# BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
## WORK INTEGRATED LEARNING PROGRAMMES
## COURSE HANDOUT

# Part A: Content Design

| | |
|---|---|
| **Course Title** | DATA STRUCTURES AND ALGORITHMS DESIGN |
| **Course No(s)** | DSECLZG519 |
| **Credit Units** | **5** |
| **Course Author** | Febin. A.Vahab (01/10/2020) |
| **Version No** | 2.0 |
| **Minor Edits** | Parthasarathy PD (01/10/2022) |

**Course Description**

The course covers design, implementation and applications of basic and advanced data structures including trees, graphs, bloom filters. The course also covers algorithm design techniques like greedy, dynamic, map reduce etc. using examples from sorting, searching, graph theory, networking and number theory. The complexity issues are also discussed further.

**Course Objectives**

| No | Objective |
|---|---|
| **CO1** | Introduce mathematical and experimental techniques to analyze algorithms |
| **CO2** | Introduce linear and non-linear data structures and best practices to choose appropriate data structure for a given application |
| **CO3** | Teach various dictionary data structures (Lists, Trees, Heaps, Bloom filters) with illustrations on possible representation, various operations and their efficiency |
| **CO4** | Exposes students to various sorting and searching techniques |
| **CO5** | Discuss in detail various algorithm design approaches ( Greedy method, divide and conquer and dynamic programming) with appropriate examples, methods to make correct design choice and the efficiency concerns |
| **CO6** | Introduce complexity classes , notion of NP-Completeness, ways of classifying problem into appropriate complexity class |
| **CO7** | Introduce reduction method to prove a problem’s complexity class. |

**Learning Outcomes:**

| No | Learning Outcomes |
|---|---|
| LO1 | Describe various fundamental and advanced data structures, their properties, algorithm design techniques and various means of evaluating algorithms |
| LO2 | Demonstrate the ability to evaluate algorithms, to select from a range of possible options, to provide justification for that selection, and to implement the algorithm in a particular context. |
| LO3 | Solve problems using Algorithms for Linear and Non-Linear Data Structures |
| LO4 | Explain with a practical example, each of the algorithm design strategies (greedy, divide-and- |

| | |
|---|---|
| | conquer, dynamic programming and map-reduce) |
| LO5 | Use brute-force, greedy, divide-and-conquer, recursive backtracking, dynamic programming and map reduce techniques to solve a given algorithm design problem. |
| LO6 | Relate the real-world problems to known data structures and algorithms leading to the recommend appropriate solutions in representation and implementation. |
| LO7 | Explain the significance of NP-completeness |
| LO8 | Classify problems into complexity classes P and NP and to prove hardness of problems |

**Textbook(s):**

| No | Author(s), Title, Edition, Publishing House |
|---|---|
| T1 | Algorithms Design: Foundations, Analysis and Internet Examples Michael T. Goodrich, Roberto Tamassia, 2006, Wiley (Students Edition) |

**Reference Book(s) & other resources:**

| No | Author(s), Title, Edition, Publishing House |
|---|---|
| R1 | Introduction to Algorithms, TH Cormen, CE Leiserson, RL Rivest, C Stein, Third Ed, 2009, PHI |
| R2 | Data Structures, Algorithms and Applications in Java, Sartaj Sahni, Second Ed, 2005, Universities Press |
| R3 | Data Structures & Algorithms in Python , Michael T. Goodrich, Roberto Tamassia, Michael H Goldwasser, Wiley, 2013 |
| R4 | Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser - Data Structures and Algorithms in Python-Wiley (2013) |

## CONTENT STRUCTURE

| No | Title of the Module | References |
|---|---|---|
| M1 | **Analyzing Algorithms**<br>1.1. Theoretical Foundation<br>1.1.1. Algorithms and it’s Specification<br>1.1.2. Random Access Machine Model<br>1.1.3. Notion of best case, average case and worst case<br>1.1.4. Notion of Algorithm Correctness<br>1.2. Characterizing Run Time<br>1.2.1. Use of asymptotic notation<br>1.2.2. Big-Oh, Omega and Theta Notations<br>1.3. Analyzing Recursive Algorithms<br>1.3.1. Recurrence relations<br>1.3.2. Specifying runtime of recursive algorithms<br>1.3.3. Master Theorem<br>1.3.4 Solving Recurrences: Substitution Method, Recursion Tree Method | T1: 1.1, 1.2<br>T1:1.1.4<br>R1: 4.3,4.4,4.5 |
| M2 | **Elementary Data Structures**<br>2.1. Stacks ADT , Implementation and Applications<br>2.2. Queues ADT , Implementation and Applications<br>2.3. Amortized Analysis – Stack, Queue operations- Aggregate Method<br>2.4. List ADT , Implementation and Applications | R1:10.1<br>R1:17.1<br>R1:10.2 |

| | | |
|---|---|---|
| M3 | **Non-Linear Data Structures**<br>3.1. Trees<br>3.1.1. Terms and Definition<br>3.1.2. Tree ADT<br>3.1.3. Applications<br>3.2. Binary Trees<br>3.2.1. Properties<br>3.2.2. Representations (Array Based and Linked Structure)<br>3.2.3. Binary Tree traversal (In Order, Pre Order, Post Order)<br>3.2.4. Applications<br>3.3. Heaps<br>3.3.1. Definition and Properties<br>3.3.2. Representations (Array Based and Linked)<br>3.3.3. Insertion and deletion of elements<br>3.3.4. Heap sort<br>3.3.5. Priority Queue<br>3.4. Graphs<br>3.4.1. Terms and Definitions<br>3.4.2. Properties<br>3.4.3. Representations (Edge List, Adjacency list, Adjacency Matrix)<br>3.4.4. Graph Traversals (Depth First and Breadth First Search )<br>3.5.5. Applications<br>3.5. Directed Graph and Reachability-<br>Floyd-Warshall's Transitive Closure | T1: 2.3<br>R2:6<br>R1: 22.1,<br>22.2,22.3<br>R1:25.2 |
| M4 | **Dictionaries**<br>4.1. Dictionary ADT , Applications<br>4.2. Hash Tables<br>4.2.1. Notion of Hashing and Collision<br>4.2.2. Methods for Collision Handling<br>4.2.2.1. Separate Chaining<br>4.2.2.2. Notion of Load Factor<br>4.2.2.3. Rehashing<br>4.2.2.4. Open Addressing [ Linear & Quadratic Probing, Double Hash]<br>4.2.2.5. Applications<br>4.3. Universal Hashing<br>4.4. Introduction to Bloom Filters, Applications<br>4.5. Binary Search Tree<br>4.5.1. BST Operations<br>4.5.2. Applications<br>4.6. AVL trees<br>4.7. Rank and Range Queries, Performance<br>4.8 k-d Trees<br>4.6.1 Representation<br>4.6.2 Insertion, Deletion and Complexity | R2:11<br>Bloom Filter<br>R1: 12<br>T1:3.1<br>T1:3.2<br>T1:12.1<br>T1:12.3.2 |

| | | |
|---|---|---|
| M5 | **Algorithm Design Techniques**<br>5.1. Greedy Method<br>5.1.1. Design Principles and Strategy<br>5.1.2. Fractional Knapsack Problem<br>5.1.3. Minimum Spanning Tree<br>5.1.4. Shortest Path Problem - Djikstra's Algorithm<br>5.1.5. Task Scheduling Problem<br>5.2. Divide and Conquer<br>5.2.1. Design Principles and Strategy<br>5.2.2. Integer Multiplication Problem<br>5.2.3. Merge Sort<br>5.2.4. QuickSort<br>5.3. Dynamic Programming<br>5.3.1. Design Principles and Strategy<br>5.3.2. Matrix Chain Product Problem<br>5.3.3. All-pairs Shortest Path Problem<br>5.3.4. 0/1 Knapsack Problem | T1: 5.1, 7.3,7.1.1<br>T1: 5.2.2, 4.1,4.3<br>T1: 5.3,7.2 |
| M6 | **Complexity Classes**<br>6.1. Definition of P and NP classes and examples<br>6.2. Understanding NP-Completeness: CNF SAT<br>6.3. Cook-Levin theorem<br>6.4. Polynomial time Reducibility:<br>6.4.1 CNF SAT<br>6.4.2 Clique | T1: 13 |

## Part B: Session Plan

| | |
|---|---|
| **Academic Term** | **2022-2023 First Semester** |
| **Course Title** | **DATA STRUCTURES AND ALGORITHMS DESIGN** |
| **Course No** | **DSECCZG519** |
| **Lead Instructor** | Prof Parthasarathy |

***SESSION CONTENTS***

| **Session (#)** | **List of Topic Title (from content structure in Course Handout)** | **Text/Ref Book** |
|---|---|---|
| 1 | **Analyzing Algorithms**<br>**Theoretical Foundation**<br>• Algorithms and it's Specification<br>• Random Access Machine Model<br>• Notion of best case, average case and worst case<br>• Notion of Algorithm Correctness | T1: 1.1, 1.2 |
| 2 | **Analyzing Algorithms (Continued…)**<br>**Characterizing Run Time**<br>• Use of asymptotic notation<br>• Big-Oh, Omega and Theta Notations<br>**Analyzing Recursive Algorithms**<br>• Recurrence relations<br>• Specifying runtime of recursive algorithms<br>• Master Theorem | T1:1.1.4<br>R1: 4.3,4.4,4.5 |
| 3 | **Elementary Data Structures**<br>• Stacks ADT , Implementation and Applications<br>• Queues ADT , Implementation and Applications<br>• Amortized Analysis -Stack, Queue operations-Aggregate Method<br>• List ADT , Implementation and Applications | R1:10.1<br>R1:17.1<br>R1:10.2 |

| | | |
|---|---|---|
| 4 | **Non-Linear Data Structures**<br>**Trees**<br>• Terms and Definition<br>• Tree ADT<br>• Applications<br>**Binary Trees**<br>• Properties<br>• Representations (Array Based and Linked Structure)<br>• Binary Tree traversal (In Order, Pre Order, Post Order)<br>• Applications | T1: 2.3 |
| 5 | **Heaps**<br>• Definition and Properties<br>• Representations (Array Based and Linked)<br>• Insertion and deletion of elements<br>• Heap sort<br>• Priority Queue | R2:6 |
| 6 | **Graphs**<br>• Terms and Definitions<br>• Properties<br>• Representations (Edge List, Adjacency list, Adjacency Matrix)<br>• Graph Traversals (Depth First and Breadth First Search )<br>• Applications [Self-Study] | R1: 22.1,<br>22.2,22.3 |
| 7 | **Dictionaries**<br>• Dictionary ADT , Applications<br>• Hash Tables<br>• Notion of Hashing and Collision<br>**Methods for Collision Handling**<br>• Separate Chaining<br>• Notion of Load Factor<br>• Rehashing<br>• Open Addressing [ Linear &Quadratic Probing, Double Hash]<br>• Applications [Self-Study] | R1:25.2<br><br>R2:11 |
| 8 | **Introduction to Bloom Filters, Applications**<br>**Binary Search Tree**<br>• BST Operations<br>• Applications | R2:11 |
| 9 | **AVL Trees**<br>• Rotations, Insertion, Deletion<br>**k-d Trees**<br>• Representation<br>• Insertion, Deletion and Complexity | T1:3.1,3.2 |
| 10 | **Algorithm Design Techniques**<br>**Greedy Method**<br>• Design Principles and Strategy<br>• Fractional Knapsack Problem<br>• Minimum Spanning Tree - Kruskal | T1:12.1<br>T1:12.3.2 |
| 11. | **Greedy Method (Continued…)**<br>• Minimum Spanning Tree - Prims<br>• Shortest Path Problem - Djikstra’s Algorithm<br>**Divide and Conquer**<br>• Design Principles and Strategy<br>• Merge Sort | T1: 5.1 |

| | | |
|---|---|---|
| 12 | • Integer Multiplication Problem<br>• Quick Sort<br>**Dynamic Programming**<br>• Design Principles and Strategy<br>• Fibonacci Series | T1: 7.3,7.1.1 |
| 13 | • Matrix Chain Product Problem<br>• Directed Graph & Reachability – Warshall's Algorithm<br>• All-pairs Shortest Path Problem – Floyd's Algorithm | T1: 5.2.2, 4.1 |
| 14 | • 0/1 Knapsack<br>**Complexity Classes**<br>• Definition of P and NP classes and examples<br>• Understanding NP-Completeness: CNF SAT | T1: 5.3 |
| 15 | **Complexity Classes (Continued…)**<br>• Cook-Levin theorem<br>Polynomial time Reducibility:<br>• CNF SAT<br>• Clique | T1: 7.2<br>T1: 13 |
| 16 | Buffer Class for missed topics + Revision | T1: 13 |

***Webinar:*** Topics and date/time will be informed by the instructor via Canvas.

## *Evaluation Scheme*

Legend: EC = Evaluation Component

| No | Name | Type | Duration | Weight | Day, Date, Session, Time |
|---|---|---|---|---|---|
| EC-1 | Assignment-1 (12%)<br>Assignment-2 (13%)<br>Quiz (Best of 2 – 5%) | Will be informed | | 30% | Will be informed |
| EC-2 | Mid Term | Open Book | | 30% | |
| EC-3 | Comprehensive Exam | Open Book | | 40% | |

***Note*** - *Evaluation components can be tailored depending on the proposed model.*

## *Important Information*

Syllabus for Mid-Semester Test (Closed Book): Topics in Weeks 1-8
Syllabus for Comprehensive Exam (Open Book): All topics given in plan of study
Evaluation Guidelines:

1. EC-1 consists of two Assignments (25% weightage) and 2 Quizzes (best score of quiz will be considered – 5%). Announcements regarding the same will be made in a timely manner.
2. For Open Book exams: Use of prescribed and reference textbooks, in original (not photocopies) is permitted. Class notes/slides as reference material in filed or bound form is permitted. However, loose sheets of paper will not be allowed. Use of calculators is permitted in all exams. Laptops/Mobiles of any kind are not allowed. Exchange of any material is not allowed.
3. If a student is unable to appear for the Regular Test/Exam due to genuine exigencies, the student should follow the procedure to apply for the Make-Up Test/Exam. The genuineness of the reason for absence in the Regular Exam shall be assessed prior to giving permission to appear for the Make-up Exam. Make-Up Test/Exam will be conducted only at selected exam centers on the dates to be announced later.

It shall be the responsibility of the individual student to be regular in maintaining the self-study schedule as given in the course handout, attend the lectures, and take all the prescribed evaluation components such as Assignment/Quiz, Mid-Semester Test and Comprehensive Exam according to the evaluation scheme provided in the handout.

********** ALL THE BEST **********