| **Course Code** | **Course Name** | **Credit** |
| --- | --- | --- |
| **CSC402** | Analysis of Algorithms | **3** |

| **Prerequisite:** Data structure concepts, Discrete structures | |
| --- | --- |
| **Course Objectives:** | |
| 1 | To provide mathematical approaches for Analysis of Algorithms |
| 2 | To understand and solve problems using various algorithmic approaches |
| 3 | To analyze algorithms using various methods |
| **Course Outcomes:** At the end of the course learner will be able to | |
| 1 | Analyze the running time and space complexity of algorithms. |
| 2 | Describe, apply and analyze the complexity of divide and conquer strategy. |
| 3 | Describe, apply, and analyze the complexity of greedy strategy. |
| 4 | Describe, apply and analyze the complexity of dynamic programming strategy. |
| 5 | Explain and apply backtracking, branch, and bound. |
| 6 | Explain and apply string matching techniques. |

| **Module** |  | **Detailed Contents** | **Hours** |
| --- | --- | --- | --- |
| 1 |  | **Introduction** | 8 |
|  | 1.1 | Performance analysis, space, and time complexity Growth of function, Big-Oh, Omega Theta notation Mathematical background for algorithm analysis.  Complexity class: Definition of P, NP, NP-Hard, NP-Complete Analysis of selection sort, insertion sort. |  |
|  | 1.2 | Recurrences: The substitution method, Recursion tree method, Master method |  |
| 2 |  | **Divide and Conquer Approach** | 6 |
|  | 2.1 | General method, Merge sort, Quicksort, Finding minimum and maximum algorithms and their Analysis, Analysis of Binary search. |  |
| 3 |  | **Greedy Method Approach** | 6 |
|  | 3.1 | General Method, Single source shortest path: Dijkstra Algorithm Fractional Knapsack problem, Job sequencing with deadlines, Minimum cost spanning trees: Kruskal and Prim‟s algorithms |  |
| 4 |  | **Dynamic Programming Approach** | 9 |
|  | 4.1 | General Method, Multistage graphs, Single source shortest path: Bellman-Ford Algorithm  All pair shortest path: Floyd Warshall Algorithm, Assembly-line scheduling Problem 0/1 knapsack Problem, Traveling Salesperson problem, Longest common subsequence |  |
| 5 |  | **Backtracking and Branch and bound** | 6 |
|  | 5.1 | General Method, Backtracking: N-queen problem, Sum of subsets, Graph coloring |  |
|  | 5.2 | Branch and Bound: Traveling Salesperson Problem, 15 Puzzle problem |  |
| 6 |  | **String Matching Algorithms** | 4 |
|  | 6.1 | The Naïve string-matching algorithm, The Rabin Karp algorithm, The Knuth-Morris-Pratt algorithm |  |

| **Textbooks:** | |
| --- | --- |
| 1 | T. H. Cormen, C.E. Leiserson, R. L. Rivest, and C. Stein, “Introduction to algorithms”, 2nd Edition, PHI Publication 2005. |
| 2 | Ellis Horowitz, Sartaj Sahni, S. Rajsekaran. “Fundamentals of computer algorithms ' University Press. |

| **References:** | |
| --- | --- |
| 1 | Sanjoy Dasgupta, Christos Papadimitriou, Umesh Vazirani, “Algorithms”, Tata McGraw Hill Edition. |
| 2 | S. K. Basu, “Design Methods and Analysis of Algorithm”, PHI |

| **Assessment:** | |
| --- | --- |
| **Internal Assessment:** | |
| Assessment consists of two class tests of 20 marks each. The first-class test is to be conducted when approx. 40% syllabus is completed and second class test when additional 40% syllabus is completed. The duration of each test shall be one hour. | |
| **End Semester Theory Examination:** | |
| 1 | Question paper will comprise a total of six questions. |
| 2 | All question carries equal marks |
| 3 | Questions will be mixed in nature (for example supposed Q.2 has part (a) from module 3 then part (b) will be from any module other than module 3) |
| 4 | Only Four questions need to be solved. |
| 5 | In question, paper weightage of each module will be proportional to the number of respective lecture hours as mentioned in the syllabus. |

Performance analysis,

Space and time complexity,

Growth of function,

Big-Oh, Omega and Theta notation.

Mathematical background for algorithm analysis.

Complexity class: Definition of P, NP, NP-Hard, NP-Complete

Exp 1.

{

1. selection sort

2. insertion sort.

**}**

**Divide and Conquer Approach**3.Merge sort.

4. Quick sort.

Assignment 1

{

5. Binary search.

}

**Greedy Method Approach**

Single source shortest path- Dijkstra

Fractional Knapsack problem

Job sequencing with deadlines

Minimum cost spanning trees-Kruskal and Prim‟s algorithm

**Dynamic Programming Approach**

Single source shortest path- Bellman-Ford

All pair shortest path- Floyd Warshall

0/1 knapsack

Traveling salesperson problem

Longest common subsequence

**Backtracking and Branch and bound**

N-queen problem

Sum of subsets

Graph coloring

**String Matching Algorithms**

The Naïve string-matching Algorithms

The Rabin Karp algorithm

The Knuth-Morris-Pratt algorithm

| **Course Name** | **Lab Name** | **Credit** |
| --- | --- | --- |
| CSL401 | **Analysis of Algorithms Lab** | **1** |

| **Prerequisite: Basic knowledge of programming and data structure** | |
| --- | --- |
| **Lab Objectives:** | |
| 1 | To introduce the methods of designing and analyzing algorithms |
| 2 | Design and implement efficient algorithms for a specified application |
| 3 | Strengthen the ability to identify and apply the suitable algorithm for the given real-world problem. |
| 4 | Analyze worst-case running time of algorithms and understand fundamental algorithmic problems. |
| **Lab Outcomes:** At the end of the course, the students will be able to | |
| 1 | Implement the algorithms using different approaches. |
| 2 | Analyze the complexities of various algorithms. |
| 3 | Compare the complexity of the algorithms for specific problem. |

| **Description** | | |
| --- | --- | --- |
| Implementation can be in any language. | | |
| **Suggested Practical List:** | | |
| **Sr No** |  | **Suggested Experiment List** |
| **1** |  | **Introduction** |
|  | **1.1** | Selection sort, Insertion sort |
| **2** |  | **Divide and Conquer Approach** |
|  | **2.1** | Finding Minimum and Maximum, Merge sort, Quick sort, Binary search |
| **3** |  | **Greedy Method Approach** |
|  | **3.1** | Single source shortest path- Dijkstra  Fractional Knapsack problem  Job sequencing with deadlines  Minimum cost spanning trees-Kruskal and Prim‟s algorithm |
| **4** |  | **Dynamic Programming Approach** |
|  | **4.1** | Single source shortest path- Bellman Ford  All pair shortest path- Floyd Warshall  0/1 knapsack  Traveling salesperson problem  Longest common subsequence |
| **5** |  | **Backtracking and Branch and bound** |
|  | **5.1** | N-queen problem  Sum of subsets  Graph coloring |
| **6** |  | **String Matching Algorithms** |
|  | **6.1** | The Naïve string-matching Algorithms  The Rabin Karp algorithm  The Knuth-Morris-Pratt algorithm |

| **Term Work:** | |
| --- | --- |
| 1 | Term work should consist of 10 experiments. |
| 2 | Journal must include at least 2 assignments on content of theory and practical of “Analysis of Algorithms” |
| 3 | The final certification and acceptance of term work ensures satisfactory performance of laboratory work and minimum passing marks in term work. |
| 4 | Total 25 Marks (Experiments: 15-marks, Attendance Theory & Practical: 05-marks, Assignments: 05-marks) |