RV UNIVERSITY प्रज्ञा: ★ धीरा: *Go, change the world* *an initiative of RV EDUCATIONAL INSTITUTIONS*	**School of Computer Science & Engineering** **B.Sc(H) Program** **Academic Year: Jan-May 2025** **Semester 4 Version 3.0**

COURSE DESIGN, DELIVERY AND ASSESSMENT

Course Code: CS2050	**Course Name:** Analysis and Design of Algorithms	
Semester: 4	**Area:** Major Computer Science	**SEE Type:** Theory (30 Marks)
Level: 300	**Credits:** 3(2: 0: 2)	**Contact Hours:** 60 Hours
Prerequisite (Course/Skill/Knowledge): ● Fundamentals of Python Programming ● Fundamentals of Data Structures		

Course Faculty:

Sl #	Section	Course Faculty Name	Contact: Email/Contact Number	Sign with Date
1	B & C	Prof. Anik Acharjee	anika@rvu.edu.in	
2	A	Prof.Ashwini Prasad S	ashwinips@rvu.edu.in	

Course Lead (Name, Sign, Date): Prof. Anik Acharjee

1. Course Context & Overview

Algorithm analysis is an important part of computational complexity theory, which provides theoretical estimation for the required resources of an algorithm to solve a specific computational problem. This course provides a comprehensive understanding of algorithmic thinking, emphasizing its definition, significance, and the characteristics of a good algorithm. It covers foundational concepts such as pseudocode, flowcharts, and the systematic steps in algorithm development. Students will delve into analyzing algorithm efficiency using asymptotic notations (Big O, Big Omega, and Big Theta) and explore search algorithms like linear and binary search. The course also focuses on mastering sorting techniques, including bubble sort, selection sort, insertion sort, merge sort, and quick sort, with an emphasis on time complexity and optimization. Advanced design paradigms, such as greedy algorithms (e.g., fractional knapsack, Huffman coding) and dynamic programming, are introduced to tackle complex problems. Essential graph algorithms like minimum spanning trees (Kruskal's and Prim's) and shortest path algorithms (Dijkstra's, Bellman-Ford, and Floyd-Warshall) are also covered, equipping students with critical tools for problem-solving and algorithm design in real-world scenarios.

2. Course Contents

Unit-1 **2 Hours**
Algorithmic Thinking: Definition and Significance
Definition and importance of algorithmic thinking, Characteristics of a good algorithm, Steps in algorithm development, Pseudocode and flowchart.

Unit-2 **3 Hours**
Comprehensive Analysis of Algorithm Efficiency:Asymptotic Notations and Search Algorithms
Analyzing time and space complexity, Big O notation, Big Omega, and Big Theta. Best, worst, and average-case analysis, linear search algorithm, binary search algorithm.

Unit-3 **8 Hours**
Analysis and Optimization of Sorting Techniques
Importance of sorting, Different types of sorting algorithms, Stability and in-place sorting, Step-by-step explanation Analysis of time complexity (best, worst, and average cases), Optimization techniques of the following sorting algorithms - bubble sort , selection sort , insertion sort, merge sort and quick sort algorithms.

Unit-4 **8 Hours**
Advanced Algorithm Design Paradigms: Greedy Algorithms and Dynamic Programming Design paradigms: greedy algorithms - greedy choice, fractional knapsack, Huffman coding. Dynamic programming - optimal substructure and overlapping sub-problems.

Unit-5 **9 Hours**
Essential Graph Algorithms: Minimum Spanning Trees and Shortest Paths
Graph Algorithms: minimum spanning tree algorithms - cut property, Kruskal's algorithm using disjoint sets, Prim's algorithm. Shortest path algorithms - Dijkstra's algorithm, Bellman-Ford algorithm, Floyd-Warshall algorithm.

Course Outcomes: After completing the course, the students will be able to:	
CO1	Describe basic algorithm design strategies and their applications to solve problems.
CO2	Estimate computational complexity of algorithms
CO3	Select appropriate design techniques for real-world problem-solving.
CO4	Implement algorithmic design techniques using greedy method.
CO5	Apply dynamic programming approaches for problem-solving.

Assessment Methodologies

(Tick✔ the Relevant Methodologies)

Assignments✔	Closed book tests✔	Open book tests
Case study	Student Presentation	Mini projects / Model Building
MOOC✔	Quiz✔	Peer Review

Textbooks (With ISBN No)

1. Gilles Brassard, Paul Bratley, Pearson, Fundamentals of Algorithmics, 1st Edition, 2015,ISBN: 9789332549999
2. Steven S. Skiena, The Algorithm Design Manual, ISBN: 978-3030542559, 3030542556, 3rd Edition, Springer, 2020, ISBN: 978-3030542559
3. Fundamentals of Computer Algorithms · Ellis Horowitz;Sartaj Sahni;Sanguthevar Rajasekaran · ISBN: 9788173716126
4. Introduction to the Design and Analysis of Algorithms, Anany Levitin:, 2rd Edition, 2009. Pearson.ISBN:978-9332585485

Reference Material (With ISBN No)

1. Algorithm Efficiency Cases-Course-Swayam-NPTEL
   https://youtu.be/Q0pWMzPAdYM?si=J1AbDi0smb7u37mp
2. Asymptotic Notation-Course-Swayam-NPTEL
   https://youtu.be/msj_2TEzRP0?si=gs4QDzCRAYhQchxW
3. Examples for Calculating Algorithm's Efficiency
   https://youtu.be/xFRea6xyPgM?si=36-DWPDepqyuIYrh
4. DR Stinston, Techniques for Designing and Analyzing Algorithms,2021, ISBN:9780429277412

3. CO Mapping: Cognitive Levels, Knowledge Category, Contact & Activity Hours

SI.No	Course Outcomes	Cognitive Level	Knowledge Category	PO (optional)	Class Conceptual hours	Class Activity Hours	Weightage of CO%
CO1	Describe basic algorithm design strategies and their applications to solve problems.	Understand	Conceptual	PO1,PO3 ,PO5	2	2	10%
CO2	Estimate computational complexity of algorithms.	Analyze	Procedural	PO1,PO3 ,PO5	3	3	15%
CO3	Select appropriate design techniques for real-world problem-solving.	Apply	Procedural	PO1,PO2 ,PO3,PO 4,PO5	5	5	25%
CO4	Implement algorithmic design techniques using greedy method.	Apply	Procedural	PO1,PO3 ,PO4,PO 5	10	10	25%
CO5	Apply dynamic programming approaches for problem-solving.	Apply	Procedural	PO1,PO3 ,PO4,PO 5	10	10	25%
	Total Contact Hours 60 Hours(15 Weeks)				**30**	**30**	**100%**

4. Course Plan

Week 1	Outcome	Topics	Activity Based Teaching Learning Process
Week1	CO1	Algorithmic Thinking: Definition and Significance, Characteristics of a Good Algorithm,Steps in Algorithm Development, Pseudocode, and Flowcharting	Lecture on algorithm definition and importance; Discuss examples of good algorithms; Class activity on identifying algorithm characteristics.
			Hands-on session: Write pseudocode and create flowcharts for simple problems.
Week 2	CO2	Analyzing Time and Space Complexity; Big O, Omega, and Theta Notations. Best, Worst, and Average-Case Analysis. Linear & Binary Searching Algorithms with time and space complexity analysis.	NPTEL Video Learning.Hands-on: Linear & Binary Search Algorithms.
			Quiz Conduction for the NPTEL Videos
Week 3	CO3	Introduction to Sorting: Importance, Characteristics, Stability, and In-place Sorting.	Deliver a lecture explaining the significance of sorting in computer science and introduce various sorting algorithms.
			Conducting hands-on sessions for various sorting algorithms.
Week 4	CO3	Bubble Sort and Selection Sort: Step-by-Step Explanation and Analysis.	Use visual aids and examples to illustrate working of sorting techniques and hands-on sessions.
			Hands-on sessions for different sorting techniques.
Week 5	CO3	Insertion Sort and Merge Sort: Step-by-Step Explanation and Analysis.	Use visual aids and examples to illustrate working of sorting techniques and hands-on sessions.
Week 6	CO3	Quick Sort: Step-by-Step Explanation and Analysis	Use visual aids and examples to illustrate working of sorting techniques and hands-on

			sessions.
Week 7	CO3	Optimization Techniques for sorting algorithms.	Discuss optimization strategies for each sorting algorithm, such as bidirectional selection sort (Cocktail Sort) for selection sort, and how to choose good pivots for quick sort.
			Different examples for Quick Sort.
Week 8	CO5	Greedy Algorithms: Greedy Choice, Fractional Knapsack Problem	Lecture and demonstration of greedy algorithms; Solve the fractional knapsack problem
Week 9	CO5	Huffman Coding: Algorithms and Applications.	Explain the principles of Huffman coding, including building a Huffman tree and assigning variable-length codes to symbols based on their frequencies.
			Conducting different hands-on sessions on Huffman Coding.
Week 10	CO5	Dynamic Programming: Optimal Substructure and Overlapping Subproblems.	Lecture on dynamic programming concepts and solve problems.
Week 11	CO4	Introduction to minimum spanning tree,cut-property.Kruskal's algorithm using disjoint sets, Prim's algorithm.	Use visual aids and explain the steps of Kruskal's algorithm, including sorting edges by weight and using disjoint sets to avoid cycles with hand simulation.Explain the steps of Prim's algorithm, including starting from an arbitrary vertex and growing the minimum spanning tree by adding the minimum-weight edge that connects a vertex in the tree to a vertex not in the tree.
			Different hands-on sessions on Kruskal's Algorithm

Week 12,13,14	CO4	Shortest Path Algorithms: Dijkstra's, Bellman-Ford, and Floyd-Warshall Algorithms	Hands-on session: Implement shortest path algorithms; Discuss their practical applications
Week 15	All Outcomes	Course Review and Problem-Solving Practice	Review of all units

5. Partial Delivery

Sl.No:	Description	Topics Beyond Syllabus/ Industry Visit/ Guest Lectures/ Technical Talks/ Workshops/ NPTEL etc.
1.	Guest Lectures	Dynamic Programming: 0/1 Knapsack Problem, Longest Increasing Subsequence, Importance of Analysis and Design of Algorithms in Industry

6. Instructional Methodologies (Tick the relevant)

Blackboard & chalk ✔	PowerPoint presentations ✔	Student seminars
Mini-Projects	Industry Guest Lectures	Flipped Classroom
Web resources /certification ✔	MOOC	Any other (Specify) Other methods will be adopted as per nature and requirements of the cohort

7. Assessment Methodologies - Indirect (Tick the relevant)

Student Feedback on Course (Exit Survey) ✔	Feedback from Industry Expert
Feedback from Alumni	If any other (Please Specify)

8. Course Outcomes to Program Outcome Mapping

Program Outcomes / Course Outcomes		PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8
CO1	Describe basic algorithm design strategies and their applications to solve problems.	3	-	3	-	2	-	-	-
CO2	Estimate computational complexity of algorithms.	3	-	3	-	2	-	-	-
CO3	Select appropriate design techniques for real-world problem-solving.	3	2	3	2	3	-	-	-
CO4	Implement algorithmic design techniques using greedy method.	3	-	3	2	3	-	-	-
CO5	Apply dynamic programming approaches for problem-solving.	3	-	3	2	3	-	-	-

9. Justification of CO-PO Mapping [For all the CO's]

CO1: Describe basic algorithm design strategies and their applications to solve problems.

PO	Level of Correlation	Justification
PO1	Level 3	Understanding algorithm design strategies is essential for addressing complex problems effectively
PO3	Level 3	Describing algorithm design strategies lays the foundation for analytical and research competence.
PO5	Level 2	Understanding algorithms fosters lifelong learning and enhances digital proficiency.

CO2: Estimate computational complexity of algorithms.

PO	Level of Correlation	Justification
PO1	Level 3	Estimating computational complexity requires critical thinking and problem-solving skills. It is fundamental for understanding the efficiency of algorithms, which directly contributes to complex problem-solving abilities.
PO3	Level 3	This CO builds analytical and research competence by teaching students how to evaluate algorithm performance, a skill emphasized throughout the syllabus. This prepares students for further research and optimization in algorithm design.
PO5	Level 2	Estimating complexity strengthens digital proficiency and lifelong learning by helping students grasp fundamental concepts in algorithm analysis, providing a solid foundation for future learning in software development and data science.

CO3: Select appropriate design techniques for real-world problem-solving.

PO	Level of Correlation	Justification
PO1	Level 3	Selecting suitable algorithmic techniques is a critical skill for solving complex, real-world problems.
PO2	Level 2	Effective communication is moderately involved when justifying and presenting selected techniques in a team or professional setting.
PO3	Level 3	This CO heavily relies on analytical and research skills to identify and implement appropriate solutions.
PO4	Level 2	Collaboration may play a role in solving real-world problems, contributing moderately to teamwork and leadership skills.
PO5	Level 3	Strongly promotes lifelong learning and enhances the ability to apply theoretical knowledge in practical contexts.

CO4: Implement algorithmic design techniques using greedy method.

PO	Level of Correlation	Justification
PO1	Level 3	Implementing greedy algorithms directly addresses complex problem-solving needs.
PO3	Level 3	Applying algorithmic design techniques requires analytical proficiency and technical expertise.
PO4	Level 2	Practical implementation may involve collaboration, contributing moderately to teamwork skills.
PO5	Level 3	Encourages lifelong learning and strengthens technical skills through practical applications.

CO5: Apply dynamic programming approaches for problem-solving.

PO	Level of Correlation	Justification
PO1	Level 3	Applying dynamic programming directly supports solving advanced and complex problems.
PO3	Level 3	This CO focuses on deep analytical thinking to utilize dynamic programming effectively.
PO4	Level 2	Problem-solving using dynamic programming may involve collaborative efforts, enhancing teamwork to some extent.
PO5	Level 3	Strongly reinforces lifelong learning and digital proficiency by promoting the application of advanced techniques.

10. Assessment Plan

Internal Assessment Plan: 70 Marks				
Sl#	Component	Marks	Type of Assessment	Timeline
Continuous Internal Evaluation-1 (20 Marks)				
1	CO1	05	Graded Component 1 (Course Completion)	Week 1
2	CO2	15	Graded Component 2 (Quiz)	Week 3
Continuous Internal Evaluation-2 (25 Marks)				
3	CO1 - C04	25	Mid Sem Examination (Theory)	Week 9
Continuous Internal Evaluation-3 (25 Marks)				
4	CO1 - CO5	25	Practical Component	Week 14,15

Mid Sem Assessment Pattern: 25 Marks		
Sl#	Content	
Part A- 5 Marks (Minimum 1 mark, Max 2 marks, Max 5 Questions)		
1	5 Questions of 1 mark each	5
Part B- 20 Marks (Compulsory 4 questions of 5 marks each, there can be max 3 sub divisions)		
2	4 Questions of 5 marks each with 2 or 3 sub divisions possibly	20

SEE Assessment Pattern: 30 Marks		
Sl#	Content	
Part A - 10 Marks (2 Questions, 5 Marks Each, max 3 Subdivisions)		
1	2 Questions of 5 marks each	10
Part B - 20 Marks (2 Questions, 10 Marks Each, Max Four Subdivisions per Question)		
2	2 Questions of 10 marks each	20

11. Rubrics for Assessment Components

Assessment Component	Type of Component	Rubrics for Assessment
1.	CIE 1	Graded CIE Component 1: : Completion of Course on Algorithms,Flowcharts, Pseudocodes from Coursera which carries 05 Marks.
		Graded CIE Component 2: After thorough watching the NPTEL videos, a quiz will be conducted based on the content, carrying a total of 15 marks from both Unit-1 & 2.
2.	Mid Sem	Test will be conducted for 25 marks and will be evaluated according to the Scheme of Evaluation prepared by the team of course faculty.
3.	Practical Component	Lab record+ Viva = 5 marks Program Write-up & Execution = 20 marks
4.	SEE	Test will be conducted out of 30 marks and will be evaluated according to the Scheme of Evaluation prepared by the team of course faculty.

12. Detailed Rubrics of Assessment Components

Lab Write-up and Execution rubrics (Max: 5 marks)					
Sl. No	**Criteria**	**Measuring Methods**	**Excellent**	**Good**	**Poor**
1	**Understanding of problem statements. (0.5 Marks)**	Observations	Student exhibits thorough understanding of requirements and applies suitable techniques for the problem **(0.5 M)**	Students have sufficient understanding of requirements and apply suitable techniques for the problem. **(0.5 M)**	Student does not have a clear understanding of requirements and is unable to apply any techniques for the problem. **(0 M)**
2	**Execution (2 Marks)**	Observations	Students demonstrate the execution of the program with error correction and show performance efficiency. Appropriate validations with all test cases are handled. **(2 M)**	Students demonstrate the execution of the program without error correction of the code and show performance efficiency with only few test cases. **(1 M)**	Student has not executed the program. **(0 M)**

3	**Format of Results** **(0.5 Marks)**	Observations	Execution Results should show in proper format **(0.5 M)**	Execution Results without format **(0.5 M)**	Execution Results with no comments and no output **(0 M)**
4	**Activities Execution** **(2 Marks)**	Observations	Insightful explanation of appropriate techniques for the given problem to derive a solution. **(2 M)**	Sufficiently explains the use of appropriate techniques for the given problem to derive a solution. **(1 M)**	Unable to explain the techniques for the given problem. **(0 M)**
Viva Voce Rubrics (Max: 5 marks)					
1	**Conceptual Understanding** **(3 Marks)**	Viva Voce	Explains thoroughly the program executed along with related concepts. **(2 M)**	Adequately explains the program executed along with related concepts **(1 M)**	Unable to explain the program executed along with related concepts **(0 M)**

2	**Communication of Concepts (2 Marks)**	Viva Voce	Communicates the concept used in problem solving well. **(2 M)**	Sufficiently communicates the concepts used in problem solving. **(1 M)**	Unable to communicate the concepts used in the problem. **(0 M)**

Lab Test rubrics (Max:20 marks)					
SN	**Criteria**	**Measuring Methods**	**Excellent**	**Good**	**Poor**
1	**Understanding of problem statements. (2 Marks)**	Observations	Student exhibits thorough understanding of requirements and applies suitable techniques for the problem **(2 M)**	Students have sufficient understanding of requirements and apply suitable techniques for the problem. **(1 M)**	Student does not have a clear understanding of requirements and is unable to apply any techniques for the problem. **(0 M)**

2	**Writing (5 Marks)**	Observations	Well-organized, clear, with proper logic of the problem. validations with all test cases are handled. **(5 M)**	Well-organized, clear, with partial logic of the problem and validations. **(1 - 4M)**	Student has not executed the program. **(0 M)**
3	**Execution and Format of Results (5 Marks)**	Observations	Students demonstrate the execution of the program with error correction and show performance efficiency. Appropriate validations with test cases (minimum 2) are handled. Execution Results should show in proper format **(5 M)**	Students demonstrate the execution of the program without error correction and appropriate validations with test case (minimum 1) is handled. Execution Results in without format **(1-4M)**	Execution Results with no comments and no output **(0 M)**
4	**Complexity / Innovation (2 Marks)**	Observations	Application based any one small concept should include in the program **(2 M)**	Without application based any one small concept should include in the program **(1M)**	Unable to include complexity/ innovation **(0 M)**
Viva Voce Rubrics (Max: 6 marks)					

1	**Conceptual Understandin g (2 Marks)**	Viva Voce	Explains thoroughly the program executed along with related concepts. **(3 M)**	Adequately explains the program executed along with related concepts **(1 – 2M)**	Unable to explain the program executed along with related concepts **(0 M)**
2	**Communicatio n of Concepts (3 Marks)**	Viva Voce	Communicates the concept used in problem solving well. **(3 M)**	Sufficiently communicat es the concepts used in problem solving. **(1-2M)**	Unable to communi cate the concepts used in the problem. **(0 M)**

13. Course Policy

- All the students
  - should bring their personal computers (fully charged) to the classroom.
  - should have an RVU mail ID.
  - should be able to connect to both RVCE and RVU WiFi.
- Use of mobile phones is not allowed during class hours. Also, they should not be connected to WiFi.
- In general, late submission by one day, without prior permission, will result in a penalty of 25%. Late submission beyond two days will not be accepted.
- Being late by more than 5 minutes in the first session and by more than 2 minutes in the remaining sessions will not be given attendance. Also, latecomers are required to not disturb others in the classroom.

Reviewed By (Name, Affiliation & Date):

Program Head (Name, Sign, Date): **SoCSE Dean (Sign & Date)**

Version History	Prepared by	Status
1:0 /	Dr. Mydhili K Nair	Valid till odd sem 2023
2:0/	Dr. Merin Thomas	Valid till odd sem 2024
3:0/	Dr. K Sailaja Kumar	Action
4.0		