|  |  |
| --- | --- |
| MONO | **DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**  **NATIONAL INSTITUTE OF TECHNOLOGY PATNA**  Ashok Raj Path, PATNA 800 005 (Bihar), India |
| Phone No.: 0612 – 2372715, 2370419, 2370843, 2371929, 2371930, 2371715 Fax – 0612- 2670631 Website: [www.nitp.ac.in](http://www.nitp.ac.in/) |

***CSL4403 Design & Analysis of Algorithms Lab***

**L-T-P-Cr: 0-0-3-1**

**Pre-requisites:** Basics of C programming language

**Objectives/Overview:**

* To provide a solid foundation in algorithm design and analysis.
* To analyse the asymptotic performance of algorithms.
* To apply important algorithmic design paradigms and methods of analysis.
* To synthesize efficient algorithms in common engineering design situations.

**Course Outcomes:**

At the end of the course, a student should:

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Outcome** | **Mapping to POs** |
|  | Analyze worst-case running times of algorithms using asymptotic analysis. | PO1, PO3 |
|  | Able to design algorithms using greedy strategy, divide and conquer approach, dynamic programming, and max flow - min cut theory. | PO1, PO2 |
|  | Able to compare between different data structures and pick an appropriate data structure for a design situation. | PO1, PO3 |
|  | Know the basic representation of undirected and directed graph and how to use analyze depth first and breath-first search of graphs. | PO2, PO3 |

**List of Experiments:**

1. Divide and Conquer:
   * 1. Implement Binary Search using Divide and Conquer approach.
     2. Find Maximum and Minimum element from an array of integer using Divide and Conquer approach.
     3. Implement Merge Sort using Divide and Conquer.
2. Greedy method:
   * + 1. Minimum Cost Spanning Tree by Prim's Algorithm/ Kruskal's Algorithm.
       2. Knapsack problem.
3. Dynamic Programming:
4. Implement Single Source shortest Path for a graph ( Dijkstra , Bellman Ford Algorithm ).
5. Implement Traveling Salesman Problem.
6. Allocation of mini projects.
7. Backtracking :
8. Implement 8 Queen Problem
9. Graph Coloring Problem
10. Brunch and Bound:
11. Implement 15 Puzzle Problem
12. Graph Traversal Algorithm:
13. Implement Breadth First Search (BFS)
14. Implement Depth First Search (DFS)

Course instructor can add experiments to the above list and/or modify some of the experiments in the above list depending upon course contents covered and examples used in the corresponding theoretical course.