

UNIVERSITY SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Syllabus

Master of Technology (Computer Science and Engineering)

For Working Professionals

Batch: 2024-26



**GAUTAM BUDDHA UNIVERSITY
GAUTAM BUDH NAGAR, GREATER NOIDA**

Semester-I



ADVANCED DATABASE MANAGEMENT SYSTEM			
Course Code:	WCS521	Course Credits:	4
Course Category:	CC1/SEC	Course (U / P)	P
Course Year (U / P):	1P	Course Semester (U / P):	1P
No. of Lectures + Tutorials (Hrs/Week):	03 + 01	Mid Sem. Exam Hours:	1.5
Total No. of Lectures (L + T):	45 + 15	End Sem. Exam Hours:	3
COURSE OBJECTIVES			
1 Knowledge of database design			
2 A general understanding of database, design and dependency			
3 Understanding of different types of databases			
4 Knowledge of databases on the internet			
5 Application on enhanced database			
COURSE OUTCOMES			
At the end of the course the students should be able to:			
1 Basic knowledge and understanding of ER diagram and UML class diagram.			
2 Ability to apply functionality and Normalization in relational databases.			
3 Recognize and fetch data from object oriented, parallel and distributed databases.			
4 Use XML and understand unstructured data			
5 Implement concept and deduction of enhanced database on different applications			

UNIT I: INTRODUCTION TO DATABASE DESIGN

Entities, Attributes, Entity Sets, Relationships, Key Constraints, Participation Constraints, Weak Entities, UML Class Diagrams, Subclasses, Super-classes, Inheritance, Specialization, Generalization, Constraints and Characteristics of Specialization and Generalization Hierarchies, Modeling of UNION Types Using Categories, Representing Specialization and Generalization In UML Class Diagrams, Data Abstraction, Knowledge Representation and Ontology Concepts.

UNIT II: DATABASES DESIGN THEORY

Problems Caused by Redundancy, Decompositions, Problems Related to Decomposition, Reasoning About FD's, FIRST, SECOND, THIRD Normal Form, BCNF, Fourth Normal Form, Lossless Join Decomposition, Dependency Preserving Decomposition, Schema Refinement in DataBase Design, Multi Valued Dependencies.

UNIT III: OBJECT- ORIENTED, PARALLEL AND DISTRIBUTED DATABASES

Overview of Object-Oriented Concepts, Object Identity, Object Structure, Type Constructor, Encapsulation of Operations, Methods and Persistence; Architectures for Parallel Databases, Parallel Query Evaluation, Parallelizing Individual Operations, Sorting Joins, Distributed Database Concepts, Data Fragmentation, Replication and Allocation Techniques for Distributed Database Design, Query Processing in Distributed Databases, Concurrency Control and Recovery in Distributed Databases.

UNIT IV: DATABASES ON THE WEB AND SEMI-STRUCTURED DATA

Web interface, XML, structure of XML data, querying XML data, storage of XML data, XML applications, semi- structured data model, indexes for text data.

Handwritten signatures and initials at the bottom of the page.

UNIT V: ENHANCED DATA MODELS FOR ADVANCED APPLICATIONS

Active database concepts, temporal database concepts, spatial databases: concept and architecture, deductive databases and query processing, mobile databases, Geographic Information Systems (GIS).

Textbooks:

1. Elmasri and Navathe, Fundamentals of Database Systems,
2. Ramakrishnan and Gehrke, Database Management Systems,

References Books:

3. Korth, Silberschatz, Sudarshan, Database System Concepts,
4. Rob and Coronel, Database Systems: Design, Implementation and Management,
5. Date and Longman, Introduction to Database Systems.

JAVA PROGRAMMING			
Course Code:	WCS523	Course Credits:	4
Course Category:	CC2/SEC	Course (U / P)	P
Course Year (U / P):	1P	Course Semester (U / P):	1P
No. of Lectures + Tutorials (Hrs/Week):	03 + 01	Mid Sem. Exam Hours:	1.5
Total No. of Lectures (L + T):	45 + 15	End Sem. Exam Hours:	3
COURSE OBJECTIVES			
1 Knowledge of basic Object-Oriented paradigm, practices and application.			
2 A general understanding of class, object and methods.			
3 Understanding of multithreading and applets.			
4 Basic knowledge of swings and Beans with implementation.			
5 Understanding of Servlet programming.			
COURSE OUTCOMES			
At the end of the course the students should be able to:			
1 Basic knowledge and understanding of object-oriented programming.			
2 Ability to apply OOP's concept in real life problems.			
3 Ability to design, develop, maintain and evaluate large-scale software systems.			
4 To produce efficient, reliable, robust and cost-effective software solutions using Java.			
5 Ability to perform independent research and analysis.			

UNIT I OBJECT-ORIENTED PROGRAMMING

Concept of object-oriented programming (OOP), benefits of OOP, application of OOP, Java history, Java features, Data types, Java key words, identifiers, constants, variables, declaration and scope of the variable, symbolic constant, type casting, arrays, strings, vectors, wrappers classes, operator, expressions, 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 II 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, Interface and Packages.

UNIT III MULTITHREADING AND APPLET PROGRAMMING

[Handwritten signatures and initials at the bottom of the page]

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.

UNIT IV SWING AND BEANS

Introduction to Swing, Differences between AWT Controls & Swing Controls, JApplet, Swing Button: JButton, JToggleButton, CheckBoxes, Radio Button, JComboBox, Text Boxes etc., Icons, Labels, JTabbed Pains, JScroll Pains, JList, JTrees, JTables Java Beans: Introduction to Java Beans, Advantages of Java Beans, BDK Introspection, Developing a Home page using Applet & Swing.

UNIT V SERVLET PROGRAMMING

Introduction to Servlets: Lifecycle of a Servlet, The Servlet API, The javax. Servlet Package, Reading Servlet parameters, Reading Initialization parameters; The javax.servlet HTTP package, Handling Http Request & Responses, Security Issues Introduction to JSP, Problem with Servlet. The Anatomy of a JSP Page, JSP Processing, JSP Application Design with MVC Setting Up and JSP Environment: Installing the Java Software Development Kit, Tomcat Server & Testing Tomcat.

REFERENCE BOOKS:

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

The bottom of the page contains several handwritten signatures and initials in blue ink. There are approximately seven distinct marks, including a large stylized signature, a checkmark-like mark, and several sets of initials or smaller signatures.

ADVANCED DATA STRUCTURE AND ALGORITHM			
Course Code:	WCS525	Course Credits:	4
Course Category:	CC3	Course (U / P)	P
Course Year (U / P):	1P	Course Semester (U / P):	1P
No. of Lectures + Tutorials (Hrs/Week):	03 + 01	Mid Sem. Exam Hours:	1.5
Total No. of Lectures (L. + T):	45 + 15	End Sem. Exam Hours:	3
COURSE OBJECTIVES			

- Rel 1. Recognizing the appropriate data structures, ADT libraries, and using it to design algorithms for a specific problem.
2. Be capable of solving problems using abstraction techniques.
 3. Be able to choose appropriate algorithms for a specific problem.
 4. Be able to analyse algorithms in terms of their efficiency and correctness.
 5. To accept the recent developments in the area of algorithm design.

COURSE OUTCOMES

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

1. Design and analyse programming problem statements.
2. Choose appropriate data structures and algorithms for a specific problem.
3. Gain the necessary mathematical abstraction to solve problems.
4. Come up with an analysis of efficiency and proof of correctness.
5. Comprehend and select algorithm design approaches in a problem specific manner.

UNIT I-INTRODUCTION

Review of Basic Concepts: Abstract data types, Data structures, Algorithms, Big-Oh, Small-Oh, Omega, Small-Omega and Theta Notations, finding time complexity of programs. **Recurrence Relations:** Solving Recurrence Relations, Substitution Method, Master Theorem.

UNIT II

Hashing: Review of Hashing, Hash Function, Collision Resolution Techniques in Hashing, Separate Chaining, Open Addressing, Linear Probing, Quadratic Probing, Double Hashing, Rehashing, Extendible Hashing, Recent Trends in Hashing.

UNIT III TREES & GRAPH

Trees: Binary Search Trees, AVL Trees, Red Black Trees, 2-3 Trees, B-Trees, Splay Trees, Minimum Spanning Tree (MST), Kruskal's Algorithm and Prim's Algorithm, Applications to MST.

Graph: Graph, Breadth First Search, Depth First Search, Shortest path in edge-weighted case (Dijkstra's), Bellman Ford Algorithms, Topological Sorting.

UNIT IV SELECTED TOPICS

Strassen's Matrix Multiplication, Greedy method VS Dynamic Programming, Job sequencing with deadlines, Fractional Knapsack Problem, 0/1 Knapsack Problem, Travelling Salesman Problem, Huffman coding, Pre order, Post order, Inorder traversal, Postfix to infix notation, Infix to Postfix notation.

UNIT V

Linear Programming: Geometry of the feasibility region and Simplex algorithm

NP-completeness: Examples, proof of NP-hardness and NP-completeness.

Recent Trends: Recent Trends in problem solving paradigms using recent searching and sorting techniques by applying recently proposed data structures.

[Handwritten signatures and initials in blue ink]

Text Books:

1. Introduction to Algorithms, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein.
2. Algorithms Unlocked: Thomas H. Cormen.
3. The Algorithm Design Manual, Steven S. Skiena.

References Books:

1. Algorithms: Robert Sedgewick and Kevin Wayne.
2. Advanced Data Structures: Peter Brass.



RESEARCH TECHNIQUES IN ICT			
Course Code:	WCS527	Course Credits:	4
Course Category:	CC4	Course (U / P)	P
Course Year (U / P):	1P	Course Semester (U / P):	1P
No. of Lectures + Tutorials (Hrs./Week):	03 + 01	Mid Sem. Exam Hours:	1.5
Total No. of Lectures (L + T):	45 + 15	End Sem. Exam Hours:	3

COURSE OBJECTIVES

1. To know what research is.
2. To make the students aware and apply ICT in Research.
3. To make the students recognize Scientific Research Methodology.
4. To equip the students to examine the philosophical and socio-cultural context of research and relate it to the contemporary paradigm shift.
5. To explore different traditions such as empiricism, rationalism, and constructivism etc.

COURSE OUTCOMES

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

1. Discuss issues of research and knowledge creation
2. Apply appropriate research methods to research problems
3. Critique research methodologies
4. Able to write research papers so as to develop a great career in the field of research.
5. Design a research program.

UNIT I INTRODUCTION TO RESEARCH TECHNIQUES

Meaning of research, objectives of research, motivation in research, types of research, characteristics and prerequisites of research, significance of research, research process, sources of research problem, criteria of identifying the problem, necessity of defining the problem, errors in selecting research problem, technique involved in defining the problem, report, and paper writing.

UNIT II DATA ANALYSIS AND STATISTICAL TECHNIQUES

Data and their analyses, quantitative methods and techniques, Measure of central tendency, measures of variation, frequency distribution, analysis of variance, methods, Correlation analysis, regression analysis, time series and forecasting, introduction to discriminant analysis, factor analysis, cluster analysis, conjoint analysis, probability distribution, binomial distribution, poisson distribution, uniform distribution, exponential distribution, and normal distribution, sampling methods, test of hypothesis.

UNIT III MATHEMATICAL MODELING

Steps of modelling, operations research models like queuing theory, stochastic processes, application of models, conceptual framework development and validation techniques, optimization techniques.

UNIT IV ALGORITHMIC RESEARCH

Algorithmic research problems, types of algorithmic research, types of solution procedure, steps of algorithm development, steps of algorithmic research, design of experiments.

UNIT V SIMULATION AND SOFT COMPUTING TECHNIQUES

Introduction to soft computing, artificial neural network, genetic algorithm, fuzzy logic and their applications, tools of soft computing, need for simulation, types of simulation, simulation language, fitting the problem to simulation study, simulation models, output analysis, data simulation packages like MATLAB, NS2, ANSYS, Cadence.



Text books:

1. Research Methodology: Methods and Techniques, C.R. Kothari







Reference Books:

1. Research Methodologies, R. Panneerselvam, Prentice Hall, 2007.
2. Research in Education, Best John V. and James V Kahn, Wiley eastern, 2005.
3. Elements of Educational Research, Sukhia, S.P., P.V. Mehrotra, and R.N. Mehrotra, PHI publication, 2003.
4. Methodology of Research Education, K. Setia, IEEE publication, 2004.
5. Research methodology, Methods and Techniques, Kothari, C.R., 2000.



ADVANCED DATABASE MANAGEMENT SYSTEM LAB			
Course Code:	WCS581	Course Credits:	2
Course Category:	CC-LI/SEC	Course (U / P)	P
Course Year (U / P):	1P	Course Semester (U / P):	1P
No. of Lab (Hrs/Week) / Total No. of Lab	03 / 10	End Sem. Exam Hours:	3
COURSE OBJECTIVES			
1. To explore the features of a Database Management Systems			
2. To interface a database with front end tools			
3. To comprehend the internals of a database system			
4. To provide a strong foundation in advanced database concepts from an industry perspective.			
5. To learn query processing and transaction management concepts for object-relational database and distributed database			
COURSE OUTCOME			
At the end of the course the students should be able to:			
1. Develop and apply critical thinking skills.			
2. Design and present Lab as well as project reports			
3. Built appropriate methods for the analysis of raw data			
4. Perform logical troubleshooting as and when required.			
5. Verify and implement the concepts and theory learnt in class.			

1. Introduction to MySQL, Postgre Sql, Microsoft Sql softwares.
2. An exercise of data types in PostGresql & Data Definition Language Commands
3. Exercise on Data Manipulation Language and Transaction Control Commands using PostgreSql.
4. Exercise on Types of Data Constraints using PostgreSql.
5. Exercise on JOINS (Single-Table) Using Normalization
6. Exercise on JOINS (Multiple-Table) Using Normalization
7. Exercise on GROUP BY/ORDER BY Clause and Date Arithmetic using PostgreSql.
8. Exercise on different Functions (Aggregate, Math and String)
9. Exercise on different types of sub queries
10. Procedures, View and Triggers.

JAVA PROGRAMMING LAB			
Course Code:	WCS583	Course Credits:	2
Course Category:	CC-L2/SEC	Course (U / P)	P
Course Year (U / P):	1P	Course Semester (U / P):	1P
No. of Lab (Hrs/Week) / Total No. of Lab	03 / 10	End Sem. Exam Hours:	3

COURSE OBJECTIVES

- 1 Knowledge of basic Object-Oriented paradigm, practices, and application.
- 2 A general understanding of class, object, and methods.
- 3 Understanding of multithreading and applet.
- 4 Basic knowledge of Swings and Beans with implementation.
- 5 Understanding of Servlet programming.

COURSE OUTCOMES


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

- 1 Basic knowledge and understanding of object-oriented programming.
 - 2 Ability to apply OOPs concept in real life problem.
 - 3 Ability to design, develop, maintain, and evaluate large-scale software systems.
 - 4 To produce efficient, reliable, robust, and cost-effective software solutions using Java.
 - 5 Ability to perform independent research and analysis.
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, Wrappers Classes, Type Casting, Operators, Decision statement, Loops statement and Branch statements and Exception handling
 3. Write a separate Java Code to implement each of the following OOP's concepts: Abstraction, Encapsulation, Inheritance, Polymorphism, Method Overloading and MethodOverriding
 4. Write a separate Java Code to implement each of the following: Class, Object, Constructors, Method, and Visibility Controls: Private, Public and Protected
 5. Write a separate Java Code to implement each of the following: Final variable, final class, final method, abstract class, abstract method, and concretemethod
 6. Write a separate Java Code to implement each of the following: Interface, extending and implementing interface
 7. Write a separate Java Code to implement each of the following: Multithreading: Create thread with thread class and runnable interface, thread priorities,synchronization
 8. Write a separate Java Code to implement each of the following:Swing and Beans
 9. Write a separate Java Code to implement each of the following:
Swing Button: JButton, JToggleButton, CheckBoxes, Radio Button, JComboBox, Text Boxes
 10. Write a separate Java Code to implement each of the following:Servlet and JSP.

M.Tech (CSE)

Effective from session 2024-25

SEMESTER-2



Python Programming			
Course Code:	WCS 522	Course Credits:	4
Course Category:	CC	Course(U/P)	P
Course Year(U/P):	1P	Course Semester(U/P):	2P
No. of Lectures + Tutorials(Hrs/Week):	03+01	Mid Sem. Exam Hours:	1.5
Total No. of Lectures	45+15	End Sem. Exam Hours:	3
Course Objectives			
1. Provide a smooth introduction to Python programming for students with minimal or no prior programming experience.			
2. Use Google Colab and Jupyter as a collaborative and user-friendly environment for hands-on coding.			
3. Develop practical problem-solving skills with real-world examples and projects.			
4. Introduce foundational programming concepts and progressively build up to data analysis and automation.			
5. Create familiarity with Python's modern libraries and their applications in industries.			
Course Outcomes			
At the end of the course the students should be able to:			
1. Get the idea of how to program in python and its working environment.			
2. Write and execute Python programs.			
3. Understand and apply basic programming constructs like loops, conditionals, and functions.			
4. Visualize data effectively using Matplotlib and Seaborn.			
5. Solve domain-specific problems using Python in automation and scripting.			

UNIT I: Introduction to Python

Access, basic operations, and using Jupyter-style notebooks, Introduction to Python syntax, variables, and basic data types, Arithmetic operations, type conversions, Programming basics: Expressions, statements, and basic I/O operations, Hands-on Lab: Writing your first Python script on Colab and debugging errors.

UNIT II: Control Structures and Iterations

Conditional statements: if, elif, else, Loops: for, while, and nested loops, Control flow: break, continue, pass.

Practical examples: Simple calculators, generating patterns, and small games, Hands-on Lab: Writing programs with real-time examples of conditionals and loops in Colab.

UNIT III: Functions, Strings, and Data Structures

Writing and using functions: Parameters, return values, and scope, String operations: Indexing, slicing, formatting, and string methods, Python data structures: Lists, tuples, sets, and dictionaries, List comprehensions and dictionary manipulations, Hands-on Lab: Case Study

UNIT IV: File Handling, Data Analysis, and Visualization

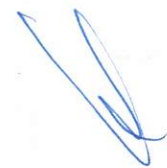
File I/O: Reading and writing files, Introduction to libraries: NumPy for numerical computations, Pandas for data analysis, Data visualization basics with Matplotlib and Seaborn, Hands-on Lab: Analysing and visualizing a sample dataset on Google Colab.

UNIT V: Applied Python

Basics of machine learning: Capstone Project: Build an application using Python to solve a real-world problem, presented via Colab.

Textbooks:

- [1] Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", 2nd edition, Updated for Python 3, Shroff/O'Reilly Publishers, 2016 (<http://greenteapress.com/wp/thinkpython/>)
- [2] Wes McKinney, Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython, O'Reilly Media.
- [3] Aurélien Géron, Hands-On Machine Learning with Scikit-Learn and TensorFlow, O'Reilly Media.
- [4] Eric Matthes, Python Crash Course: A Hands-On, Project-Based Introduction to Programming, No Starch Press.



PROBLEM SOLVING USING ARTIFICIAL INTELLIGENCE			
Course Code:	WCS524	Course Credits:	4
Course Category:	CC	Course (U / P)	P
Course Year (U / P):	1P	Course Semester (U / P):	2P
No. of Lectures + Tutorials(Hrs./Week):	03 + 1	Mid Sem. Exam Hours:	1.5
Total No. of Lectures (L + T):	45 + 15	End Sem. Exam Hours:	3
COURSE OBJECTIVES			
1. To provide a strong foundation of fundamental concepts in Artificial Intelligence			
2. To enable the student to apply these techniques in applications which involve perception, reasoning and learning			
3. To provide a basic exposition to the goals and methods of Artificial Intelligence			
4. Explain the role of agents and how it is related to environment and the way of evaluating it and how agents can act by establishing goals.			
5. Learn the different machine learning techniques to design AI machine and enveloping Applications for real world problems.			
COURSE OUTCOMES			
At the end of the course the students should be able to:			
1. Understand the various searching techniques, constraint satisfaction problem and example problems- game playing techniques.			
2. Apply these techniques in applications which involve perception, reasoning and learning			
3. Acquire the knowledge of real world Knowledge representation			
4. Analyze and design a real world problem for implementation and understand the dynamic behavior of a system.			
5. To enable the student to apply these techniques in applications which involve perception, reasoning and learning			

UNIT I INTRODUCTION TO ARTIFICIAL INTELLIGENCE

What Is AI? The Foundations of Artificial Intelligence. The History of Artificial Intelligence. the State of the Art. Risks and Benefits of AI. Agents and Environments. Good Behavior: The Concept of Rationality. The Nature of Environments. The Structure of Agents

UNIT II PROBLEM SOLVING THROUGH AI

Solving Problems by Searching, Problem-Solving Agents, Example Problems, Search Algorithms, Uninformed Search Strategies, Informed (Heuristic) Search Strategies, Heuristic Functions.

UNIT III Knowledge, reasoning, and planning

Logical Agents, Knowledge-Based Agents, The Wumpus World, Logic, Propositional Logic: A Very Simple Logic, Propositional Theorem Proving, Effective Propositional Model Checking, Agents Based on Propositional Logic, First-Order Logic, Representation Revisited Syntax and Semantics of First-Order Logic, Using First-Order Logic, Knowledge Engineering in First-Order Logic

UNIT IV MACHINE LEARNING

Fundamental Concepts in Machine Learning, Fundamental Concepts in Statistics, Linear Regression, Gradient Descent, Logistic Regression, Naive Bayes. Types of Learning (Supervised, Unsupervised, Reinforcement Learning), Applications.

Unit V DATA-DRIVEN PROBLEM SOLVING

Evaluation Metrics: Accuracy, Precision, Recall, F1 Score, ROC Curve. Data Preprocessing: Data Cleaning, Feature Selection, Dimensionality Reduction (PCA). Deep Learning and Neural Networks: Introduction to Deep Learning, Convolution Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Transfer Learning.

Text Books:

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

Nature Inspired Meta-Heuristics			
Course Code:	WCS534	Course Credits:	3
Course Category:	E1	Course (U / P)	P
Course Year (U / P):	1P	Course Semester (U / P):	2P
No. of Lectures (Hrs/Week) / Total No. of Lectures	03 / 45	End Sem. Exam Hours:	3
COURSE OBJECTIVES			
1. Description of computational intelligence techniques and methods, particularly metaheuristic algorithms for optimization problems.			
2. To identify the techniques appropriate to a given problem.			
3. To argue the utility of metaheuristic algorithms in solving real-world problems.			
4. To implement effective metaheuristic algorithms.			
5. To present basic principles of metaheuristic techniques.			
COURSE OUTCOME			
At the end of the course the students should be able to:			
1. Interpret and explain the concepts of Metaheuristics based optimization and its application in a diverse range of applications.			
2. Model single solution and population-based Metaheuristic algorithms to solve a given optimization problem.			
3. Model Metaheuristic algorithm to solve Multi-objective optimization problems.			
4. Explain algorithm and architectures for parallel implementation of Metaheuristics.			
5. Model hybrid Metaheuristic algorithms to solve a given optimization problem.			

UNIT I: INTRODUCTION

Optimization Models, Approximate Algorithms, When to use Metaheuristics, Methods and Application, Representation, Objective Functions, Constraint Handling, Parameter Tuning, and Performance Analysis.

UNIT II: SINGLE-SOLUTION-BASED METAHEURISTICS

Basic Concepts, Fitness Landscape Analysis, Local Search, Tabu Search, Iterated Local Search, Hill climbing, Simulated Annealing, Variable Neighborhood Search.

UNIT III: POPULATION-BASED METAHEURISTICS METHODS

Basic Concepts, Evolutionary Algorithms: ES, EP, GA, GP, DE, Swarm Intelligence: PSO, ACO, ABC etc.

UNIT IV: METAHEURISTICS**FOR MULTIOBJECTIVE OPTIMIZATION**

Basic Concepts, Multiobjective Continuous and Combinatorial problems, Multi-objective genetic algorithm (MOGA): Non-dominated sorting, crowding distance, elitist model, NSGA and NSGA-II.

UNIT V: HYBRID METAHEURISTICS and its applications

Design and Implementation Issues, Mathematical Programming Approaches, Classical Hybrid Approaches, Hybrid Metaheuristics with Machine Learning and Data Mining, Hybrid Metaheuristics for Multiobjective Optimization.

Text Books:

1. Metaheuristics: From Design to Implementation by El-Ghazali Talhi, Wiley, June 2009.
2. Sean Luke, 2013, Essentials of Metaheuristics, Lulu, Second Edition.

Approximation Techniques			
Course Code:	WCS540	Course Credits:	3
Course Category:	E2	Course (U / P)	P
Course Year (U / P):	1P	Course Semester (U / P):	2P
No. of Lectures (Hrs/Week):	03	Mid Sem. Exam Hours:	1.5
Total No. of Lectures (L):	45	End Sem. Exam Hours:	3
COURSE OBJECTIVES			
The objective of this course is to introduce students to various approximation techniques used in computer science and engineering. The course will cover fundamental concepts, methods, and applications of approximation techniques to solve complex computational problems efficiently.			
COURSE OUTCOMES			
At the end of the course the students should be able to understand:			
1. Understand the principles and importance of approximation techniques in computer science.			
2. Apply various approximation methods to solve computational problems.			
3. Analyze and evaluate the performance of different approximation algorithms.			
4. Develop and implement approximation algorithms for practical applications to real-world problems in computer science.			
5. Explore advanced topics and recent developments in approximation techniques.			

Unit 1: Introduction to Approximation Techniques

Definition, Importance and Need of Approximation in Computer Science, Approximation Algorithms: Concepts and Terminologies, Types of Approximation Methods, Overview of Optimization Problems, Approximation vs. Exact Solutions

Introduction to P, NP, NP-Hard and NP-Complete; Deterministic, non-Deterministic Polynomial time algorithms; Examples of Intractable Problems (NP-hard and NP-complete), Performance Guarantees: Approximation Ratio, Additive and Multiplicative Errors, Complexity Classes and Approximation.

Unit 2: Greedy Algorithms and Local Search

Basic Concepts, Greedy Choice Property, Greedy Approximation Algorithms- Minimum Spanning Tree (Prim's and Kruskal's Algorithm), Problems- Traveling Salesman Problem, Set Covering and Knapsack Problem

Local Search Techniques- Hill Climbing and Simulated Annealing, and Vertex Cover Approximation using Local Search.

Unit 3: Dynamic Programming and Linear Programming

Fundamentals of Dynamic Programming, Examples: Longest Common Subsequence, Matrix Chain Multiplication, Approximation via Dynamic Programming, Applications: Edit Distance, Performance Analysis

Basics of Linear Programming (LP), Applications: Weighted Vertex Cover and Facility Location Problem, Integrality Gap and its Role in Approximation

Unit 4: Probabilistic and Randomized Techniques

Basics of Probabilistic Analysis, Randomized Rounding: Scheduling, Routing and Resource Allocation, Markov

Chains and Monte Carlo Methods, De-randomization Techniques

Unit 5: Advanced Topics and Applications

Approximation in Graph Problems- Steiner Tree Problem, Approximation in Machine Learning: Gradient Descent for Optimization and Boosting and Bagging Techniques, Real-World Applications: Data Clustering and Network Design Problems

Textbooks

1. *V. Vazirani, Approximation Algorithms*, Springer, 2003.
2. *David P. Williamson and David B. Shmoys, The Design of Approximation Algorithms*, Cambridge University Press, 2011.

Reference Books

1. *Cormen, Leiserson, Rivest, and Stein, Introduction to Algorithms*, MIT Press, 3rd Edition, 2009.
2. *Sanjeev Arora and Boaz Barak, Computational Complexity: A Modern Approach*, Cambridge University Press, 2009.
3. *Michael R. Garey and David S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, 1979.
4. *Tim Roughgarden, Beyond the Worst-Case Analysis of Algorithms*, Cambridge University Press, 2021.

h

7.880

bn

A

h

h

PYTHON PROGRAMMING LAB			
Course Code:	WCS582	Course Credits:	2
Course Category:	CC-L3 / SEC	Course (U / P)	P
Course Year (U / P):	1P	Course Semester (U / P):	2P
No. of Lab(Hrs/Week) / Total No. of Lectures	03 /10	End Sem. Exam Hours:	3
COURSE OBJECTIVES			
1. To introduce students to the use of Python programming for industry applications and data analytics.			
2. To develop problem-solving and algorithmic thinking through hands-on Python programming.			
3. To provide exposure to Python libraries such as NumPy, Pandas, and Matplotlib for data manipulation and visualization.			
4. To enable students to automate real-world tasks using Python scripting, web scraping, and API interactions.			
5. To introduce students to Python's role in machine learning, scientific computing, and automation.			
COURSE OUTCOME			
At the end of the course the students should be able to:			
1. Develop structured Python programs to solve industry-oriented problems.			
2. Utilize Python libraries for data analysis, automation, and visualization.			
3. Implement control structures, functions, and data structures effectively.			
4. Process large datasets using NumPy and Pandas for analysis and insights.			
5. Automate repetitive tasks using Python scripts, web scraping, and API integration.			

List of Experiments:

1. Write a program in Python to set up a Google Colab notebook and run basic Python commands. The program should print system details such as Python version, available libraries, and hardware specifications.
2. Write a program to determine the largest and smallest of two numbers and another program to generate and analyze a series of random numbers, classifying them into categories such as even or odd, and positive or negative.
3. Write a program to implement a recursive and non-recursive function to compute the factorial of a number and compare their execution times using Python's time module.
4. Write a program to read a paragraph and count the occurrences of each word, storing the results in a dictionary. The program should also display the most frequent word in the paragraph.
5. Write a program to read a text file and print each word separated by the '#' symbol. The program should also count the occurrences of words and store them in a dictionary.
6. Write a program to create NumPy arrays and perform operations such as addition, multiplication, and element-wise computations. The program should also implement matrix operations, including transpose, determinant, and inverse.
7. Write a program to load a CSV dataset into Pandas and perform basic analysis such as summary statistics, handling missing values, and transforming columns. The program should also group and aggregate data to extract meaningful insights.

8. Write a program to plot time-series data using Matplotlib and generate correlation heatmaps using Seaborn for better visualization and analysis of datasets.
9. Write a program to extract live data from a public website, such as news headlines, using the BeautifulSoup library. The program should also fetch and analyze real-time data from an API, such as weather or stock prices, using Python's requests library.
10. Write a program to develop an industry-relevant application using Python by integrating multiple concepts such as data ingestion, processing, visualization, and automation. This mini-project should demonstrate the application of Python to solve real-world problems in industry or research contexts.

①

h
h
R.B.S.

PA

✓