UNIVERSITY OF KALYANI

UG 4 YEAR Computer Science (HONOURS/ HONOURS WITH RESEARCH) SYLLABUS

(Under NEP 2020)

W.E.F. the Academic Session 2023-24

(Third semester)

COURSE STRUCTURE Computer Science (NEP-2020)

SEMESTER I

Course Code	Course title	Nature of Course	Credit of Course	Class hour/week	Evaluation		Total
					Internal	SemesterEnd	
CS-MJ-T-1	Computer Fundamentals and Programming using C	Major	4	4	15	60	75
CS-MJ-P-1	Programming using C Lab	Major	2	3	Problem: 60, Viva: 10, Lab Notebook: 5		75
CS-MI-T-1	Computer Fundamentals and Programming using C	Minor	3	3	10	25	35
CS-MI-P-1	Programming using C Lab	Minor	1	2	Problem+ Viva:15		15
CS-MU-T-1	Computer Science for Beginners	Multidisciplinary Course	3	3	10	35	45
CS-SEC-P-1	Office Automation Lab	Skill Enhancement Course	3	3	Problem: 35, Viva: 5, Lab Notebook: 5		45
		Value Added Course	4	4	10	40	50
Total			**20**	**22**			**340**

SEMESTER II

Course Code	Course title	Nature of Course	Credit of Course	Class hour/week	Evaluation		Total
					Internal	Semester End	
CS-MJ-T-2	Digital System Design	Major	4	4	15	60	75
CS-MJ-P-2	Digital System Design Lab	Major	2	3	Problem: 60, Viva: 10, Lab Notebook: 5		75
CS-MI-T-2	Database Management Systems	Minor	3	3	10	25	35
CS-MI-P-2	Database Management Systems Lab	Minor	1	2	Problem + Viva =15		15
CS-MU-P-2	Office Automation	Multidisciplinary Course	3	3	10	35	45
AECC-1		Ability Enhancement Course	4	4	10	40	50
CS-SEC-P- 2	Web Development and Applications Lab	Skill Enhancement Course	3	3	Problem: 35, Viva: 5, Lab Notebook: 5		45
Total			**20**				**340**

SEMESTER III							
Course Code	**Course title**	**Nature of Course**	**Credit of Course**	**Class hour/week**	**Evaluation**		**Total**
					Internal	**Semester End**	
CS-MJ-T-3	Computer Organization & Architecture	Major	6	4	15	60	75
CS-MI-T-3	Introduction to Data Structure and Algorithms	Minor- (theory)	3	3	10	25	35
CS-MI-P-3	Introduction to Data Structures and Algorithms using C (Lab)	Minor-(Practical)	1	2	Problem + Viva: 15		15
CS-MU-T-3	AI for Everyone	Multidisciplinary Course	3	3	10	35	45
		Ability Enhancement Course					
CS-SEC-P-3	Data Analysis through Python/R(Lab)	Skill Enhancement Course	3	3	Problem: 35, Viva: 5, Lab Notebook: 5		45
		Value Added Course	4	4	10	40	50
			20	**19**			**265**

Detail Syllabus of Semester-III

CS-MJ-T-3- Theory: Computer Organization & Architecture

Major Course, Theory, Semester – III, Credits - 06, Contact hours - 60

Course description:

This course provides a foundational understanding of digital logic, data representation, computer organization, and the architecture of modern computer systems. It begins with basic concepts like logic gates, Boolean algebra, and sequential circuits. Students will then explore number systems, computer arithmetic, and essential components of computer organization, such as the central processing unit (CPU) and memory. The course also covers input/output organization, memory hierarchy, and key architectural principles like RISC, CISC, and pipelining. By the end of the course, students will gain practical insights into how computers process, store, and retrieve data, enabling them to understand the inner workings of modern computer systems.

Course Outcomes (COs):

1. After completing this course satisfactorily, a student will be able to:

Demonstrate proficiency in number systems, fixed and floating-point representation, and ALU operations.

2. Understand the structure and operation of essential computer components.
3. Understand CPU design and operation
4. Understand memory hierarchy and organization
5. Describe input-output organization

1. **Introduction** 9L

Logic gates, Boolean algebra, combinational circuits, circuit simplification, flip-flops and sequential circuits, decoders, multiplexers, registers, counters and memory units.

2. **Data Representation and Basic Computer Arithmetic** 12L

Number systems, complements, fixed and floating point representation, character representation, addition, subtraction, magnitude comparison, multiplication and division algorithms for integers.

3. **Basic Computer Organization and Design** 10L

Computer registers, bus system, instruction set, timing and control, instruction cycle, memory reference, input-output and interrupt, Interconnection Structures, Bus Interconnection design of basic computer.

4. **Central Processing Unit** 9L

Register organization, arithmetic and logical micro-operations, stack organization, micro programmed control. Instruction formats, addressing modes, instruction codes, machine language, assembly language, input output programming, RISC, CISC architectures, pipelining and parallel architecture.

5. **Memory Organization** 10L

Cache memory, Associative memory, mapping.

6. **Input-Output Organization** 10L

Input / Output: External Devices, I/O Modules, Programmed I/O, Interrupt-Driven I/O, Direct Memory Access, I/O Channels.

Recommended Books:

1. M. Mano, Computer System Architecture, Pearson Education 1992
2. A. J. Dos Reis, Assembly Language and Computer Architecture using C++ and JAVA, Course Technology, 2004
3. W. Stallings, Computer Organization and Architecture Designing for Performance, 8th Edition, Prentice Hall of India,2009
4. M.M. Mano , Digital Design, Pearson Education Asia,2013
5. Carl Hamacher, Computer Organization, Fifth edition, McGrawHill, 2012.

CS-MI-T-3: Introduction to Data Structures and Algorithms

Minor Course, Theory, Semester – III, Credits - 03, Contact hours - 40

Course Objective:

The primary objective of this course is to introduce students to the fundamental concepts of data structures, algorithm design, and analysis. Students will explore different data structures, including arrays, stacks, queues, linked lists, trees, and graphs, and understand their applications. The course also focuses on fundamental searching and sorting algorithms, along with techniques like recursion, divide and conquer, dynamic programming, and greedy algorithms. By the end of the course, students will have a strong foundation in data structure implementation and the analytical tools needed to evaluate algorithm efficiency.

Course Outcome:

1. Understand and classify linear and non-linear data structures.
2. Implement and apply single and multi-dimensional arrays.
3. Use array-based stacks and understand their applications and limitations.
4. Implement queues using arrays in practical scenarios.
5. Develop and apply recursive solutions, understanding their benefits and drawbacks.
6. Perform operations on singly, doubly, and circular linked lists using dynamic representation.
7. Implement binary search tree operations, both iteratively and recursively.
8. Apply and compare linear search, binary search, and basic sorting techniques.
9. Analyze algorithms using time and space complexity and asymptotic notations.
10. Implement divide-and-conquer, dynamic programming, and greedy algorithms.
11. Represent and manipulate graphs using static and dynamic methods.

Syllabus:

1. Basics **2L**

Basic definitions; classifications; ADT; Linear Data Structures - Sequential representations; Non-linear data structures – representations.

2. Arrays **2L**

Single and Multi-dimensional Arrays; Row- major and column-major order; different applications.

3. Stacks **2L**

Array Presentation of Stack and its applications; Limitations of Array representation of stack.

4. Queues **2L**

Array representation of Queue and its applications.

5. **Recursion** **2L**
Developing Recursive Definition of Simple Problems and their implementation; Advantages and Limitations of Recursion.

6. Linked Lists **8L**
Basic Operations(Create, Display, Insert, Delete and Concatenation) on Singly, Doubly and Circular Linked Lists (Dynamic representation only).

6. **Trees and Graphs** **5L**
Introduction to Tree as a data structure; Binary Trees (Insertion, Deletion , Recursive and Iterative Traversals on Binary Search Trees); Different properties of Binary search trees.

7. **Searching and Sorting** **6L**
Linear Search, Binary Search, Comparison of Linear and Binary Search; Bubble Sort, Selection Sort, Insertion Sort, Comparison of Sorting Techniques.

8. **Introduction to Algorithm Analysis** **4L**
Basic Design and Analysis techniques of Algorithms; Models of computation: RAM, TM etc.; time and space complexity; Asymptotic Notations: Big-O, omega, theta etc.

9. **Algorithm Design Techniques** **5L**
Divide and Conquer algorithms for Merge and quick sort, Characteristics of Dynamic Programming (DP) and example of few optimization problems which can be solved by DP. Characteristics of Greedy method and example of few optimization problems which can be solved by Greedy method.

10. **Graph Representation** **2L**
Static and dynamic representation of Graph.

Recommended Books:

1. T.H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein Introduction to Algorithms, PHI, 3rd Edition 2009

2. Sarabasse & A.V. Gelder Computer Algorithm – Introduction to Design and Analysis, Publisher– Pearson 3rd Edition 1999

4. E.Horowitz and Shani ―Fundamentals of Computer algorithms‖

5. A.Aho, J.Hopcroft and J.Ullman ―The Design and Analysis of algorithms

6. Seymour Lipschutz , Data Structures Schaums Series 2/E

CS-MI-T-3: Introduction to Data Structures and Algorithms Using C

Minor Course, Practical, Semester – III, Credits - 01, Contact hours - 30

Course Objective:

The objective of this course is to provide students with a comprehensive understanding of data structures and their implementations through hands-on programming assignments. Students will learn to work with various data structures such as arrays, linked lists, stacks, queues, trees, and binary search trees. The course will emphasize algorithmic thinking, including searching, sorting, and traversal techniques. By the end of the course, students will be able to implement and manipulate these data structures to solve computational problems efficiently.

Sample Programs:

1. Write a program to covert an infix expression into postfix one.
2. Write a program to create, display and concatenate two linear linked lists.
3. Write a program to create, display and concatenate two circular linked lists.
4. Write a program to create, display and concatenate two doubly linked lists.
5. Write a program to create a binary tree and then traverse it in preorder post order and in order.
6. Write a program to search an item from a binary search tree.
7. Write a program to delete a node from a binary search tree.

8. Write a program to print lower and upper triangular matrix.
9. Write a program to represent the operations of stack using array.
10. Write a program to represent the operations of queue using array.
11. Write a program to sort an array using bubble sort.
12. Write a program to sort an array using selection sort.
13. Write a program to sort an array using insertion sort.
14. Write a program to implement insert operation in linear linked list.
15. Write a function to create & display binary search tree.
16. Write a program to implement insert operation in circular linked list.
17. Write a program to find the maximum and minimum values in a binary search tree.
18. Write a program to implement delete operation in linear linked list.
19. Write a program to implement insert operation in circular linked list.
20. Write a program to create a double linked list.
21. A set, say S, of integer data is given. Write a program to sort the set S using tree traversal technique.
22. Write a program to insert a node into a binary search tree.
23. Write a program to sort an array using quick sort.
24. Write a program to sort an array using merge sort.
25. Write a program to implement linear and binary search.

CS-MU-T-3- Theory: AI for Everyone
Multidisciplinary Course, Theory, Semester – III, Credits - 03, Contact hours - 40.

Course Objectives:

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

1. Understand the foundational concepts and evolution of Artificial Intelligence (AI).
2. Explore key AI subfields like machine learning, deep learning, and natural language processing through real-world applications.
3. Identify and assess the impact of AI in sectors such as healthcare, finance, transportation, and education.
4. Analyze the ethical, social, and economic implications of AI, including concerns around bias, privacy, and job displacement.
5. Discuss emerging trends and future possibilities of AI, including its role in creativity, innovation, and responsible use.

Introduction to Artificial Intelligence: Definition and scope of AI; historical overview and key milestones; differentiating AI from human intelligence.

AI Subfields and Technologies: Introduction and basic concepts of machine learning, including supervised, unsupervised, and reinforcement learning, deep learning and neural networks (without technical details); basic concepts of natural language processing (NLP) and computer vision.

Applications of AI: AI in healthcare (diagnosis, treatment, medical imaging); AI in finance (fraud detection, algorithmic trading, risk assessment); AI in transportation (autonomous vehicles, traffic optimization); AI in education (personalized learning, intelligent tutoring systems).

Ethical and Social Implications of AI: Bias and fairness in AI systems; privacy and data protection concerns; impact of AI on employment and the workforce; AI and social inequality.

Emerging Issues and Future Trends: Ethical guidelines and responsible AI practices; AI and innovation;

emerging trends and future directions in AI; AI and creativity (generative models, artistic applications).

Books:

1. S. Goswami, A. K. Das, A. Chakrabarti, “AI for Everyone: A Beginner's Handbook for Artificial Intelligence (AI)”, Pearson, 2024.
2. P. Verdegem, “AI for Everyone?: Critical Perspectives”, University of Westminster Press, 2021.
3. Shawn Schuster, “AI For All: How Everyday People Can Benefit from Artificial Intelligence”, UMLAUT Publishing, 2021.

CS-SEC-P-3- Practical: Data Analysis through Python/R(Lab)

Skill Enhancement Course, Practical, Semester – III, Credits - 03, Contact hours - 40.

Objective:

The objective of the "Data Analysis through Python/R (Lab)" course is to equip students with the skills and knowledge required to analyze, manipulate, and visualize data using Python or R programming. The course focuses on fundamental data analysis techniques, data preprocessing, exploratory data analysis (EDA), statistical modeling, and machine learning. Students will learn how to work with real-world datasets, develop data-driven insights, and apply various algorithms and models to extract meaningful information. The course also emphasizes effective communication of findings through visualizations and reports.

Course Outcomes:

Upon successful completion of this course, students will be able to:

1. Understand and apply fundamental data structures in Python/R for data manipulation.
2. Import, clean, and preprocess datasets from various sources such as CSV, Excel, and databases.
3. Perform exploratory data analysis (EDA) using statistical measures and visualizations to identify patterns and trends.
4. Implement and apply various data transformation techniques such as normalization, standardization, and feature engineering.
5. Conduct statistical tests and understand concepts like correlation and hypothesis testing.
6. Build and evaluate regression and classification models using Python/R for predictive analytics.
7. Apply clustering techniques for unsupervised learning and pattern discovery.
8. Analyze and forecast time series data using appropriate methods such as ARIMA or exponential smoothing.
9. Perform dimensionality reduction using techniques like Principal Component Analysis (PCA).
10. Communicate findings and insights effectively through visualizations and comprehensive reports. Develop and evaluate end-to-end data analysis projects, including data acquisition, analysis, model building, and result presentation.

Sample Programs:

1. Write a Python/R script to create, manipulate, and perform basic operations on lists (Python) or vectors (R). Perform operations such as slicing, indexing, and appending elements.

2. Write a Python/R program to load and manipulate data using dictionaries (Python) or data frames (R).
3. Write a Python/R script to load a CSV file into a Pandas DataFrame (Python) or a data frame (R).
   Perform basic operations like viewing the first few rows, summary statistics, and exporting the modified dataset to a new CSV file. Import data from an Excel file and perform similar operations.

4. Write a Python/R script to identify and handle missing data (drop or impute missing values) in a dataset. Detect and treat outliers using statistical techniques. Perform data normalization or standardization on numerical columns.
5. Perform univariate and multivariate analysis using Python (Pandas, Matplotlib, Seaborn) or R (ggplot2, dplyr).
   Generate summary statistics (mean, median, mode, standard deviation, etc.) for numerical columns.
   Create visualizations such as histograms, boxplots, scatter plots, and pair plots to explore relationships between variables.
6. (a) Write a Python/R script to apply log transformations, binning, or scaling to numerical data.
   (b) Create new features using existing ones (e.g., adding a column for a calculated field).
   (c) Perform one-hot encoding for categorical variables and label encoding for target variables.
7. Write a Python/R script to compute and visualize the correlation matrix for a dataset. Perform hypothesis testing using t-tests or ANOVA to compare means between groups. Apply chi-square tests for independence on categorical data.
8. Create line plots, bar plots, pie charts, and heatmaps using Python (Matplotlib/Seaborn) or R (ggplot2).
   Create advanced visualizations like violin plots, KDE plots, and facet grids to represent multi-dimensional data.
   Customize plots by adding titles, labels, legends, and changing color schemes.

9. Implement a simple linear regression model in Python (Scikit-learn) or R to predict a target variable. Evaluate the model using metrics like R-squared, mean squared error, and visualize the regression line. Extend to multiple linear regression and evaluate its performance.

10. Implement a logistic regression model in Python/R to classify binary data. Evaluate model performance using confusion matrix, accuracy, precision, recall, and F1-score. Explore other classification algorithms like Decision Trees and K-Nearest Neighbors (KNN) for comparison.
11. Implement the K-means clustering algorithm in Python/R to group data based on similarity. Visualize clusters and calculate the silhouette score to evaluate cluster quality. Apply hierarchical clustering and compare the results with K-means.
12. Load and visualize time series data using Python (Pandas) or R. Perform decomposition of time series into trend, seasonality, and residuals. Implement ARIMA or exponential smoothing to forecast future data points.
13. Write a Python/R script to perform PCA on a dataset and reduce its dimensions. Visualize the explained variance and transformed data. Use the reduced dataset to perform further analysis or machine learning.
14. Implement cross-validation techniques to evaluate model performance. Use techniques like GridSearchCV (Python) or tune. grid (R) to find the best hyperparameters for models. Compare multiple models and choose the best one based on performance metrics.