursion in C, efficiency of recursion, hashing: hash function, open hashing, closed hashing: linear probing, quadratic probing, double hashing, rehashing, extendible hashing.

UNIT II STACK, QUEUE AND LINKED LIST

Stack definition and examples, primitive operations, example -representing stacks in C, push and pop operation implementation, queue as ADT, C Implementation of queues, insert operation, priority queue, array implementation of priority queue, inserting and removing nodes from a list-linked implementation of stack, queue and priority queue, other list structures, circular lists: stack and queue as circular list - primitive operations on circular lists, header nodes, doubly linked lists, addition of long positive integers on circular and doubly linked list.

UNIT III TREES

Binary trees: operations on binary trees, applications of binary trees, binary tree representation, node representation of binary trees, implicit array representation of binary tree, binary tree traversal in C, threaded

binary tree, representing list as binary tree, finding the Kth element, deleting an element, trees and their applications: C representation of trees, tree traversals, evaluating an expression tree, constructing a tree.

UNIT IV SORTING AND SEARCHING

General background of sorting: efficiency considerations, notations, efficiency of sorting, exchange sorts: bubble sort; quick sort; selection sort; binary tree sort; heap sort, heap as a priority queue, sorting using a heap, heap sort procedure, insertion sorts: simple insertion, shell sort, address calculation sort, merge sort, radix sort, sequential search: indexed sequential search, binary search, interpolation search.

UNIT V GRAPHS

Application of graph, C representation of graphs, transitive closure, Warshall's algorithm, shortest path algorithm, linked representation of graphs, Dijkstra's algorithm, graph traversal, traversal methods for graphs, spanning forests, undirected graph and their traversals, depth first traversal, application of depth first traversal, efficiency of depth first traversal, breadth first traversal, minimum spanning tree, Kruskal's algorithm, round robin algorithm.

Text Books:

1. Aaron M. Tenenbaum, Yeedidyah Langsam, Moshe J. Augenstein, 'Data structures using C', Pearson Education, 2004 / PHI.
2. E. Balagurusamy, 'Programming in Ansi C', Second Edition, TMH, 2003.
3. Robert L. Kruse, Bruce P. Leung Clovis L.Tondo, 'Data Structures and Program Design in C', Pearson Education, 2000 / PHI.

COMPILER DESIGN			
Course Code:	**MAI106**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
● Acquire knowledge of different phases and passes of the compiler and able to use the compiler tools like LEX, YACC, etc.
● Understand the parser and its types i.e., Top-Down and Bottom-up parsers and construction of LL, SLR, CLR, and LALR parsing table.
● Understand backend of compiler: intermediate code, Code optimization Techniques and Error Recovery mechanisms
● Understand issues of run time environments and scheduling for instruction level parallelism.

COURSE OUTCOMES
At the end of the course the students should be able to:
● design different types of compiler tools to meet the requirements of the realistic constraints of compilers.
● Implement the compiler using syntax-directed translation method and get knowledge about the synthesized and inherited attributes.
● Acquire knowledge about run time data structure like symbol table organization and different techniques used in that.
● Understand the target machine's run time environment, its instruction set for code generation and techniques used for code optimization.
● . Understand and Implement a Parser.

UNIT I Introduction to Compiler

Phases and passes, Bootstrapping, Finite state machines and regular expressions and their applications to lexical analysis, Optimization of DFA-Based Pattern Matchers implementation of lexical analyzers, lexical-analyzer generator, LEX compiler, Formal grammars, and their application to syntax analysis, BNF notation, ambiguity, YACC. The syntactic specification of programming languages: Context free grammars, derivation, and parse trees, capabilities of CFG.

UNIT II Basic Parsing Techniques

Parsers, Shift reduce parsing, operator precedence parsing, top down parsing, predictive parsers Automatic Construction of efficient Parsers: LR parsers, the canonical Collection of LR(0) items, constructing SLR parsing tables, constructing Canonical LR parsing tables, Constructing LALR parsing tables, using ambiguous grammars, an automatic parser generator, implementation of LR parsing tables

UNIT III Syntax-directed Translation

Syntax-directed Translation schemes, Implementation of Syntax- directed Translators, Intermediate code, postfix notation, Parse trees & syntax trees, three address code, quadruple & triples, translation of assignment statements, Boolean expressions, statements that alter the flow of control, postfix translation, translation with a top-down parser. More about translation: Array references in arithmetic expressions, procedures call, declarations, and case statements.

UNIT IV Symbol Tables

Data structure for symbols tables, representing scope information. Run-Time Administration: Implementation of simple stack allocation scheme, storage allocation in block structured language. Error Detection & Recovery: Lexical Phase errors, syntactic phase errors
semantic errors.

UNIT V Code Generation:

Design Issues, the Target Language. Addresses in the Target Code, Basic Blocks and Flow Graphs, Optimization of Basic Blocks, Code Generator. Code optimization: Machine-Independent Optimizations, Loop optimization, DAG representation of basic blocks,
value numbers and algebraic laws, Global Data-Flow analysis.

Textbooks:

1. K. Muneeswaran,Compiler Design,First Edition,Oxford University Press.
2. P. Bennet, “Introduction to Compiler Techniques”, Second Edition, Tata McGraw-Hill,2003.
3. Henk Alblas and Albert Nymeyer, “Practice and Principles of Compiler Building with C”, PHI, 2001.
4. Aho, Sethi & Ullman, "Compilers: Principles, Techniques and Tools”, Pearson Education
5. V Raghvan, “ Principles of Compiler Design”, TMH
6. Kenneth Louden,” Compiler Construction”, Cengage Learning. Charles Fischer and Ricard LeBlanc,” Crafting a Compiler with C”, Pearson Education

EXPERT SYSTEM DESIGN			
Course Code:	**MAI108**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs./Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES

- To study the idea of intelligent agents and search methods.
- To study about representing knowledge.
- To study the reasoning and decision making in uncertain world.
- To construct plans and methods for generating knowledge.
- To study the concepts of expert systems.

COURSE OUTCOMES

At the end of the course the students should be able to:

- Understands the basics of Artificial intelligence and the various searching techniques
- Analyses the knowledge representation through forward and backward chaining techniques
- Applies and articulate the knowledge over the design of semantic nets
- Evaluates the architecture and frameworks of Truth Maintenance systems through machine Learning algorithms
- Creates the design of real time AI and expert systems through various case studies

UNIT I

Define expert system, problem domain and knowledge domain, the advantages of an expert system, general stages in the development of an expert system, general characteristics of an expert system, history, and uses of expert systems today, rule-based expert systems, procedural and nonprocedural paradigms, characteristics of artificial neural systems. -The study of logic, difference between formal logic and informal logic, meaning of knowledge, how knowledge can be represented.

UNIT II

Semantic nets, how to translate semantic nets into PROLOG, limitations of semantic nets, schemas, frames, and their limitations, how to use logic and set symbols to represent knowledge, the meaning of propositional and first order predicate logic, quantifiers, imitations of propositional and predicate logic. Trees, lattices, and graphs, state, and problem spaces, AND-OR trees and goals, methods of inference, rules of inference, limitations of propositional logic, logic systems, resolution rule of inference, resolution systems, and deduction, shallow and causal reasoning, applying resolution to first-order predicate logic, forward and backward chaining, additional methods of reference, Meta knowledge, the Markov decision process.

UNIT III

Uncertainty and theories devised to deal with, types of errors attributed to uncertainty, errors associate, with

induction, features of classical probability, experimental and subjective probabilities, compound and conditional probabilities, hypothetical reasoning and backward induction, temporal reasoning.

UNIT IV

Markov chains, odds of belief, sufficiency and necessity, role of uncertainty in inference chains, implications of combining evidence, role of inference nets in expert systems, how probabilities are propagated. Sources of uncertainty in rules, methods of dealing with uncertainty, Dempster-Shafer theory, theory of uncertainty based on fuzzy logic, commercial applications of fuzzy logic. How to select an appropriate problem?

UNIT V

Stages in the development of an expert system, types of errors to expect in the development stages, the role of the knowledge engineer in the building of expert systems, the expected life cycle of an expert system, how to do a life cycle model.

Textbook:

[1] J. Giarratano and G. Riley, "Expert Systems -- Principles and Programming". 4th Edition, PWS Publishing Company, 2004.
[2] Durkin, J., Expert systems Design and Development, Macmillan, 1994 2. Elias M. Awad, Building Expert Systems, West Publishing Company 1996.

NATURAL LANGUAGE PROCESSING			
Course Code:	**MAI110**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs./Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
● 1 To Understanding the basics of natural language processing and understand various steps in it.
● 2 To introduce the fundamentals of language processing from the algorithmic viewpoint.
● To discuss various issues that make natural language processing a hard task.
● Understand the importance and need of information retrieval system.
● To discuss some well-known applications of natural language processing.

COURSE OUTCOMES
At the end of the course the students should be able to:
● Appreciate the fundamental concepts of natural language processing.
● Design algorithms for natural language processing tasks.
● Develop useful systems for language processing and related tasks involving text processing.
● Learn about machine translation.
● Ability to perform independent research and analysis.

UNIT I INTRODUCTION

Natural Language Processing tasks in syntax, semantics, and pragmatics – Issues – Applications – The role of machine learning – Probability Basics –Information theory – Collocations -N-gram Language Models –Estimating parameters and smoothing – Evaluating language models.

UNIT II WORD LEVEL AND SYNTACTIC ANALYSIS

Word Level Analysis: Regular Expressions-Finite-State Automata-Morphological Parsing- Spelling Error Detection and correction-Words and Word classes-Part-of Speech Tagging. Syntactic Analysis: Context-free Grammar-Constituency- Parsing-Probabilistic Parsing**.**

UNIT III SEMANTIC ANALYSIS AND DISCOURSE PROCESSING

Semantic Analysis: Meaning Representation-Lexical Semantics- Ambiguity-Word Sense Disambiguation. Discourse Processing: cohesion-Reference Resolution- Discourse Coherence and Structure.

UNIT IV NATURAL LANGUAGE GENERATION AND MACHINE TRANSLATION

Natural Language Generation: Architecture of NLG Systems- Generation Tasks and Representations Application of NLG. Machine Translation: Problems in Machine Translation- Characteristics of Indian Languages- Machine Translation Approaches- Translation involving Indian Languages.

UNIT V : INFORMATION RETRIEVAL AND LEXICAL RESOURCES

Information Retrieval: Design features of Information Retrieval Systems-Classical, Non- classical, Alternative Models of Information Retrieval – valuation Lexical Resources: WorldNet-Frame Net-Stemmers-POS Tagger-

Research Corpora.

Text and Reference Books:

[1] Daniel Jurafsky , James H. Martin , “Speech & language processing”, Pearson publications.
[2] Allen, James. Natural language understanding. Pearson, 1995.
[3] Akshar Bharti, Vineet Chaitanya and Rajeev Sangal, “NLP: A Paninian Perspective”, Prentice Hall, New Delhi.
[4] L. M. Ivansca, S. C. Shapiro, “Natural Language Processing and Language Representation”, AAAI Press, 2000.
[5] T. Winograd, Language as a Cognitive Process, Addison-Wesley.

COMPUTER VISION			
Course Code:	**MAI118**	**Course Credits:**	**3**
Course Category:	**E1/DSE**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
● To understand basics of graphics theorem and algorithms
● understand visual processing from both “bottom-up” (data oriented) and “top-down” (goals oriented) perspectives
● understand the roles of image transformations and their invariances in pattern recognition and classification
● understand in depth at least one important application domain, such as face recognition, detection, or interpretation
● understand basics of object detection and motion.

COURSE OUTCOMES
At the end of the course the students should be able to:
● decompose visual tasks into sequences of image analysis operations, representations, specific algorithms, and inference principles
● analyses the robustness, brittleness, generalizability, and performance of different approaches in computer vision
● to describe key aspects of how biological visual systems encode, analyze, and represent visual information
● be able to think of ways in which biological visual strategies might be implemented in machine vision, despite the enormous differences in hardware
● work in graphics industries with more knowledge.

UNIT I Digital Image Formation and low-level processing

Overview and State-of-the-art, Fundamentals of Image Formation, Transformation: Orthogonal, Euclidean, Affine, Projective, etc; Fourier Transform, Convolution and Filtering, Image Enhancement, Restoration, Histogram Processing.

UNIT II Depth estimation and multi-camera views

Perspective, Binocular Stereopsis: Camera and Epipolar Geometry; Homography, Rectification, DLT, RANSAC, 3-D reconstruction framework; Auto-calibration.

UNIT III Feature Extraction

Edges - Canny, LOG, DOG; Line detectors (Hough Transform), Corners - Harris and Hessian Affine, Orientation Histogram, SIFT, SURF, HOG, GLOH, Scale-Space Analysis- Image Pyramids and Gaussian derivative filters, Gabor Filters and DWT.

UNIT IV Image Segmentation

Region Growing, Edge Based approaches to segmentation, Graph-Cut, Mean-Shift, MRFs, Texture Segmentation, Object detection. Pattern Analysis Clustering: K-Means, K-Medoids, Mixture of Gaussians, Classification: Discriminant Function, Supervised, Un-supervised, Semi-supervised; Classifiers: Bayes, KNN, ANN models; Dimensionality Reduction: PCA, LDA, ICA; Non-parametric methods.

UNIT V Motion Analysis

Background Subtraction and Modeling, Optical Flow, KLT, Spatio-Temporal Analysis, Dynamic Stereo; Motion parameter estimation. Shape from X: Light at Surfaces; Phong Model; Reflectance Map; Albedo estimation; Photometric Stereo; Use of Surface Smoothness Constraint; Shape from Texture, color, motion, and edges.

Text and Reference Books:

[1] Richard Szeliski, Computer Vision: Algorithms and Applications, Springer-Verlag London Limited 2011.
[2] Computer Vision: A Modern Approach, D. A. Forsyth, J. Ponce, Pearson Education, 2003.
[3] Richard Hartley and Andrew Zisserman, Multiple View Geometry in Computer Vision, Second Edition, Cambridge University Press, March 2004.
[4] Christopher M. Bishop; Pattern Recognition and Machine Learning, Springer, 2006
[5] R.C. Gonzalez and R.E. Woods, Digital Image Processing, Addison- Wesley, 1992.
[6] K. Fukunaga; Introduction to Statistical Pattern Recognition, Second Edition, Academic Press, Morgan Kaufmann, 1990.

DATA STRUCTURE AND ALGORITHMS LAB			
Course Code:	**MAI182**	**Course Credits:**	**2**
Course Category:	**CC-L**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**2P**
No. of Labs (Hrs/Week):	**03**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Introduce the concept of data structures through ADT including List, Stack, Queues .
2. To design and implement various data structure algorithms.
3. To introduce various techniques for representation of the data in the real world.
4. To develop application using data structure algorithms
5. Compute the complexity of various algorithms.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Select appropriate data structures as applied to specified problem definition
2. Implement operations like searching, insertion, and deletion, traversing mechanism etc. on various data structures.
3. Students will be able to implement Linear and Non-Linear data structures.
4. Implement appropriate sorting/searching technique for given problem.
5. Design advance data structure using Non-Linear data structure

List of Experiments:

1. Run time analysis of Fibonacci Series
2. Study and Application of various data Structure
3. Study and Implementation of Array Based Program
   a. Searching (Linear Search, Binary Search)
   b. Sorting (Bubble, Insertion, Selection, Quick, Merge etc)
   c. Merging
4. Implementation of Link List
   a. Creation of Singly link list, Doubly Linked list
   b. Concatenation of Link list
   c. Insertion and Deletion of node in link list
   d. Splitting the link list into two link list
5. Implementation of STACK and QUEUE with the help of
   a. Array
   b. Link List
6. Implementation of Binary Tree, Binary Search Tree, Height Balance Tree
7. Write a program to simulate various traversing Technique
8. Representation and Implementation of Graph
   a. Depth First Search
   b. Breadth First Search
   c. Prims Algorthim
   d. Kruskal's Algorithms
   e. Implementation of Hash Table

JAVA PROGRAMMING LAB			
Course Code:	**MAI184**	**Course Credits:**	**2**
Course Category:	**CC-L2**	**Course (U / P)**	**P**
Course Year (U / P):	**1P**	**Course Semester (U / P):**	**1P**
No. of Labs (Hrs/Week):	**03**		
Total No. of Labs :	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
● Knowledge of basic Object-Oriented paradigm, practices, and application.
● A general understanding of class, object, and methods.
● To make the student learn an object-oriented way of solving problem using java.
● To make the students, to write programs that connects to a database and be able to perform various operations.
● To make the students able to create the graphical user interface using Applets, AWT Components.

COURSE OUTCOMES
● At the end of the course the students should be able to:
● Basic knowledge and understanding of object-oriented programming.
● Ability to apply OOPs concept in real life problem.
● Ability to design, develop, maintain, and evaluate large-scale software systems.
● To produce efficient, reliable, robust, and cost-effective software solutions using Java.
● Ability to perform independent research and analysis.

List of Experiments:

1. Write a separate Java Code to implement each of the following: Class, Command Line. Argument, how to enter value through keyboard
2. Write a separate Java code to implement each of the following data types: Variable, Constant, Arrays, Strings, Vectors, Wrapper Classes, Type Casting
3. Write a separate Java code to implement each of the following operators: Arithmetic operator, Relational operator, Logical operator, Assignment operator, increment & Decrement operator, Conditional operator, Bitwise operator, ? Operator
4. Write a separate Java code to implement each of the following control statements: Decision statement, Loop statement and Brach statements
5. Write a separate Java code to implement each of the following sorting: Bubble Sort, Selection sort, Insertion Sort, Merge sort
6. Write a separate Java Code to implement each of the following: Class, Object, Constructors, Method, Method overloading and Method overriding
7. Write a separate Java Code to implement each of the following: Final variable, final class, final method, abstract class, abstract method and concrete method
8. Write a separate Java code to implement each of the following: OOPs concepts: Abstraction, Polymorphism, Encapsulation, Inheritance
9. Write a separate Java code to implement each of the following: Exception handling with Try, Catch, Throws, Finally Multiple catch statement with the following exceptions: Arithmetic Exception, ArrayOutOfBoundsException and ArrayStoreException.
10: Write a separate Java code to implement each of the following: Interface and How to import Packages.

SEMESTER III

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	MAI201	Analysis and Design of Algorithm	3	0	0	3	CC13
2	MAI203	Machine Learning	2	0	0	2	CC14
3	MAI205	Soft Computing Techniques	3	0	0	3	CC15
4	MAI207	Python Programming	2	0	0	2	CC16 / SEC
5		Elective-3	3	0	0	3	E3 / DSE
6		Elective-4	3	0	0	3	E4 / DSE
7	MA033	Numerical Algorithms for Machine Learning	3	0	0	3	GE1
8	MAI291	Minor Project	0	0	10	5	MP1 / E
9	MAI281	Analysis and Design of Algorithm Lab	0	0	3	2	CC-L5 / SEC
10	MAI283	Machine Learning using Python Lab	0	0	3	2	CC-L6 / SEC
11	GP	General Proficiency	Non Credit				
		Total Hours and Credits	**19**	**0**	**16**	**28**	

ANALYSIS AND DESIGN OF ALGORITHM			
Course Code:	**MAI201**	**Course Credits:**	3
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Analyze the asymptotic performance of algorithms.
2. Write rigorous correctness proofs for algorithms.
3. Demonstrate a familiarity with major algorithms and data structures.
4. Apply important algorithmic design paradigms and methods of analysis.
5. Synthesize efficient algorithms in common engineering design situations.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Argue the correctness of algorithms using inductive proofs and invariant
2. Explain the major graph algorithms and their analyses. Employ graphs to model engineering problems, when appropriate. Synthesize new graph algorithms and algorithms that employ graph computations as key components, and analyze them.
3. Describe the divide-and-conquer paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize divide-and-conquer algorithms. Derive and solve recurrences describing the performance of divide-and-conquer algorithms.
4. Describe the dynamic-programming paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize dynamic-programming algorithms, and analyze them.
5. Analyze worst-case running times of algorithms using asymptotic analysis.

UNIT I BASIC CONCEPT OF ALGORITHMS

What is an algorithm, notion of algorithm, fundamentals of algorithmic solving, Mathematics for Algorithmic sets, Functions and Relations, Vectors and Matrices, linear Inequalities and Linear Equations, fundamentals of analysis framework, the efficient algorithm, Average, Best and Worst case analysis, asymptotic notation, Analyzing Control statement, Loop invariant and the correctness of the algorithm.

UNIT II MATHMATICAL ASPECTS AND ANALYSIS OF ALGORITHM

Mathematical analysis of non- recursive algorithm , mathematical analysis of recursive algorithm, example: fibonacci numbers, empirical analysis of algorithms, algorithm visualization.

UNIT III ANALYSIS OF SORTING AND SEARCHING ALGORITHM

Sorting Algorithms and Analysis: Bubble sort, Selection sort, Insertion sort,Shell sort Heap sort, Sorting in linear time: Bucket sort, Radix sort and Counting sort. sequential search and brute-force string matching, divide

and conquer, merge sort, binary search, binary tree, traversal and related properties, depth first search and breadth forst search.

UNIT IV ALGORITHM TECHNIQUES

Transform and conquer, presorting, balanced search trees, avl trees, heaps and heap sort, dynamic programming, Warshall's and Floyd's algorithm, optimal binary search trees, greedy techniques, Prim's algorithm, Kruskal's algorithm, Dijkstra's algorithm, Huffman trees.

UNIT V ALGORITHM DESIGN METHODS

Backtracking, n-Queen's problem, Hamiltonian circuit problem, subset-sum problem, branch and bound, assignment problem, knapsack problem, traveling salesman problem.
Text Books:

1. Anany Levitin, "Introduction to the Design and Analysis of Algorithm", Pearson Education Asia, 2003
2. T.H. Cormen, C.E. Leiserson, R. L. Rivest and C. Stein, "Introduction to Algorithm", PHI Pvt. Ltd., 2001
3. Sara Baase and Allen Van Gelder,"Computer Algorithms-Introduction to the Design and Analysis ", Pearson Education Asia, 2003
4. A. V. Aho, J.E. Hopcroft and J.D. Ullman, "the Design and Analysis of Computer Algorithms", Pearson Education Asia, 2003.

MACHINE LEARNING			
Course Code:	**MAI203**	**Course Credits:**	**2**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**
COURSE OBJECTIVES			
1 To introduce students to the basic concepts and techniques of Machine Learning			
2 To develop skills of using recent machine learning software for solving practical problem			
3 To gain experience of doing independent study and research.			
4 Understanding of Python Programming and its module			
5 Understanding of Deep learning.			
COURSE OUTCOMES			
At the end of the course the students should be able to:			
1Develop an appreciation for what is involved in Learning models from data			
2 Understand a wide variety of learning algorithms			
3 Apply the algorithms to a real problem, optimize the models learned and report on the expected accuracy that can be achieved by applying the models			
4 Understand how to evaluate models generated from data			
5 Understanding of ML models			

UNIT I INTRODUCTION

What is Machine Learning, Types of Machine Learning, Supervised Learning, Unsupervised Learning, Reinforcement Learning, Applications of Machine Learning – Stock Price Prediction, Face Recognition, Handwriting Recognition, Image Recognition, Virtual Personal Assistants, Medical Diagnosis, Online Fraud Detection.

UNIT II SUPERVISED LEARNING (REGRESSION/CLASSIFICATION)

Basic methods: Distance-based methods, Nearest-Neighbours, Decision Trees, Naive Bayes Linear models: Linear Regression, Logistic Regression, Generalized Linear Models, Support Vector Machines, Nonlinearity and Kernel Methods.

UNIT III UNSUPERVISED LEARNING

Clustering: K-means/Kernel K-means, Self-Organizing Maps. Dimensionality Reduction: PCA and kernel PCA Matrix Factorization and Matrix Completion Generative Models (mixture models and latent factor models).

UNIT IV ARTIFICIAL NEURAL NETWORKS

Biological Neurons and Biological Neural Networks, Artificial Neural Network, Types of Neural Network, Perceptron, History behind Perceptron, Importance of Perceptron, Working of Perceptron, Perceptron Learning, Percetron Learning Rule, Perceptron Learning of AND & OR gate, XOR gate, Activation functions, Binary Activation function, ReLU, Sigmoid, Hyperbolic, Softamax Activation function, Multilayer Perceptrons, Back

propagation Neural Networks, and Feed-Forward Neural Networks, Applications and Future of Neural Networks.

UNIT V SELECTED TOPICS

Ensemble Methods (Boosting, Bagging, Random Forests), Sparse Modeling and Estimation, Modeling Sequence/Time-Series Data, Deep Learning and Feature Representation Learning, Recent trends in various learning techniques of machine learning and classification methods, Case studies in interdisciplinary domains.

Text and Reference Books:

[1] Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow by AurélienGéron, O'Reilly publication
[2] An Introduction to Statistical Learning with Applications in R by Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, Springer publication (springer.com)
[3] Python Machine Learning: Machine Learning and Deep Learning with Python, scikit- learn, and Tensor Flow 2, Publisher: Packt Publishing (December 12, 2019), Language: English, ISBN-10: 1789955750, ISBN-13: 978-1789955750
[4] Machine Learning: The Absolute Complete Beginner's Guide to Learn and Understand Machine Learning From Beginners, Intermediate, Advanced, To Expert Concepts by Steven Samelson Publisher: Independently published (May 5, 2019)Language: English,ISBN-10: 1096853205,ISBN-13: 978-109685320

SOFT COMPUTING TECHNIQUES			
Course Code:	**MAI205**	**Course Credits:**	**3**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. The primary objective of this course is to provide an introduction to the basic principles, techniques, and applications of soft computing.
2. Understanding of the basic areas of Soft Computing including Artificial Neural Networks, Fuzzy Logic and Genetic Algorithms.
3. Provide the mathematical background for carrying out the optimization associated with neural network learning.
4. Aim of this course is to develop some familiarity with current research problems and research methods in Soft Computing by working on a research or design project.
5. Genetic algorithms, its applications and advances.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Apply basics of Fuzzy logic and neural networks..
2. Discuss the ideas of fuzzy sets, fuzzy logic and use of heuristics based on human
3. Describe with genetic algorithms and other random search procedures useful while seeking global optimum in self-learning situations
4. Develop some familiarity with current research problems and research methods in Soft Computing Techniques
5. Experience Relate with neural networks that can learn from available examples and generalize to form appropriate rules for inference systems

UNIT I INTRODUCTION

Introduction to Soft Computing; Definition, requirement, necessity and adequacy; various dialects of soft computing – Evolutionary Algorithms, Fuzzy Sets and Fuzzy Logic, Artificial Neural Networks - their suitability in Searching, optimization, decision matching and pattern related problems; potential areas of applications.

UNIT II FUZZY SETS AND FUZZY LOGIC

Introduction to fuzzy sets and fuzzy logic; difference between classical and fuzzy sets; chance vs fuzziness; limitations of fuzzy systems; typical shapes of membership functions and their usage; operations on fuzzy sets: compliment, intersection, union; combinations on operations, aggregation operation.

UNIT III FUZZY RELATIONS AND FUZZY SYSTEMS

Cartesian Product; Classical Relations and Fuzzy Relations; Cardinality, operations and properties of crisp and fuzzy relations; Composition of operations, Fuzzy cartesian product; The linguistic variables, Reasoning in fuzzy logic, Fuzzification and defuzzification; Mamdani and Sugano Fuzzy Inference Systems.

UNIT IV NEURAL NETWORK

Overview of biological neurons; McCulloch-Pitts model, Rosenblatt's Perceptron model, difference, capabilities and limitations; Model of generic computational neuron; Basic activation functions; Basic Learning laws of neurons; Single layer and multilayer architectures; Feedforward and feedback networks.

UNIT V LEARNING FUNDAMENTALS

Learning paradigms, supervised and unsupervised learning, reinforced learning; back propagation algorithm; Radial basis neurons, Generalized Regression Neural network, Probabilistic Neural Networks; Competitive learning; Self Organizing Features Map, Hopfield networks, associative memories, applications of artificial neural networks. Elasticity vs plasticity dilemma, preprocessing, post processing, early stopping.

UNIT VI EVOLUTIONARY ALGORITHMS

Problems suitable and not suitable for applying evolutionary algorithms; Various dialects of evolutionary Algorithms; Terminology of Genetic Algorithms; Canonical Genetic Algorithm; Common representations and related reproduction operators; premature convergence, schema theorem, minimal deceptive problem and Royal Road function; fitness function, Roulette wheel selection, Rank selection, Tournament Selection; termination criteria, survivor selection, population models; parallel implementations.

Text Books:

1. Artificial Neural Networks: An introduction to ANN Theory and Practice, Peteus J. Braspenning, PHI publication, 2005.
2. Fuzzy Logic: A spectrum of Theoretical and Practical issues, Paul P. Wang, pearson publication 2004.
3. An Introduction to Genetic Algorithms, Milanie Mitchell, MIT Press 1998.
4. A Genetic Algorithm Tutorial, Darrell Whitley.
5. Fuzzy Sets, Fuzzy logic, and Fuzzy Systems: Selected Papers- Lotfi Asker Zadeh, George J. Kilr, Bo yuan, 2005.
6. Foundations of Fuzzy logic and Soft Computing: 12th International Fuzzy conference proceeding, 2005.
7. Neural Networks Theory, Particia Melin, Oxford University press, 2003
8. Neural Networks Theory and Application, Oscar Castillo, Wiley Eastern publication
9. Genetic Algorithms in Search, Optimization and Machine Learning, David E Goldberg, Eddison-Wesley, 1988.

PYTHON PROGRAMMING			
Course Code:	**MAI207**	**Course Credits:**	**2**
Course Category:	**CC**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**02**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**30**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Master the fundamentals of writing Python scripts.
2. Learn core Python scripting elements such as variables and flow control structures.
3. Discover how to work with lists and sequence data.
4. Write Python functions to facilitate code reuse.
5. Use Python to read and write files.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Problem solving and programming capability.
2. Explain basic principles of Python programming language
3. Implement database and GUI applications.
4. Implement object oriented concepts
5. Define and demonstrate the use of built-in data structures “lists” and “dictionary”

UNIT I PYTHON BASICS, CONDITIONAL &LOOPS

Installation of Python and python Notebook, Python Objects, Number & Booleans, Strings, Container objects, Mutability of objects, Operators - Arithmetic, Bitwise, comparison and Assignment operators, Operators Precedence and associativity. Conditions (If else, if-elif-else), Loops (While ,for), Break and Continue statements, Range Functions

UNIT II STRING OBJECTS AND LIST OBJECTS

String object basics, String methods, Splitting and Joining Strings, String format functions, list object basics, list methods, List as stack and Queues, List comprehensions,

UNIT III TUPLES, SET, DICTIONARIES & FUNCTIONS

Tuples, Sets, Dictionary Object basics, Dictionary Object methods, Dictionary View Objects. Functions basics, Parameter passing, Iterators, Generator functions, Lambda functions, Map, Reduce, filter functions

UNIT IV OOPS CONCEPTS & WORKING WITH FILES

OOPS basic concepts, creating classes and Objects, Inheritance, Multiple Inheritance, working with files, Reading and writing files, Buffered read and write, Other File methods

UNIT V MODULES, EXCEPTION HANDLING & DATABASE PROGRAMMING

Using Standard Module, Creating new modules, Exceptions Handling with Try-except, Creating, inserting and retrieving Table, Updating and deleting the data. Data Ananlysis- Numpy variable, Numpy manipulation, Scipy, Pandas intro. Descriptive analysis, Pandas Input-output, Pandas manipulation, Pandas groupby

Text Books:

1. Head First Python 2e: A Brain-Friendly Guide Paperback – Illustrated, 16 by Paul Barry, Oreilly
2. Python: The Complete Reference Paperback – 20 March 2018 by Martin C. Brown (Author), TMH Publication
3. Let Us Python by Yashavant Kanetkar , 1 January 2019, BPB publication
4. Python Programming, A modular approach , First Edition, By Pearson Publication by Taneja Sheetal and Kumar Naveen , 26

September 2017

NUMERICAL ALGORITHMS FOR MACHINE LEARNING			
Course Code:	**MA033**	**Course Credits:**	**3**
Course Category:	**GE1**	**Course (U / P):**	**P**
Course Year (U / P):	**2P**	**Course Semester :**	**3P**
No. of Lectures + Tutorials (Hrs/Week):	**03 + 00**	**Mid Sem. Exam Hours:**	**1.5**
Total No. of Lectures (L + T):	**45 + 00**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. To explain the concept of how to learn patterns and concepts from data without being explicitly programmed in various IOT nodes.
2. To design and analyze various machine learning algorithms and techniques with a modern outlook focusing on recent advances.
3. Explore supervised and unsupervised learning paradigms of machine learning.
4. To explore Deep learning technique and various feature extraction strategies.
5. Acquire Data Analysis skills, and Create AI/ML solutions for various business problems.
COURSE OUTCOMES
At the end of the course the students should be able to:
1. Recognize the characteristics of machine learning that make it useful to real-world problems.
2. To compare and contrast pros and cons of various machine learning techniques and to get an insight of when to apply a particular machine learning approach.
3. To mathematically analyze various machine learning approaches and paradigms.
4. Characterize machine learning algorithms as supervised, semi-supervised, and unsupervised.
5. Understand the concept behind neural networks.

UNIT I

Introduction to learning Techniques: Supervised Learning (Regression/Classification), Basic methods: Distance-based methods, Nearest-Neighbours, Decision Trees, Naive Bayes, Linear models: Linear Regression, Logistic Regression, Generalized Linear Models, Support Vector Machines, Nonlinearity and Kernel Methods, Beyond Binary Classification: Multi-class/Structured Outputs, Ranking

UNIT II

Unsupervised Learning: Clustering: K-means/Kernel K-means, Dimensionality Reduction: PCA and kernel PCA, Matrix Factorization and Matrix Completion, Generative Models (mixture models and latent factor models)

UNIT III

Evaluating Machine Learning algorithms and Model Selection, Introduction to Statistical Learning Theory, Ensemble Methods (Boosting, Bagging, Random Forests).

Sparse Modeling and Estimation, Modeling Sequence/Time-Series Data, Deep Learning and Feature Representation Learning.

UNIT IV

Neurons and biological motivation, linear threshold units, Perceptrons: representational limitation and gradient descent training, Multilayer networks and backpropagation, Hidden layers and constructing intermediate, distributed representations, Overfitting, learning network structure, recurrent networks.

UNIT V

Scalable Machine Learning (Online and Distributed Learning) A selection from some other advanced topics, e.g., Semi-supervised Learning, Active Learning, Reinforcement Learning, Inference in Graphical Models, Introduction to Bayesian Learning and Inference. Introduction to Simulation Tool for Machine Learning (like WEKA, R, MATLAB). Recent trends in various learning techniques of machine learning and classification methods.

Text and References Books:

1. Kevin Murphy, Machine Learning: A Probabilistic Perspective, MIT Press.
2. Trevor Hastie, Robert Tibshirani, Jerome Friedman, The Elements of Statistical Learning, Springer.
3. Christopher Bishop, Pattern Recognition and Machine Learning, Springer.

4. Shai Shalev-Shwartz, Shai Ben-David, Understanding Machine Learning: From Theory to Algorithms, Cambridge University Press.

ANALYSIS AND DESIGN OF ALGORITHMS LAB			
Course Code:	**MAI281**	**Course Credits:**	**2**
Course Category:	**CC-P**	**Course (U / P)**	**P**
Course Year (U / P):	**2P**	**Course Semester (U / P):**	**3P**
No. of Labs (Hrs/Week):	**3**	**Mid Sem. Exam Hours:**	**-**
Total No. of Labs:	**10**	**End Sem. Exam Hours:**	**3**

COURSE OBJECTIVES
1. Write sorting programs using Divide-and-Conquer techniques.
2. Implement to find the minimum cost spanning tree and shortest path using different Greedy techniques
3. Construct DFS, BFS programs and topological ordering using Decrease-and-Conquer technique
4. Implement knapsack, travelling salesperson
5. Design different searching & sorting techniques and finding the complexities.

COURSE OUTCOMES
At the end of the course the students should be able to:
1. Demonstrate Quick sort and Merge sort and calculate the time required to sort the elements.
2. Implement the topological ordering of vertices, travelling salesman problem and Knapsack problem
3. Construct programs to check graph is connected or not using BFS and DFS methods
4. Implement programs on divide and conquer, decrease and conquer
5. Experiment finding the minimum cost of spanning tree using Prim's algorithms and shortest path using Dijkstra' algorithm

PRACTICALS

(Note: Use any programming tools like C/Java/Python to execute.)
1. Sort a given set of elements using the Quick sort method and also analyze it's runtime complexity for different inputs.
2.Sort a given set of elements using merge sort method and also analyze it's runtime complexity for different inputs.
3. Write a program to obtain the topological ordering of vertices in a given digraph.
4. Implement travelling salesman problem and knapsack problem (0/1).

5. Print all the nodes reachable from a given starting node in a digraph using BFS method.
6. Check whether a given graph is connected or not using DFS method.
7. Write a program to implement binary search using divide and conquer technique
8. Write a program to implement insertion sort using decrease and conquer technique
9. Find minimum cost spanning tree of a given undirected path using a Prim's algorithm.
10. From a given vertex in a weighted connected graph, find shortest paths to other vertices using Dijkstra's algorithm.

MACHINE LEARNING USING PYTHON LAB			
Course Code	**MAI283**	**Course Credit**	**02**
Course Category	**CC**	**Course(U/P)**	**P**
Course Year(U/P)	**2P**	**Course Semester(U/P):**	**3P**
No of Lectures + Tutorials(Hrs./Week)	**03**	**Mid Semester Exam Hours:**	**-**
Total no of Lectures(L+T)	**10**	**End Term Exam Hours:**	**03**

COURSE OBJECTIVES
1 To introduce students to the basic concepts and techniques of Machine Learning
2 A general understanding of ML process models.
3 To introduce students to the basic concepts and techniques of Machine Learning
4 Understanding of Python Programming and its module
5 Understanding of Deep learning.
Course Outcomes
At the end of the course the student should be able to understand the :
1 Installation of python and its module &ipython notebook.
2 Ability to apply Python principles and techniques.
3 Ability to design ML models and test and train data set .
4 To Understand working of tensorflow.
5 Ability to perform deep learning algorithms.

LIST OF PRACTICALS

1. Installation of Python and python Notebook.
2. Implement- Data Types and Containers in Python.
3. A scatter plot is a diagram where each value in the data set is represented by a dot.
4. Implement Regression to find the relationship between variables.
5. Machine Learning - Train/Test- Evaluate Your Model
6. Implement polynomial regression - R-squared, Predict the future and Bad fit.
7. Implement - Machine Learning - Decision Tree.
8. Install - Python MySQL,MySQLDatabase,Install MySQL Driver,Test MySQL Connector,Create Connection
9. Introduction to Deep Learning - Deep Learning basics with Python, TensorFlow and Keras p.1
10. Optimizing Models with TensorBoard - Deep Learning basics with Python, TensorFlow.

SEMESTER IV

S.No.	Course Code	Course Name	L	T	P	Credits	Types
1	MAI390	Seminar	0	0	3	2	S / E
2	MAI392	Major Project	0	0	20	10	MP2 / E
3	MAI394	Intenship	0	0	32	16	I / E
4	GP	General Proficiency	Non Credit				
		Total Hours and Credits	**0**	**0**	**55**	**28**	

GRAND TOTAL OF CREDITS = 112