Deccan Education Society's

FERGUSSON COLLEGE (AUTONOMOUS), PUNE

Syllabus

For

M. Sc. (Industrial Mathematics with Computer Applications) Part III

(Semester-V and Semester-VI)

[Pattern 2019]

From Academic Year

2021-22

Program Structure of M.Sc. (Industrial Mathematics with Computer Applications) Part-III

Particulars	Paper	Paper code	Title of Paper	Type of Paper	No. of Credits	CE Marks	ESE Marks	Total Marks
M.Sc. Semester V	Paper -1	MTS 6501	Numerical Analysis	D Elect-1	4	50	50	100
		MTS 6502	Optimization Techniques	D Elect -2	4	50	50	100
		MTS 6503	Simulation	D Elect -3	4	50	50	100
	Paper -2 And Paper - 3 And Paper - 4	MTS 6504	Compiler Construction	D Elect -4	4	50	50	100
		MTS 6505	Data mining	D Elect -5	4	50	50	100
		MTS 6506	Introduction to UML and Design patterns	D Elect -6	4	50	50	100
		MTS 6507	Mobile Application Development	D Elect -7	4	50	50	100
		MTS 6508	Soft Computing	D Elect -8	4	50	50	100

	Paper -5	MTS 6509	Experiential Training course on Project Implementation	Project - 1	4	50	50	100
Note: Students need to opt any ONE from MTS6501 to MTS6503 and any THREE from MTS6504 to MTS6508. MTS6509 is compulsory course.								
M.Sc. Semester VI	Paper -1	MTS 6601	Industrial Training	Industrial Training	8	50	50	100

Title of the Course and Course Code	Numerical Analysis MTS6501	Number of Credits :4
Course Outcomes (COs) **On completion of the course, the students will be able to:**		
CO1	State and apply different methods of numerical integration, Numerical Differentiation and Numerical Optimization.	
CO2	Explain the basic principles and theory of Interpolation.	
CO3	Implement all standard curve fitting techniques.	
CO4	Explain basic methods of solving Linear and Non-Linear Equations and Linear systems.	
CO5	Test different methods of solving differential equations and Compute and evaluate differential equations numerically.	
CO6	Develop knowledge of basic concepts and principles related to Mean Value Theorems, Error Term in Taylor Series, Big O notation.	

Unit No.	Title of Unit and Contents
I	**Review of Calculus, Error Analysis** Mean Value Theorems, Error Term in Taylor Series, Big O notation
II	**Solution to Linear and Non-Linear Equations** Fixed point iterative method, bracketing method of locating roots, Initial approximation and convergence criteria, Newton Raphson and Secant method, Muller's method
III	**Solutions to Linear systems AX=B** Triangular Factorization, Iterative methods to Linear Systems (Jacobi and Gauss Seidel Methods), Iteration for Non-Linear System: Newton's Method for Non-Linear System
IV	**Interpolation** Introduction to Interpolation, Lagrange's Approximation (Interpolating Formulas), Newton's Polynomials (Divided Difference, Forward and Backward Approximation)
V	**Curve Fitting** Least Square line and its related problems, Curve Fitting and Non-Linear Leasr Squares, Interpolation by splines.
VI	**Numerical Differentiation and Integration** Approximating the derivative by Numerical Differentiation Formulas, Introduction to Quadrature Formulas, Analysis of Simpsons and Trapezoidal Rule

VII	**Numerical Optimization** Minimization of function (Nedler-Mead Method)
VIII	**Solution to Differential Equations** Introduction to differential equations, Euler's method and its analysis, Heun's Method (Modified Euler's Method), Taylor Series Method, Runge Kutta Method

Reference Books:

1. Numerical Analysis using Matlab: John Mathews and Kurtis Fink, Prentice Hall
2. Numerical Analysis: K.E. Atkinson
3. Numerical Analysis: S.S.Sastry.

Title of the Course and Course Code	**Optimization Techniques** **MTS6502**	**Number of Credits :4**
Course Outcomes (COs) **On completion of the course, the students will be able to:**		
CO1	Identify and state basic concepts in Linear, Non-linear programming and Game theory.	
CO2	Interpret the Game as a Linear Programming problem and discuss methods to solve them.	
CO3	Apply methods to solve Integer programming problems and examine the solutions	
CO4	Analyse the primal-dual relationship of a Linear programming problem and compute the dual.	
CO5	Determine local solutions to develop techniques and solve non-linear programming problems.	
CO6	Formulate and solve a Linear Programming problem using Simplex method.	

Unit No.	Title of Unit and Contents
I	**Introduction to Linear Programming** Prototype Example, The Linear Programming Model, Assumptions of Linear Programming, Additional Examples, Case Studies
II	**Solving Linear Programming Problem: Simplex Method** The Essence of Simplex Method, Setting up the Simplex Method, Algebra of Simplex method, Simplex Method in Tabular Form, Tie Breaking in Simplex Method, Adapting to Other forms, Post Optimality Analysis, Conclusions, Case Studies

III	**Duality and Sensitivity Analysis** The Essence of Duality Theory, Economic Interpretation of Duality, Primal Dual Relationships, Adapting to Other Primal Forms, The Role of Duality in Sensitivity Analysis, The Essence of Sensitivity Analysis, Applying Sensitivity Analysis, Conclusions, Case Studies
IV	**Integer Programming** Prototype Example, Some BIP Applications, Innovative use of Binary Variables in Model Formulation, Some Formulation Examples, Some Perspectives of solving Integer Programming Problems, The Branch and Bound Technique and its applications to Integer Programming, A Branch and Bound Technique for Mixed Integer Programming, Other Developments in solving BIP Problems, Conclusions, Case Studies
V	**Non Linear Programming** Sample Applications, Graphical Illustration of Non Linear Programming Problems, Types of Non Linear Programming Problems, One Variable unconstrained Optimization, Multivariable unconstrained Optimization, The Karush Kuhn Tucker conditions for constrained Optimization, Quadratic Programming, Separable Programming, Convex Programming, Non Convex Programming, Conclusions, Case Studies
VI	**Game Theory** The Formulation of Two Person Zero Sum Games, Solving Simple Games- Prototype Example, Games with Mixed Strategies, Graphical Solution Procedure, Solving by Linear Programming, Extensions , Conclusion

References:

1. Introduction to Operational Research, Frederick Hiller & Gerald Lieberman, McGrawHill
2. Algorithms for Optimization, Mykel J Kochenderfer and Tim Wheeler, MIT Press

Title of the Course and Course Code	**Simulation** **MTS6503**	**Number of Credits :4**
Course Outcomes (COs) **On completion of the course, the students will be able to:**		
CO1	Outline key concepts in Simulations to build Conceptual Models.	
CO2	Discuss various methods for Random Variate Generation.	
CO3	Implement the Monte Carlo Simulation method and variance reduction techniques to solve problems.	
CO4	Analyse simulation models for Single server Queuing systems.	
CO5	Evaluate and develop methods required for Statistical analysis of Simulated data.	

CO6	Design methods to simulate Random Variables and Stochastic Processes

Unit No.	Title of Unit and Contents
I	**Building Conceptual Models** What is a Conceptual Model, Elements of a Conceptual Model, Single Server Queuing System, State Diagrams, Actual time versus Simulated time
II	**Simulating Random Variables and Stochastic Processes** Probability, Probability as a Sample Mean, Revision of concepts of probability mass functions, Cumulative distribution functions, Probability Density Functions, Histograms, Binomial, Poison and Normal Random Variables, Stochastic Processes, Dynamic System Evolution, Simulating Discrete and Continuous time Markov Chains
III	**Simulating the Single Server Queuing Systems** Simulation model, Collecting Simulated Data, Performance Laws, Independent Simulation Runs, Transient and Steady Phases
IV	**Statistical Analysis of Simulated Data** Populations and Samples, Probability distribution of Sample Means, Confidence Intervals, Comparing Two System Designs
V	**The Monte Carlo Method** Estimating the value of pi, Numerical Integration, Estimating probability, Variance Reduction Techniques
VI	**Random Variate Generation** The Inversion method, The Rejection Method, The Composition Method, The Convolution Method, Specialised Methods
VII	**Random Number Generation** Psuedo Random Numbers, Characteristics of a Good Generator, Number Theory Revision, The Linear Congruential Method, The Multiplicative Congruential Method, Linear Feedback Shift Registers, Statistical Testing of Random Number Generators

Title of the Course and Course Code	**Compiler Construction** **MTS6504**	**Number of Credits :4**
Course Outcomes (COs) **On completion of the course, the students will be able to:**		
CO1	Describe compiler, aspects of compilation, structure and phases of compiler, One pass and Multi-pass compilers, cross compiler. Outline Bootstrapping.	
CO2	Interpolate Applications of Regular Expressions and Finite Automata, Recognition of tokens, LEX: A Lexical analyzer generator. Explain	

	Compilation of expression and three address code.
CO3	Implement the Top-Down Parser, Recursive Descent Parsing, Predictive [LL (1)] Parser, Operator Precedence Parser , Shift Reduce Parser, LR Parser concepts, Syntax Directed Definitions and syntax trees.
CO4	Compare SDD and SDT.
CO5	Discriminate Triples and quadruples, expression trees.
CO6	Hypothesize issues in Design of Code Generator

Unit No.	Title of Unit and Contents
I	**Introduction** Definition of Compiler, Aspects of compilation, The structure of Compiler, Phases of Compiler, Error Handling, Introduction to one pass & Multipass compilers, cross compiler, Bootstrapping.
II	**Lexical Analysis (Scanner)** Review of Finite automata as a lexical analyzer, Applications of Regular Expressions and Finite Automata (lexical analyzer, searching using RE), Input buffering, Recognition of tokens LEX: A Lexical analyzer generator (Simple Lex Program)
III	**Syntax Analysis (Parser)** Definition, Types of Parsers **Top-Down Parser:** Top-Down Parsing with Backtracking: Method & Problems, Drawbacks Elimination of Left Recursion (direct & indirect), Need for Left Factoring & examples Recursive Descent Parsing: Definition Implementation of Recursive Descent Parser Using Recursive Procedures **Predictive [LL (1)] Parser:** Definition, Model, Implementation of Predictive Parser [LL (1)], FIRST & FOLLOW, Construction of LL (1) Parsing Table Parsing of a String using LL (1) Table, Bottom-Up Parsers **Operator Precedence Parser -** Basic Concepts, Operator Precedence Relations Form Associativity & Precedence, Operator Precedence Grammar, Algorithm for LEADING & TRAILING with examples, Algorithm for Operator Precedence Parsing with examples, Precedence Functions Shift Reduce Parser Reduction, Handle, Handle Pruning, Stack Implementation of Shift Reduce Parser (with examples) LR Parser Model Types: SLR (1), Canonical LR, LALR (Method & examples) YACC: program sections, simple YACC program for expression evaluation
IV	**Syntax Directed Definition**

	(Syntax Directed Analysis) Syntax Directed Definitions (SDD) Inherited & Synthesized Attributes, Evaluating an SDD at the nodes of a Parse Tree, Example, Evaluation Orders for SDD's Dependency Graph Ordering, Evaluation of Attributes: S-Attributed Definition, L-Attributed Definition **Application of SDT:** Construction of syntax trees and The Structure of a Type Translation Schemes: Definition, Postfix Translation Scheme
V	**Code Generation and Optimization** **Compilation of expression:** Concepts of operand descriptors and register descriptors with example. Intermediate code for expressions – postfix notations, Triples and quadruples, expression trees. Code Optimization: Optimizing transformations – compile time evaluation, elimination of common sub expressions, dead code elimination, frequency reduction, strength reduction **Three address code:** DAG for Three address code The Value-number method for constructing DAG's Definition of Basic Block, Basic blocks and Flow Graphs Directed acyclic graph (DAG) representation of basic block Issues in Design of Code Generator

Reference:

1. Compilers: Principles, Techniques, and Tools, Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman
2. Principles of Compiler Design By: Alfred V. Aho, Jeffrey D. Ullman (Narosa Publication House)
3. LEX & YACC (O'reilly Publication)
4. *System Software: An Introduction to Systems Programming*, Leland L Bech, Pearson Education Asia, 1997.
5. *Compiler Construction: Principles and Practice*, Kenneth C. Louden, Thompson Learning, 2003.
6. *Introduction to Compiler Techniques*, J.P. Bennet, Second Edition, Tata McGrawHill, 2003.
7. *"Engineering a Compiler"*, Keith D Cooper and Linda Torczon, Morgan Kaufmann Publishers Elsevier Science, 2004.

Title of the Course and Course Code	**Data Mining** **MTS6505**	**Number of Credits :4**

Course Outcomes (COs) On completion of the course, the students will be able to:	
CO1	Define Data Mining and its functionalities, Machine learning, DBMS, Statistics, Data warehouses, terminologies used for Attribute oriented analysis.
CO2	Compare operational database systems and data warehouses. Explain Data warehouse Characteristics, Architecture and its Components. Articulate Advanced Techniques, Data Mining software and applications
CO3	Apply Data Cleaning and Data Transformation concepts.
CO4	Analyse major issues in Data Mining. Classify Interestingness measures, implement Visualization Techniques. Categorize types of Attributes. Explain clustering concepts.
CO5	Compare Relevance Measures, Statistical Measures. Compare and Interpret Data mining algorithms: Associative rules, classification, prediction. Discriminate different schemas and Measures.
CO6	Hypothesize the algorithms, characterization.

Unit No.	Title of Unit and Contents
I	**Introduction to Data Mining** Introduction to Data Mining, Data Mining functionalities, Related technologies: Machine Learning, DBMS, Statistics, Classification of Data Mining Systems Data mining architecture, Major Issues in Data Mining, Applications of Data Mining
II	**Data Warehouse and OLAP** Data warehouse: Introduction to Data warehouse, Difference between operational database systems and data warehouses, Data warehouse Characteristics, Data warehouse Architecture and its Components, Extraction - Transformation – Loading, Data Modelling, Schema Design, Star and Snow - Flake Schema, Fact Consultation, Fact Table, Fully Addictive, Semi - Addictive, Non-Addictive Measures, Multidimensional databases**,** OLAP Cube, OLAP Operations, MDX
III	**Overview of Data Pre-Processing** Data Cleaning: Concept, Handling Missing Values, Data Smoothing Techniques (Binning, Outlier Analysis)

	Data Transformation, Data Reduction: Concept, Types, Principal Component Analysis, Attribute Subset Selection, Sampling, Discretization and Generating concept hierarchies
IV	**Data mining Knowledge Representation** Task relevant data Interestingness measures: Types of Measures - Objective Measures (Support, Confidence, Lift) Visualization Techniques: Pie Chart, Bar Chart, Data Cubes, Histogram
V	**Attribute-Oriented Analysis** Terminologies: Data Set, Data object, Data Tuple, Attribute/Dimension Types of Attributes Concept Description Vs. OLAP Data Generalization: Attribute Removal, Attribute Generalization (AOI) Presentation of Generalized Result Attribute Relevance Analysis Relevance Measures (Information Gain, Gain Ratio, Gini Index, Chi^2 Contingency Table) Analytical Characterization Analytical Discrimination (Class Comparison) Statistical Measures (Central Tendency, Dispersion of Data: Range, Quartiles, Inter-quartile Range, Five Number Summary, Variance, Standard Deviation)
VI	**Data mining algorithms: Association rules** Terminologies, Association Analysis, Example of Association Rules, Applications of Association Rules, Mining Association Rules, Evaluation of Association Patterns, Correlation Analysis, Apriori Algorithm, FP growth Algorithm, Sampling Algorithms and Types
VII	**Data mining algorithms: Classification** Terminologies, Supervised Vs. Unsupervised Learning, Model construction, Model Usage, Types of Classification Algorithms (Naive Bayes' Classifier, K-Nearest Neighbour) Applications of Classification Algorithms Decision Tree Induction (Information Gain, Gain Ratio, Gini Index) Measuring performance of classifiers (Precision, |Recall, F-Measure, Confusion matrix) Cross-Validation, Bootstrap Sampling, Inferring Rudimentary Rules: 1R algorithm

VIII	**Data mining algorithms: Prediction** The prediction tasks, Statistical (Bayesian) classification, Conditional Probability, Naive Bayes' Algorithm, Bayesian Interpretation, Bayesian Networks, Instance-Based methods (K- Nearest neighbour)
IX	**Clustering** Concept, Applications of Clustering, Types of Clustering Algorithm Partitioning methods: K-means, Expectation Maximization (EM) Hierarchical methods: Agglomerative, Divisive
X	**Advanced Techniques, Data Mining software and applications** Overview of Text Mining (Tokenization, Stemming, Lemmatization, Remove Stop Words, Document Term Matrix) using R Bayesian approach to classifying text Overview of Web Mining,Data Mining software and applications

Text Book:

1. Alex Berson and Stephen, J. Smith, "Data Warehousing, Data Mining & OLAP", Tata McGraw Hill Edition, Tenth Reprint 2007.
2. Jiawei Hanand Micheline Kamber, "Data Mining Concepts and Techniques", Second Edition, Elsevier, 2007.

References:

1. Pang Ning Tan, Michael Steinbach and Vipin Kumar, "Introduction to Data Mining", PersonEducation, 2007.
2. K. P. Soman, Shyam Diwakar and V. Ajay "Insight into Data Mining Theory and Practice," Easter Economy Edition, Prentice Hall of India, 2006.
3. G. K. Gupta, "Introduction to Data Mining with Case Studies," Easter Economy Edition, Prentice Hall of India, 2006.
4. Daniel T. Larose, "DataMining Methods and Models," Wile-Interscience, 2006.
5. Ian H. Witten and Eibe Frank, Data Mining: Practical Machine Learning Tools and Techniques (Second Edition), Morgan Kaufmann, 2005, ISBN: 0-12-088407-0.
6. Arun K Pujari, Data Mining Techniques, 3rd Edition, Universities Press
7. Pualraj Ponnaiah, Data Warehouse Fundamentals, Wiley Student Edition.
8. P. Radha Krishna, Data Mining, Vikaram Pudi, Oxford University Press.

Title of the Course and Course Code	Introduction to UML and Design patterns MTS6506	Number of Credits :4
Course Outcomes (COs) **On completion of the course, the students will be able to:**		
CO1	Describe various UML building blocks, its diagrams.	
CO2	Distinguish between different categories of design patterns.	
CO3	Demonstrate the Conceptual model of UML and SDLC. Apply the suitable design patterns to refine the basic design for given context.	
CO4	Compare the Creational, Structural, Behavioral and Concurrency design patterns.	
CO5	Select appropriate design pattern to design solutions.	
CO6	Develop the conceptual model into various scenarios and applications using UML.	

Unit No.	Title of Unit and Contents
I	**Introduction to Unified Modeling Language** Introduction, History, Versions of UML, Goals of UML, Conceptual Model of UML, Object Oriented Analysis and Design, Role of UML in OO Design
II	**UML Building Blocks** Things Structural (Class, Interface, Collaboration, Use case, Component, Node), Behavioral (Interaction, State Machine), Grouping (Package), Annotational (Note) Relationships (Dependency, Association, Generalization, Realization), Classification of Diagrams (Structure Diagram, Behavior Diagram)
III	**Structure diagrams** Class diagram (Elements: class, interface, feature, constraint, association, generalization, dependency), Object diagram, Package diagram, Composite Structure diagram, Component diagram, Deployment diagram, Profile Diagram
IV	**Behavior diagrams** Use case diagram, Activity diagram, State Machine diagram, Interaction diagram (Sequence, Communication, Timing, Interaction Overview)
V	**Software Design Patterns** Concept, History, Classification of Design Patterns
VI	**Creational Design Patterns** Abstract Factory, Builder, Dependency Injection, Singleton

VII	**Structural Design Patterns** Adapter, Composite, Decorator, Facade, Marker
VIII	**Behavioural Design Patterns** Chain of responsibility, Command, Iterator, Observer, Strategy
IX	**Concurrency Design Patterns** Active Object, Join, Lock, Monitor Object, Thread Pool

Reference:

1. The Unified Modeling Language User Guide (Object Technology Series) 2nd Edition by Grady Booch / James Rumbaugh / Ivar Jacobson Booch / Rumbaugh / Jacobson

2. UML Distilled: A Brief Guide to the Standard Object Modeling Language 3rd Edition by Martin Fowler

3. Head First Design Patterns, 2nd Edition by Eric Freeman, Elisabeth Robson, Released December 2020, Publisher(s): O'Reilly Media, Inc.

4. Design Patterns: Elements of Reusable Object-Oriented Software by Gamma Erich, Helm Richard, Johnson Ralph, Vlissides John ,Grady Booch

Important URLs:

https://www.uml.org

https://en.wikipedia.org/wiki/Software_design_pattern

https://sourcemaking.com/design_patterns

Title of the Course and Course Code	**Mobile Application Development** **MTS6507**	**Number of Credits :4**
Course Outcomes (COs) **On completion of the course, the students will be able to:**		
CO1	Describe core concepts of Flutter Framework and components Dart Language and React Native.	
CO2	Compare the challenges of native and cross-platform mobile development.	
CO3	Use an IDE to write, compile, run, and test applications developed using Flutter framework and React Native.	
CO4	Integrate widgets and state into apps with Firebase as a data store and learn to authenticate a user.	
CO5	Test and deploy the app to emulators, simulators, and real devices.	
CO6	Design native mobile apps for both Android and iOS.	

Unit No.	Title of Unit and Contents
I	**Introduction to Mobile Application Development** What is Mobile Application Development, Various Mobile Platforms (Android, iOS etc…), Development types (Native, Hybrid), Cross platform development tools History (Apache Cordova, Xamarin, Flutter, React Native)
II	**Getting started with Flutter** Introduction, Framework Architecture, Installation, Introduction to DART Language
III	**User Interface in Flutter** Introduction to Widgets, Building Layouts, Adding interactivity, Navigation and Routing, Animations
IV	**Data and Backend in Flutter** State Management, Networking and HTTP, JSON and serialization, Firebase
V	**Deployment in Flutter** Obfuscation Dart code, Build and release Android app, Build and release iOS app
VI	**Getting started with React Native** Introduction, Environment Setup, JavaScript Fundamentals
VII	**Core Concepts in React Native** React Fundamentals (Components, JSX, Props, State), Handling input, Using ScrollView & List Views, Platform specific code
VIII	**Design and Interaction in React Native** Design (Style, Height and Width, Layout with Flexbox, Images) Interaction (Handling touches, Navigating between screens, Animations)
IX	**Deployment in React Native** Steps for Android deployment, Steps for iOS deployment

Reference Books:

1) Beginning App Development with Flutter by Rap Payne, APress
2) Programming Flutter: Native, Cross-Platform Apps the Easy Way (The Pragmatic Programmers) 1st Edition by Carmine Zaccagnino
3) Beginning Flutter: A Hands-On Guide to App Development by Marco L. NapoliReact and React Native: A complete hands-on guide to modern web and mobile development with React.js, 3rd Edition Paperback – April 30, 2020 by Adam Boduch, Roy Derks
4) React Projects: Build 12 real-world applications from scratch using React, React Native, and React 360 Paperback – December 20, 2019
5) by Roy DerksReact Native for Mobile Development: Harness the Power of React Native to Create Stunning iOS and Android Applications Paperback – June 13, 2019

Important URLs:

https://flutter.dev/

https://reactnative.dev/

Title of the Course and Course Code	Soft Computing MTS6508	Number of Credits :4
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	Describe key concepts associated with Soft computing and hard computing.	
CO2	Illustrate various concepts associated with Genetic algorithms and compare Generic Algorithm vs. Traditional Algorithms	
CO3	Apply the concept of Artificial neural network to find linear separability and solve various problems.	
CO4	Explain and Compute Mathematical properties of Fuzzy Logic, Classical Sets and Fuzzy Sets and solve examples to justify the properties.	
CO5	Review the concept of Swarm Computing and Ant colony optimization, formulate the concept of Associative Memory Network and Neuro Fuzzy Modeling to discuss real life problems.	
CO6	Design and analyze the concept of Supervised learning and unsupervised learning to solve real life problems.	

Unit No.	Title of Unit and Contents
I	**Introduction to Soft Computing** What is soft computing? Differences between soft computing and hard computing, Soft Computing constituents, Methods in soft computing, Applications of Soft Computing
II	**Introduction to Fuzzy Logic, Classical Sets and Fuzzy Sets** Operations on classical and fuzzy sets Properties of Classical and fuzzy Sets Classical Relations and Fuzzy Relations Tolerance and Equivalence Relations Membership Functions Features of the Membership Functions Fuzzification Methods of Membership Value Assignments Defuzzification Lambda-Curs for Fuzzy Sets (Alpha-Cuts) Lambda- Cuts for Fuzzy Relations Defuzzification Methods Fuzzy Arithmetic and Fuzzy Measures Extension Principle Fuzzy Measures Fuzzy Integrals Introduction to Fuzzy Rule Base and Approximate Reasoning

	Introduction to Fuzzy Decision Making Introduction to Fuzzy Logic Control Systems
III	**Genetic Algorithms (GA)** What are Genetic Algorithms? Why Generic Algorithms? Biological Background Traditional Optimization and Search Techniques Genetic Algorithm and Search Space Generic Algorithm vs. Traditional Algorithms Simple GA, General Genetic Algorithm, Operators in Generic Algorithm
IV	**Artificial Neural Network:** Fundamental Concept Evolution of Neural Networks Basic Models of Artificial Neural Network Important Terminologies of ANN McCulloch-Pits Neuron Linear Separability Hebb Network
V	**Supervised Learning Network** Introduction Perceptron Nonworks Adaptive Linear Neuron (Adaline) Multiple Adaptive Linear Neurons Back-Propagation Network
VI	**Associative Memory Network** Introduction Training Algorithms for Pattern Association Auto associative Memory Network Heteroassociative Memory Network Bidirectional Associative Memory (BAM)
VII	**Unsupervised Learning Networks** Introduction Fixed Weight Competitive Nets Kohonen Self-Organizing Feature Maps Learning Vector Quantization Counter propagation Networks Adaptive Resonance Theory Network
VII	**Neuro Fuzzy Modeling** Introduction Adaptive Neuro-Fuzzy Inference Systems Architecture - Hybrid Learning Algorithm - Learning Methods that Cross-fertilize ANFIS and RBFN
IX	**Introduction to Swarm Computing and Ant colony optimization**

Textbooks:

1. J. S. R. Jang, C. T. Sun, and E. Mizutani, "Neuro-Fuzzy and Soft Computing", PHI, 2004, Pearson Education.

2. S. N. Sivanandam, S. N. Deepa, Principles of Soft Computing (With CD) Wiley India,

ISBN: 9788126527410

3. S. Rajasekaran and G. A. Vijayalaksmi Pai, Neural Network, Fuzzy Logic and Genetic Algorithms - Synthesis and Applications", (2005), Prentice Hall

References:

1. Timothy J. Ross, Fuzzy Logic: With Engineering Applications Wiley India, Third Edition ISBN: 978-81-265-3126-4
2. Kumar Satish, Neural Networks: A Classroom Approach,1/e TMH, ISBN: 9780070482920
3. David E. Goldberg, Genetic Algorithms in search, Optimization & Machine Learning by, Pearson Education, ISBN: 81-7808-130-X
4. James A. Freeman and David M. Skapura, Neural Networks Algorithms, Applications, and Programming Techniques, Edition: Pearson Edn., 2003.V. Kecman, Learning and Soft Computing, MIT Press, 2001.

Title of the Course and Course Code	**Experiential Training course on Project Implementation MTS6509**	**Number of Credits :8**
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	Define a problem statement relevant for the Project Topic.	
CO2	Outline the prerequisites for the Proposed Project.	
CO3	Implement multiple programming languages, databases and frameworks as a part of their self-learning exercises.	
CO4	Break down complex user requirements into simple use cases using design mythologies / principles / patterns.	
CO5	Determine performance and scalability requirements.	
CO6	Formulate user journeys ,flows to minimal and reusable development and comprehensive testing.	

The objective of this course is:

- To analyze and break down complex user requirements into simple use cases using design mythologies / principles / patterns.
- Formulate user journeys / flows to minimal and reusable development and comprehensive testing.
- Prime objective after this exercise is to enable students to select the best suited technology for implementation.
- Students are also expected to consider performance and scalability requirements.
- Above considerations emphasizes the fact that the students can explore multiple programming languages, databases and frameworks as a part of their self-learning exercises.

Title of the Course and Course Code	**Industrial Training MTS6601**	**Number of Credits :8**
Course Outcomes (COs) On completion of the course, the students will be able to:		
CO1	Describe and develop the various skills, attitude, and knowledge to understand the professionalism in the IT industry.	
CO2	Discuss and explain the working culture of the Industry in view to maintain quality standards.	
CO3	Implement the confidence, presentation skills and logical thinking, communication skills in developing the system.	
CO4	Differentiate between the academics and professional work culture in timely delivery of projects.	
CO5	Compare and contrast the professional development of the programs and project.	
CO6	Combine the techniques to enhance oneself as a thorough software professional.	

The structure for the Industrial Training Project (ITP) will be as follows

A student can complete Industrial Training Project (ITP) in any I.T. industry / academic institute / with a research project of a teacher / an expert funded by any funding agency for a minimum period of three months.

1) **There will be a teacher coordinator mentoring a group of 10 students throughout the Semester. The teacher coordinator is expected to perform the following tasks**

- Maintain a weekly status/progress report of the student. **The student will report to the assigned student coordinator once a week either offline/ online mode**

regarding the progress of his/her work at the Industry/Academic Institute.

- Maintain contact with the Industry/Academic regarding the internship offered to the student.
- Help the student in solving difficulties
- Organize presentations and discussions as required
- Guide the student in preparing the Final Project Documentation
- Maintain a track record for each student through the semester
- Conduct an Internal Assessment for each student consisting of 50 Marks

The work load for the teacher coordinator is proposed as four hours per week.

2) **The workload for a teacher coordinator who is guiding 3 students doing their ITP in Fergusson College (Autonomous) Pune (no mentor from industry) is proposed as four hours per week.**

Guidelines for submitting the Final Project report

The student must include the project completion certificate issued by the respective industry/research institute/educational institute in the report. A student will submit two hard bound copies and one softcopy (pdf format): Student Copy, Department copy, Controller of Examinations copy of the work carried out during ITP (pdf format to be emailed by the respective emails).

3) **Scheme of Assessment**

- **Continuous Internal Assessment**

  Evaluation for internal 50 Marks to be done by the Internal Teacher Coordinator

Description	Marks
Weekly reporting (Minimum 12) (Online or Offline Mode as needed	25 Marks
Final Project report documentation	15 Marks
Presentation Demo	10 Marks

End Semester Assessment

Evaluation for external 50 Marks will be done by a panel of three consisting of One Industrial Expert, One Academic Expert (External from other college) and One Internal Examiner. Each examiner is expected to assess each student for 50 marks independently and average of the three scores is to be considered as the final ESE score (out of 50).

Description	Marks
Knowledge and Execution of the System	15
Final Project Report	15
Presentation	10
Viva Voce	10

The Internal Examiner will submit the total of 100 marks to the Examination Section

The final grade (to be printed on the mark list) is to be calculated on the basis of UGC 10-point scale.

Marks	Grade	Grade Point
90-100	O : Outstanding	10
80-89	A+ : Excellent	9
70-79	A : Very Good	8
65-69	B+ : Good	7
60-64	B: Above Average	6
55-59	C+ : Average	5
50-54	C: Below Average	4
45-49	D: Satisfactory	3
40-44	E: Pass	2
0-39	Fail	0
	Absent	0

Note: - A student who has obtained Grade F will have to carry out this project once again for a complete semester (minimum three months).