| CST | ALGORITHM <br> ANALYSIS AND <br> DES | Category | L | T | P | Credit | Year of <br> Introduction |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | DESIGN | PCC | 3 | 1 | 0 | 4 | 2019 |

## Preamble:

The course introduces students to the design of computer algorithms, as well as analysis of algorithms. Algorithm design and analysis provide the theoretical backbone of computer science and are a must in the daily work of the successful programmer. The goal of this course is to provide a solid background in the design and analysis of the major classes of algorithms. At the end of the course students will be able to develop their own versions for a given computational task and to compare and contrast their performance.

Prerequisite:
Strong Foundation in Mathematics, Programming in C, Data Structures and Graph Theory.

Course Outcomes: After the completion of the course the student will be able to

| CO\# | CO |
| :---: | :--- |
| CO1 | Analyze any given algorithm and express its time and space complexities in <br> asymptotic notations. (Cognitive Level: Apply) |
| $\mathbf{C O 2}$ | Derive recurrence equations and solve it using Iteration, Recurrence Tree, <br> Substitution and Master's Method to compute time complexity of algorithms. <br> (Cognitive Level: Apply) |
| $\mathbf{C O 3}$ | Illustrate Graph traversal algorithms \& applications and Advanced Data <br> structures like AVL trees and Disjoint set operations. (Cognitive Level: <br> Apply) |
| $\mathbf{C O 4}$ | Demonstrate Divide-and-conquer, Greedy Strategy, Dynamic programming, <br> Branch-and Bound and Backtracking algorithm design techniques <br> (Cognitive Level: Apply) |
| $\mathbf{C O 5}$ | Classify a problem as computationally tractable or intractable, and discuss <br> strategies to address intractability (Cognitive Level: Understand) |
| $\mathbf{C O 6}$ | Identify the suitable design strategy to solve a given problem. (Cognitive <br> Level: Analyze) |

Mapping of course outcomes with program outcomes

|  | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| CO1 | 0 | 0 | 0 | 0 |  |  |  |  |  |  |  | 0 |
| CO 2 | 0 | 0 | 0 | 0 |  |  |  |  |  |  |  | 0 |
| CO 3 | 0 | 0 | 0 | 0 |  |  |  |  |  |  |  | 0 |
| CO 4 | 0 | 0 | 0 | 0 |  |  |  |  |  |  |  | 0 |
| CO5 | 0 | 0 |  |  |  |  |  |  |  |  |  | $\checkmark$ |
| CO6 | 0 | 0 | 0 | 0 |  |  |  |  |  |  |  | 0 |


| Abstract POs defined by National Board of Accreditation |  |  |  |
| :--- | :--- | :--- | :--- |
| PO\# | Broad PO | PO\# | Broad PO |
| PO1 | Engineering Knowledge | PO7 | Environment and Sustainability |
| $\mathbf{P O 2}$ | Problem Analysis | PO8 | Ethics |
| $\mathbf{P O 3}$ | Design/Development of solutions | PO9 | Individual and team work |
| $\mathbf{P O 4}$ | Conduct investigations of complex <br> problems | $\mathbf{P O 1 0}$ | Communication |
| $\mathbf{P O 5}$ | Modern tool usage | PO11 | Project Management and Finance |
| $\mathbf{P O 6}$ | The Engineer and Society | PO12 | Life long learning |

## Assessment Pattern

| Bloom's <br> Category | Continuous Assessment Tests |  | End Semester Examination <br> Marks (\%) |
| :--- | :---: | :---: | :---: |
|  | Test 1 (\%) | Test 2 (\%) |  |
| Remember | 30 | 30 | 30 |
| Understand | 30 | 30 | 40 |
| Apply | 40 | 40 |  |


| Analyze |  |  |  |
| :--- | :--- | :--- | :--- |
| Evaluate |  |  |  |
| Create |  |  |  |

## Mark Distribution

| Total Marks | CIE Marks | ESE Marks | ESE Duration |
| :---: | :---: | :---: | :---: |
| 150 | 50 | 100 | 3 |

## Continuous Internal Evaluation Pattern:

Attendance
Continuous Assessment Tests (Average of SeriesTests1 \& 2)
Continuous Assessment Assignment

## 10 marks

25 marks
15 marks

## Internal Examination Pattern:

Each of the two internal examinations has to be conducted out of 50 marks. First series test shall be preferably conducted after completing the first half of the syllabus and the second series test shall be preferably conducted after completing remaining part of the syllabus. There will be two parts: Part A and Part B. Part A contains 5 questions (preferably, 2 questions each from the completed modules and 1 question from the partly completed module), having 3 marks for each question adding up to 15 marks for part A. Students should answer all questions from Part A. Part B contains 7 questions (preferably, 3 questions each from the completed modules and 1 question from the partly completed module), each with 7 marks. Out of the 7 questions, a student should answer any 5.

## End Semester Examination Pattern:

There will be two parts; Part A and Part B. Part A contains 10 questions with 2 questions from each module, having 3 marks for each question. Students should answer all questions. Part B contains 2 full questions from each module of which student should answer any one. Each question can have maximum 2 sub-divisions and carries 14 marks.

## Syllabus

## Module-1 (Introduction to Algorithm Analysis)

Characteristics of Algorithms, Criteria for Analysing Algorithms, Time and Space Complexity Best, Worst and Average Case Complexities, Asymptotic Notations - Big-Oh (O), Big- Omega $(\Omega)$, Big-Theta $(\Theta)$, Little-oh (o) and Little- Omega ( $\omega$ ) and their properties. Classifying functions by their asymptotic growth rate, Time and Space Complexity Calculation of simple algorithms.

Analysis of Recursive Algorithms: Recurrence Equations, Solving Recurrence Equations Iteration Method, Recursion Tree Method, Substitution method and Master's Theorem (Proof not required).

## Module-2 (Advanced Data Structures and Graph Algorithms)

Self Balancing Tree - AVL Trees (Insertion and deletion operations with all rotations in detail, algorithms not expected); Disjoint Sets- Disjoint set operations, Union and find algorithms.

DFS and BFS traversals - Analysis, Strongly Connected Components of a Directed graph, Topological Sorting.

## Module-3 (Divide \& Conquer and Greedy Strategy)

The Control Abstraction of Divide and Conquer- 2-way Merge sort, Strassen's Algorithm for Matrix Multiplication-Analysis. The Control Abstraction of Greedy Strategy- Fractional Knapsack Problem, Minimum Cost Spanning Tree Computation- Kruskal’s Algorithms - Analysis, Single Source Shortest Path Algorithm - Dijkstra's Algorithm-Analysis.

## Module-4 (Dynamic Programming, Back Tracking and Branch \& Bound))

The Control Abstraction- The Optimality Principle- Matrix Chain Multiplication-Analysis, All Pairs Shortest Path Algorithm - Floyd-Warshall Algorithm-Analysis. The Control Abstraction of Back Tracking - The N Queen's Problem. Branch and Bound Algorithm for Travelling Salesman Problem.

## Module-5 (Introduction to Complexity Theory)

Tractable and Intractable Problems, Complexity Classes - P, NP, NP- Hard and NP-Complete Classes- NP Completeness proof of Clique Problem and Vertex Cover Problem- Approximation algorithms- Bin Packing, Graph Coloring. Randomized Algorithms (Definitions of Monte Carlo and Las Vegas algorithms), Randomized version of Quick Sort algorithm with analysis.

## Text Books

1. T.H.Cormen, C.E.Leiserson, R.L.Rivest, C. Stein, Introduction to Algorithms, $2^{\text {nd }}$ Edition, Prentice-Hall India (2001)
2. Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran, "Fundamentals of Computer Algorithms", $2^{\text {nd }}$ Edition, Orient Longman Universities Press (2008)
3. Sara Baase and Allen Van Gelder -Computer Algorithms, Introduction to Design and Analysis, $3{ }^{\text {rd }}$ Edition, Pearson Education (2009)

## Reference Books

1. Jon Kleinberg, Eva Tardos, "Algorithm Design", First Edition, Pearson (2005)
2. Robert Sedgewick, Kevin Wayne, "Algorithms",4th Edition Pearson (2011)
3. GIlles Brassard, Paul Brately, "Fundamentals of Algorithmics", Pearson (1996)
4. Steven S. Skiena, "The Algorithm Design Manual", $2^{\text {nd }}$ Edition, Springer(2008)

## Course Level Assessment Questions

## Course Outcome 1 (CO1):

1. Is $2^{\mathrm{n}+1}=\mathrm{O}\left(2^{\mathrm{n}}\right)$ ? Is $2^{2 \mathrm{n}}=\mathrm{O}\left(2^{\mathrm{n}}\right)$ ? Justify your answer.
2. What is the need of asymptotic analysis in calculating time complexity? What are the notations
used for asymptotic analysis?
3. Calculate the time complexity for addition of two matrices.
4. Define time complexity and space complexity. Write an algorithm for adding n natural numbers and analyse the time and space requirements of the algorithm.

## Course Outcome 2 (CO2):

1. State Master's theorem for solving recurrences.
2. Solve the recurrence $T(n)=3 T(n-2)$, using iteration method
3. State the conditions in recurrences where Master Theorem is not applicable.
4. Solve the following recurrence equations using Master's theorem.
a) $T(n)=8 T(n / 2)+100 n^{2}$
b) $T(n)=2 T(n / 2)+10 n$
5. Using Recursion Tree method, Solve $T(n)=2 T(n / 10)+T(9 n / 10)+n$. Assume constant time for small values of n .

## Course Outcome 3 (CO3):

1. Explain the rotations performed for insertion in AVL tree with example.
2. Write down BFS algorithm and analyse the time complexity. Perform BFS traversal on the given graph starting from node A. If multiple node choices are available for next travel, choose the next node in alphabetical order.

3. Find the minimum and maximum height of any AVL-tree with 7 nodes? Assume that the height of a tree with a single node is 0 . (3)
4. Find any three topological orderings of the given graph.


Course Outcome 4 (CO4):

1. Give the control abstraction for Divide and Conquer method.
2. Construct the minimum spanning tree for the given graph using Kruskal's algorithm. Analyse the complexity of the algorithm.

3. Compare Divide and Conquer and Dynamic programming methodologies
4. What is Principle of Optimality?
5. Define Travelling Salesman Problem (TSP). Apply branch and bound algorithm to solve TSP for the following graph, assuming the start city as ' $a$ '. Draw the state space tree.


## Course Outcome 5 (CO5):

1. Compare Tractable and Intractable Problems
2. With the help of suitable code sequence convince Vertex Cover Problem is an example of NP-Complete Problem
3. Explain Vertex Cover problem using an example. Suggest an algorithm for finding Vertex Cover of a graph.
4. Write short notes on approximation algorithms.
5. Compare Conventional quick sort algorithm and Randomized quicksort with the help of a suitable example?

Course Outcome 6 (CO6): (CO attainment through assignment only, not meant for examinations)
Choosing the best algorithm design strategy for a given problem after applying applicable design strategies - Sample Problems Given.

1. Finding the Smallest and Largest elements in an array of ' $n$ ' numbers
2. Fibonacci Sequence Generation.
3. Merge Sort
4. Travelling Sales Man Problem
5. $0 / 1$ Knapsack Problem

## Model Question Paper

## QP CODE:

Reg No: $\qquad$
Name: $\qquad$ PAGES : 4

## APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY SIXTH SEMESTER B.TECH DEGREE EXAMINATION, MONTH \& YEAR

Course Code: CST 306
Course Name: Algorithm Analysis and Design
Max. Marks : 100
Duration: 3 Hours
PART A

## Answer All Questions. Each Question Carries 3 Marks

1. Define asymptotic notation? Arrange the following functions in increasing order of asymptotic growth rate.

$$
\mathrm{n}^{3}, 2^{\mathrm{n}}, \log \mathrm{n}^{3}, 2^{100}, \mathrm{n}^{2} \log \mathrm{n}, \mathrm{n}^{\mathrm{n}}, \log \mathrm{n}, \mathrm{n}^{0.3}, 2^{\operatorname{logn}}
$$

2. State Master's Theorem. Find the solution to the following recurrence equations using Master's theorem.
a) $T(n)=8 T(n / 2)+100 n^{2}$
b) $T(n)=2 T(n / 2)+10 n$
3. Find any two topological ordering of the DAG given below.

4. Show the UNION operation using linked list representation of disjoint sets.
5. Write the control abstraction of greedy strategy to solve a problem.
6. Write an algorithm based on divide-and-conquer strategy to search an element in a given list. Assume that the elements of list are in sorted order.
7. List the sequence of steps to be followed in Dynamic Programming approach.
8. Illustrate how optimal substructure property could be maintained in FloydWarshall algorithm.
9. Differentiate between P and NP problems.
10. Specify the relevance of approximation algorithms.

## Part B

(Answer any one question from each module. Each question carries 14 Marks)
11. (a) Define $\operatorname{Big} O, \operatorname{Big} \Omega$ and $\operatorname{Big} \Theta$ Notation and illustrate them graphically.
(b) Solve the following recurrence equation using recursion tree method

$$
\begin{align*}
& \mathrm{T}(\mathrm{n})=\mathrm{T}(\mathrm{n} / 3)+\mathrm{T}(2 \mathrm{n} / 3)+\mathrm{n}, \text { where } \mathrm{n}>1  \tag{7}\\
& \mathrm{~T}(\mathrm{n})=1 \text {, Otherwise }
\end{align*}
$$

12. (a) Explain the iteration method for solving recurrences and solve the following recurrence equation using iteration method.

$$
\mathrm{T}(\mathrm{n})=3 \mathrm{~T}(\mathrm{n} / 3)+\mathrm{n} ; \mathrm{T}(1)=1
$$

(b) Determine the time complexities of the following two functions fun1() and fun2().

```
i) int fun1(int n)
    \{
        if \((\mathrm{n}<=1\) ) return n ;
                                return \(2 *\) fun \(1(n-1)\);
    \}
ii) int fun2 (int n)
    \{
        if ( \(\mathrm{n}<=1\) ) return n ;
        return fun2 \((\mathrm{n}-1)+\) fun \(2(\mathrm{n}-1)\)
        \}
```

13. (a) Write DFS algorithm and analyse its time complexity. Illustrate the classification of edges in DFS traversal.
(b) Find the strongly connected components of the digraph given below:


## OR

14. (a) Illustrate the advantage of height balanced binary search trees over binary search trees? Explain various rotations in AVL trees with example.
(b) Perform the following operations in the given AVL trees.
i) Insert 70
ii) Delete 55

15. (a) State Fractional Knapsack Problem and write Greedy Algorithm for

Fractional Knapsack Problem.
(b) Find the optimal solution for the following Fractional Knapsack problem.

Given the number of items $(\mathrm{n})=7$, capacity of $\operatorname{sack}(m)=15$,
$\mathrm{W}=\{2,3,5,7,1,4,1\}$ and $\mathrm{P}=\{10,5,15,7,6,18,3\}$

## OR

16. (a) Write and explain merge sort algorithm using divide and conquer strategy using the data $\{30,19,35,3,9,46,10\}$. Also analyse the time complexity.
(b) Write the pseudo code for Dijkstra's algorithm. Compute the shortest distance from vertex 1 to all other vertices using Dijkstra's algorithm.

17. (a) Write Floyd-Warshall algorithm and analyse its complexity.
(b) Write and explain the algorithm to find the optimal parenthesization of matrix chain product whose sequence of dimension is $4 \times 10,10 \times 3,3 \times 12,12 \times 20$.

## OR

18. (a) Explain the concept of Backtracking method using 4 Queens problem.
(b) Define Travelling Salesman Problem (TSP). Apply branch and bound algorithm to solve TSP for the following graph, assuming the start city as ' $a$ '. Draw the state space tree.

19. (a) State bin packing problem? Explain the first fit decreasing strategy
(b) Prove that the Clique problem is NP-Complete.

## OR

20. (a) Explain the need for randomized algorithms. Differentiate Las Vegas and Monte Carlo algorithms.
(b) Explain randomized quicksort and analyse the expected running time of randomized quicksort with the help of a suitable example?

## Teaching Plan

| No | Topic | No. of Hours <br> $\mathbf{( 4 5 ~ h r s )}$ |
| :--- | :--- | :---: |
| Module -1 (Introduction to Algorithm Analysis) 9 hrs. |  |  |
| 1.1 | Introduction to Algorithm Analysis: Characteristics of Algorithms. | 1 hour |
| 1.2 | Criteria for Analysing Algorithms, Time and Space Complexity - Best, <br> Worst and Average Case Complexities. | 1 hour |
| 1.3 | Asymptotic Notations - Properties of Big-Oh (O), Big- Omega ( $\Omega$ ), Big- <br> Theta ( $\Theta$ ), Little-Oh (o) and Little- Omega ( $\omega$ ). | 1 hour |
| 1.4 | Illustration of Asymptotic Notations | 1 hour |


| 1.5 | Classifying functions by their asymptotic growth rate | 1 hour |
| :---: | :---: | :---: |
| 1.6 | Time and Space Complexity Calculation of algorithms/code segments. | 1 hour |
| 1.7 | Analysis of Recursive Algorithms: Recurrence Equations, Solving Recurrence Equations - Iteration Method. | 1 hour |
| 1.8 | Recursion Tree Method | 1 hour |
| 1.9 | Substitution method and Master's Theorem and its Illustration. | 1 hour |
| Module-2 (Advanced Data Structures and Graph Algorithms) 10 Hrs. |  |  |
| 2.1 | Self Balancing Trees - Properties of AVL Trees, Rotations of AVL Trees | 1 hour |
| 2.2 | AVL Trees Insertion and Illustration | 1 hour |
| 2.3 | AVL Trees Deletion and Illustration | 1 hour |
| 2.4 | Disjoint set operations. | 1 hour |
| 2.5 | Union and find algorithms. | 1 hour |
| 2.6 | Illustration of Union and find algorithms | 1 hour |
| 2.7 | Graph Algorithms: BFS traversal, Analysis. | 1 hour |
| 2.8 | DFS traversal, Analysis. | 1 hour |
| 2.9 | Strongly connected components of a Directed graph. | 1 hour |
| 2.10 | Topological Sorting. | 1 hour |
| Module-3 (Divide \& Conquer and Greedy Method) 8 Hrs |  |  |
| 3.1 | Divide and Conquer: The Control Abstraction. | 1 hour |
| 3.2 | 2-way Merge Sort, Analysis. | 1 hour |
| 3.3 | Strassen's Algorithm for Matrix Multiplication, Analysis | 1 hour |


| 3.4 | Greedy Strategy: The Control Abstraction. | 1 hour |
| :---: | :---: | :---: |
| 3.5 | Fractional Knapsack Problem. | 1 hour |
| 3.6 | Minimum Cost Spanning Tree Computation- Kruskal's Algorithm, Analysis. | 1 hour |
| 3.7 | Single Source Shortest Path Algorithm - Dijkstra's Algorithm | 1 hour |
| 3.8 | Illustration of Dijkstra's Algorithm-Analysis. | 1 hour |
| Module-4 (Dynamic Programming, Back Tracking and Branch and Bound) 8 Hrs. |  |  |
| 4.1 | Dynamic Programming: The Control Abstraction, The Optimality Principle. | 1 hour |
| 4.2 | Matrix Chain Multiplication-Analysis. | 1 hour |
| 4.3 | Illustration of Matrix Chain Multiplication-Analysis. | 1 hour |
| 4.4 | All Pairs Shortest Path Algorithm- Analysis and Illustration of FloydWarshall Algorithm. | 1 hour |
| 4.5 | Back Tracking: The Control Abstraction . | 1 hour |
| 4.6 | Back Tracking: The Control Abstraction - The N Queen's Problem. | 1 hour |
| 4.7 | Branch and Bound:- Travelling salesman problem. | 1 hour |
| 4.8 | Branch and Bound:- Travelling salesman problem. | 1 hour |
| Module-5 (Introduction to Complexity Theory) 10 Hrs |  |  |
| 5.1 | Introduction to Complexity Theory: Tractable and Intractable Problems. | 1 hour |
| 5.2 | Complexity Classes - P, NP. | 1 hour |
| 5.3 | NP- Hard and NP-Complete Problems. | 1 hour |
| 5.4 | NP Completeness Proof of Clique Problem. | 1 hour |


| 5.5 | NP Completeness Proof of Vertex Cover Problem. | 1 hour |
| :--- | :--- | :---: |
| 5.6 | Approximation algorithms- Bin Packing Algorithm and Illustration. | 1 hour |
| 5.7 | Graph Colouring Algorithm and Illustration. | 1 hour |
| 5.8 | Randomized Algorithms (definitions of Monte Carlo and Las Vegas <br> algorithms). | 1 hour |
| 5.9 | Randomized Version of Quick Sort Algorithm with Analysis. | 1 hour |
| 5.10 | Illustration of Randomized Version of Quick Sort Algorithm with <br> Analysis. | 1 hour |

