BANGALORE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, UVCE, BENGALURU
B.Tech. PROGRAMME IN COMPUTER SCIENCE AND ENGINEERING

Course Code	**18CIPC403**					
Category	Engineering Science Courses : Professional Core					
Course title	**DESIGN AND ANALYSIS OF ALGORITHMS – THEORY**					
Scheme and Credits	No. of Hours/Week					Semester - IV CSE/ISE
	L	T	P	SS	Credits	
	2	2	0	0	3	
CIE Marks: 50	SEE Marks: 50	Total Max. Marks: 100				Duration of SEE: 03 Hours
Prerequisites (if any): NIL						

COURSE OBJECTIVES:
The course will enable the students to
1. Understand the importance of algorithm and need for finding the time complexity of an algorithm.
2. Learn the algorithms under Brute force, Divide and Conquer, Greedy and Dynamic programming concepts.
3. Compute the time complexity of various algorithmic techniques.
4. Acquire the Knowledge of P, NP and NP Hard problems.
5. Learn to apply the algorithmic techniques to real world problems.

UNIT I: INTRODUCTION 09 Hours
The Notion of Algorithm, Fundamentals of algorithmic Problem Solving. The Analysis of Framework. Asymptotic Notations and Standard Efficiency Classes. Mathematical analysis of Non-Recursive Algorithms. Mathematical Analysis of recursive algorithms. An Example: the Fibonacci Numbers. Empirical Analysis of Algorithms. Algorithm Visualization.

UNIT II: BRUTE FORCE AND DECREASE & CONQUER 10 Hours
Brute-Force: Selection Sort and Bubble Sort. Sequential Search and Brute-Force String Matching. Closest-Pair and Convex-Hull Problems by Brute Force. Exhaustive Search. Depth First Search (DFS), Breadth First Search (BFS), Applications of DFS and BFS, Decrease and conquer: Insertion Sort, Topological Sort, Generating Permutations, Binary search, Computing Median and the Selection problem.

UNIT III: DIVIDE & CONQUER AND TRANSFORM & CONQUER 10 Hours
Divide-and-Conquer: Mergesort. Quicksort. Binary Tree Traversals and Related Properties. Multiplication of Large Integers and Strassen's Matrix Multiplication. Transform and Conquer: Presorting and its Applications, Balanced Search Trees, Heaps and Heap sort. Horner’s rule and Binary exponentiation, Space & Time Tradeoff: Horspool string matching algorithm, Btrees.

UNIT IV: DYNAMIC PROGRAMMING AND GREEDY TECHNIQUES 10 Hours

Dynamic Programming: Basic examples, The Knapsack Problem and Memory Functions Binomial Coefficients, Optimal Binary Search Trees, Warshall's and Floyd's Algorithms Greedy Approach: Prim's Algorithm. Kruskal's Algorithm. Dijkstra's Algorithm. Huffman Trees.

UNIT V: COPING WITH LIMITATIONS OF ALGORITHM POWER 09 Hours

Backtracking: n-Queens Problem, Subset-Sum Problem, Branch-and-Bound: Travelling Salesman problem, Knapsack Problem, Approximation Algorithms for NP hard problems, Limitations of Algorithm Power: Decision Trees, P, NP, and NP-Complete Problems.

TEXT BOOKS:

1. Introduction to the Design and Analysis of Algorithms, by Anany Levitin, Pearson Education, Third Edition, 2014.
2. Computer Algorithms, by Horowitz E., Sahani S., Rajasekharan S., Galgotia Publications, 2001.

REFERENCE BOOKS:

1. Introduction to Algorithms, Cormen T.H, Leiserson C. E, Rivest R.L, Stein C, 3rd Edition, PHI 2010.
2. Data Structures and Algorithm Analysis in C++, by Mark Allen Weiss, Pearson Education, 4th edition, 2012.
3. Data Structures - Seymour Lipschutz, Schaum's Outlines, Revised 1st edition, McGraw Hill, 2014.

e-BOOKS/ONLINE RESOURCES:

1. https://india.oup.com/product/design-and-analysis-of-algorithms-9780198093695.
2. https://www.pdfdrive.com/design-and-analysis-of-algorithms-books.html.

MOOCs:

1. https://nptel.ac.in/courses/106106093/35.
2. https://eu.udacity.com/course/intro-to-algorithms--cs215.
3. https://www.edx.org/course/algorithms-data-structures-microsoft-dev285x-1.
4. https://visualgo.net/en.
5. https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-046j-design-and-analysis-of-algorithms-spring-2015/.

COURSE OUTCOMES:

The students at the end of the course, will be able to

CO1: Acquire the knowledge on fundamentals of algorithmic design steps, analyse concepts and types of algorithm design techniques.

CO2: Understand and analyze the design of algorithms using Brute force, Divide & Conquer, Decrease & Conquer, Transform & conquer, Dynamic Programming, Greedy technique, Backtracking, Branch & Bound techniques.

CO3: Assess the performance and correctness of algorithms.

CO4: Design and Implement efficient algorithms by applying appropriate design techniques for solving real world problems.

CO5: Design solutions for various engineering applications using appropriate algorithms.

SCHEME OF EXAMINATION:

CIE – 50 Marks	Test I (Any Three Units) - 20 Marks	Quiz I – 5 Marks	25 Marks	Total: 50 Marks
	Test II (Remaining Two Units) - 20 Marks	Quiz II – 5 Marks	25 Marks	
SEE – 100 Marks	**Q1 (Compulsory):** MCQs or Short answer type questions for 15 Marks covering entire syllabus.		15 Marks	Total: 100 Marks
	Q2 & Q3 from Units which have 09 Hours are compulsory.		17 * 2 = 34 Marks	
	Q4 or Q5, Q6 or Q7 and Q8 or Q9 from Units which have 10 Hours shall have Internal Choice.		17 * 3 = 51 Marks	

Note: SEE shall be conducted for 100 Marks and the Marks obtained is scaled down to 50 Marks.

*****

BANGALORE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, UVCE, BENGALURU
B.Tech. PROGRAMME IN COMPUTER SCIENCE AND ENGINEERING

Course Code	**18CIPC407**					
Category	Engineering Science Courses : Professional Core					
Course title	**DESIGN AND ANALYSIS OF ALGORITHMS – LABORATORY**					
Scheme and Credits	No. of Hours/Week					Semester - IV CSE/ISE
	L	T	P	SS	Credits	
	0	0	3	0	1.5	
CIE Marks: 50	SEE Marks: 50	Total Max. Marks: 100				Duration of SEE: 03 Hours
Prerequisites (if any): NIL						

COURSE OBJECTIVES:
The course will enable the students to
1. Design and implement various algorithms in C++.
2. Determine the time complexity of various sorting algorithms.
3. Employ various design strategies for problem solving.
4. Measure and compare the performance of different algorithms.
5. Understand, develop and analyse the various algorithms under Divide & Conquer, Greedy, Dynamic and backtracking techniques.

DESCRIPTION:
Design, develop, and implement the specified algorithms for the following problems using C++ language under LINUX /Windows environment.

LAB PROGRAMS:

1. Sort a given set of elements using Merge sort and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot graph of the time taken versus number of elements. The elements can be read from file or generated using random number generator.
2. Sort a given set of elements using Quick sort and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot graph of the time taken versus number of elements. The elements can be read from file or generated using random number generator.
3. Write a program to perform insert and delete operations in Binary Search Tree.
4. Print all the nodes reachable from a given starting node in a digraph using BFS method.
5. a) Obtain the Topological ordering of vertices in a given digraph.
   b) Compute the transitive closure of a given directed graph using Warshall’s algorithm.
6. a) Check whether a given graph is connected or not using DFS method.
   b) Implement Floyd’s algorithm for the All-Pairs- Shortest-Paths problem.
7. Sort a given set of elements using the Heap sort method and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot a graph of the time taken versus number of elements.

8. Search for a pattern string in a given text using Horspool String Matching algorithm.
9. Implement 0/1 Knapsack problem using dynamic programming.
10. Find Minimum Cost Spanning Tree of a given undirected graph using Prim’s algorithm.
11. Find Minimum Cost Spanning Tree of a given undirected graph using Kruskal's algorithm.
12. From a given vertex in a weighted connected graph, find shortest paths to other vertices using Dijkstra's algorithm.
13. Write a program to solve Travelling Sales Person problem using dynamic programming approach.
14. Implement N Queen's problem using Back Tracking.
15. Write a program to construct an AVL tree for a given set of integers.

COURSE OUTCOMES:

The student at the end of the course, will be able to

CO1: Design algorithms using appropriate design techniques (brute-force, greedy, dynamic programming etc).

CO2: Implement a variety of algorithms such as sorting, graph related, combinatorial, etc, in a high level language.

CO3: Develop programs and analyse its time complexity.

CO4: Analyze and compare the performance of algorithms using language features.

CO5: Apply and implement learned algorithm design techniques and data structures to solve real world problems.

SCHEME OF EXAMINATION:

Continuous Internal Evaluation (CIE) Laboratory - (50 Marks)	**Marks**	**Semester End Evaluation (SEE) Laboratory - (100 Marks)**	**Marks**
Performance of the student in the laboratory, every week	20	Write up	20
Test at the end of the semester	20	Execution	60
Viva voce	10	Viva voce	20
Total	**50**	**Total**	**100**

Note: SEE shall be conducted for 100 Marks and the Marks obtained is scaled down to 50 Marks.

*****