Suggested Teaching Guidelines for

Algorithm and Data Structures – PG-DAC February 2020

Duration: 34 class room hours + 36 lab hours (7**0hrs**)

Objective: To reinforce knowledge of Problem solving techniques, Data Structure Concepts and analysis of different Algorithm.

Prerequisites: Knowledge of Programming in C/C++ with Object Oriented concepts

Evaluation method: Theory exam– 40% weightage
Lab exam – 40% weightage
Internal exam– 20% weightage

List of Books / Other training material

Test Book:

1. Fundamentals of Data Structures in C++ by Horowitz, Sahani & Mehata / Orient Longman

Reference:

1. Problem Solving: Best Strategies to Decision Making, Critical Thinking and Positive Thinking by Thomas Richards / Kindle Edition
2. Data Structures, Algorithms and Applications in C++ by Sartaj Sahni
3. Object-oriented Analysis And Design Using Umlan Introduction To Unified Process And Design Patterns 1st Edition by Mahesh P. Matha / PHI
4. Introduction to Algorithms by Cormen, Leiserson, Rivest and Stein
5. Data Structures Via C++ Objects by Evolution by A Michael Berman / Oxford University Press
6. Design and Analysis of Algorithm by S Sridhar / Oxford University Press
7. Fundamentals of Computer Algorithms 2nd Edition by Sanguthevar Rajasekaran, Sartaj Sahni, Ellis Horowitz / Orient Longman
8. Introduction to Algorithms by Al. Cormen / PHI Learning
9. The Algorithm Design Manual by Steven S. Skiena / Springer
10. Algorithmic Puzzles by Anany Levitin, Maria Levitin / Oxford University Press

Session 1 & 2: Introduction to STL

Lecture

- Object Design and Templates
- STL (Standard Type Libraries)

Assignment – Lab:

- Find the number of students who are passes or failed using MAP.
- Find the prime numbers from 2 to n using sieves algorithm, use SET

Session 3 & 4: Problem Solving & Computational Thinking

Lecture

- Define the problem
    - Identify the problem

Suggested Teaching Guidelines for
Algorithm and Data Structures – PG-DAC February 2020

  - Introduction to Problem Solving
  - Problem solving basics
  - Defining creativity v/s innovation
- Find Creative Solutions using creativity tools
  - Effective problem solving approaches
  - Critical thinking and information analysis
  - Brainstorming, Reverse Brainstorming, Imagineering, Mind Mapping,
  - Six Thinking Hats: A Tool to Strengthen Critical Thinking, Collaboration, Communication, and Creativity Skills
  - Analyzing the situation, Gathering information, Identifying solution criteria
  - Decision Making Methods
  - Charts and Diagrams
  - Applying outcome-based thinking
  -
- Evaluate and Select solution
  - Pro's and Con's, Force field analysis, Feasibility/Capability Analysis,
  - Decision analysis, evaluating problems
  - Choosing among alternatives
  - Qualitative analysis, discussing qualitative analysis techniques
  - Establishing objectives
  - Assigning weight to objectives in order to make the best decision
  - Creating a satisfaction scale to choose between alternatives
- Implementing Decisions
  - Create an action plan
  - Break solution into action steps
  - Prioritize actions and assign roles (setting priorities for taking action)
  - Follow-up at milestones

Assignment – Lab:

- Faculties need to assign different problems, mostly real world problems
- Students (by team wise, there are two students in a team) need to analyze as per the techniques learned
- Students need to solve by the thinking approaches learned.
- Based on the above problems students need to select as per the selection criteria learned
- They need to implement the selected solution and need to do the documentations.

Session 5 & 6: Algorithm design
Lecture

- How to write efficient Algorithm
- Introduction to algorithm design techniques
- Algorithm Design techniques
- Analysis of an Algorithm
  - Asymptotic analysis
  - Algorithm analysis
- Analysis of different type of Algorithms

  - Divide and Conquer Algorithm
  - Greedy Algorithm
  - Dynamic Programming Algorithm
  - Brute force Algorithm
  - Backtracking algorithms
  - Branch-and-bound algorithms
  - Stochastic algorithms
- Complexity
  - Complexity Analysis
  - Space complexity of algorithm
  - Time complexity of algorithm
- Case study on Algorithm Design techniques
- Application of Data structures

Assignment – Read:

- Study on different Algorithms
- Compare different Algorithms previously programmed and do the analysis

Session 7 & 8: Algorithm & Data Structures

Lecture:

- Introductory Concepts
- Algorithm Constructs
- OO design: Abstract Data Types (ADTs)
- Basic Data Structures
  - Arrays
  - Stacks
  - Queues
  - Circular Queues
  - Priority Queues
  - Deques

Assignment – Lab:

- Implement Stack through Array
- Implement C-Stack, C2-Stack and CN-Stack in same memory block.
- Implement Queues with inserting element at different location (First, Last and at specific location)
- Implement circular queue, Priority Queues and Dqueue
- Implement program to convert infix expression into postfix expression & evaluate postfix expression.

Session 9 & 10: Linked List Data Structures

Lecture

- Linked lists
  - Single Linked Lists
  - Double Linked Lists
  - Circular Linked Lists

- Node-based storage with arrays

Assignment – Lab:

- Implement circular queue using linked list
- Design an iterator using circular linked list

Session 11 & 12: Trees & Applications

Lecture

- Introduction to trees
- Trees and Terminology
- Tree traversals
- Ordered trees
- Binary trees
- Complete binary trees
- Search trees
- Binary search trees
- Introduction to self balancing tree & variants

Assignment – Lab:

- Write a program to implement a binary search tree and the following operations on it:
  - Create()
  - InsertNode()
  - Tree traversals (Inorder(), Preorder(), Postorder())
  - deleteNode()
- Design a threaded binary tree and implement the orders.

Session 13 & 14: Searching & Sorting algorithms

Lecture

- Objectives of Searching
  - The Sequential Search
  - Analysis of Sequential Search
  - The Binary Search
- Analysis of Binary Search
- Introduction to sorting
  - Selection sort
  - Insertion sort
  - Bubble sort
  - Heap sort
  - Merge sort
  - Quick sort
- Analysis of sorting algorithms

Assignment – Lab:

- Writing program to search an item through sequential search technique.
- Implement to find an item in a list through binary search
- Implement sorting algorithm for selection sort, Bubble sort, heap sort and quick sort

- Write a program to merge two sorted linked lists

Session 15: Hash functions and hash tables
Lecture

- Hashing & Introduction to hash tables
- Hash functions
- Mapping down to 0 ... M − 1
- Chained hash tables
- Scatter tables
- Open addressing
- Linear probing
- Quadratic probing
- Double hashing
- Poisson distribution
- Collision Resolution
- Analysis of Hashing

Assignment – Lab:

- Implement hashing techniques in different programs solved earlier
- Implement collision and solution to it on any previous solved problem
- Write a program to implement Hash table

Session 16 & 17: Graph & Applications
Lecture

- Introduction to graph theory
- Graph Terminology
- Different types of Graphs
- Representation of Graphs
    - Connectedness, Single source un-weighted path length, identifying bipartite graphs
    - Graph Traversal Algorithms (Breadth First Search, Depth First Search)
    - Single-source shortest path algorithms, Dijkstra's algorithm, A* search algorithm, Bellman-Ford algorithm
    - All-pairs shortest path, Floyd-Warshall algorithm, Johnson's algorithm
    - Maximum flow algorithms, Ford-Fulkerson algorithms
- Spanning Trees
    - Minimum spanning tree algorithms, Prim's algorithm, Kruskal's algorithm

Assignment – Lab:

- Implement a graph using adjacency links and traverse using Depth First Search.
- Write a program using STL to implement Dijkstra's Shortest Path Algorithm.