**­­­**



**University of Central Punjab**

**Faculty of Information Technology**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Course** | BSCS and BSSE | | | |
| **Course Title** | Data Structures and Algorithms | | | |
| **Credit Hours** | 4 (3 + 1) | | | |
| **Prerequisites** | Object Oriented Programming | | | |
| **Assessment Instruments with Weights** (homework, quizzes, midterms, final, programming assignments, lab work, etc.) | **Theory (3 credit hours)** | | | |
| Quiz (4-6) | | 15% | |
| Assignments (4) | | 10% | |
| Class Activity (2-4) | | 10% | |
| Midterm | | 25% | |
| Final Term | | 40% | |
| **Lab (1 credit hour)** | | | |
| Graded Laboratory sessions (6) | | 30% | |
| Midterm | | 30% | |
| Final | | 40% | |
| **Textbook** | **Text Book:**   * D. S. Malik, Data Structures Using C++, Cengage Learning   **Reference:**   * Nell Dale, C++ plus Data Structures, Jones & Bartlet Learning * Y. Langsam, M. J. Augenstein, A. M. Tenenbaum, Data Structures Using C and C++, Prentice-Hall * Mark Allen Weiss, Data Structures and Algorithm Analysis in C++, Second/Third Edition, Edition, Addison Wesley. * V. Aho, J. E. Hopcroft, J. D. Ullman, Data Structures and Algorithms, Addison-Wesley. * Thomas H. Cormen et al, Introduction to Algorithms, Prentice-Hall * Adam Drozdek, Data Structures and Algorithms in C++, Broooks/Cole | | | |
| **Current Catalog Description** | Data structures allow one to efficiently manage large amounts of data, which is key to designing an efficient algorithm, such as an internet indexing service. Ultimately data structures are a way of organizing and storing data so it can be used more efficiently. This course familiarizes students with concepts of creating, storing, retrieving, ordering, and manipulation of data structures and the basics of analysis of algorithms.  The students will learn formal specification of data structures in depth. | | | |
| **Course Objectives** | After following this course, students will be able to:   1. Assess how the choice of data structures and algorithm design methods impacts the performance of programs. 2. Choose the appropriate data structure and algorithm design method for a specified application. 3. Write programs using object-oriented design principles. 4. Solve problems using data structures such as linear lists, stacks, queues, hash tables, binary trees, heaps, tournament trees, binary search trees, and graphs and writing programs for these solutions. 5. Solve problems using algorithm design methods such as the greedy method, divide and conquer, dynamic programming, backtracking, and branch and bound and writing programs for these solutions. | | | |
| **Course Learning Outcomes (CLO)** | On completion of this module, students should be able to:   1. Understand the properties of various data structures. 2. Identify the strengths and weaknesses of different data structures. 3. Design and employ appropriate data structures for solving computing problems 4. Possess the knowledge of various existing algorithms. 5. Analyze and compare the efficiency of algorithms. | | | |
| **Class Time Spent**  (in credit hours) | **Theory** | **Problem Design and Analysis** | | **Laboratory Sessions** |
| 2 | 1 | | 1 |

|  |  |  |
| --- | --- | --- |
| **Theory Course Outline** | | |
| **Lecture** | **Topics** | **Assessments** |
| **1** | * Course Introduction * Revision of previous concepts * Data Structures using Object Oriented Paradigm   + Abstract classes   + Pure virtual functions   + Template classes and functions |  |
| **2** | * Introduction to Time Complexity (Big O): O(1), O(N), O(N2), O(log2N) * Binary Search: explanation of O(log2N) |  |
| **3** | * Binary Search Code * Array Data Structure | Assignment 1  Quiz 1 |
| **4** | * Stack Data Structure * Postfix, Infix, Prefix Notations * Applications of Stack |  |
| **5** | * Queue * Circular Queue |  |
| **6** | * Arrays vs Linked Lists * Singly Linked List using head and tail * Stack and Queue using Singly Linked List * Singly Linked List using head only (insertion of data in sorted order) | Assignment 2  Quiz 2 |
| **7** | * Singly Circular Linked list using tail only * Doubly Linked Lists (using head only, tail only, and both) * Doubly Circular Linked Lists (using head only, tail only, and both) | Quiz 3 |
| **8** | * Revision |  |
| **Midterm Exam** | | |
| **9** | * Recursion |  |
| **10** | * Binary Trees * Binary Search Trees |  |
| **11** | * In-order, post-order, and pre-order traversals * Binary Search Tree Conclusion | Assignment 3  Quiz 4 |
| **12** | * Self-Balancing Trees (AVL or Red-Black): Insertion only * Self-Balancing Trees conclusion |  |
| **13** | * Hash Tables Introduction * Hash Tables: Separate Chaining, Open Addressing, Linear Probing, Quadratic Probing etc. | Quiz 5 |
| **14** | * Hash Tables: Unordered Map * Heaps (Min and Max - using arrays only) * Heap Sort | Assignment 4  Quiz 6 |
| **15** | * Huffman Encoding * Introduction to graphs: adjacency matrix and adjacency lists * BFS, DFS |  |
| **16** | * Revision |  |
| **Final Exam** | | |

**The lab and theory teachers must ensure that the lectures and lab sessions are in sync.**

|  |  |  |
| --- | --- | --- |
| **Laboratory Outline** | | |
| **Lab** | **Topic** | **Objective** |
| **1** | Revision | * Array revision * Simple sorting algorithms — Selection sort * Simple searching algorithms — Linear search, binary search |
| **2** | Basic Data Structure - Arrays | * Basic concepts of 2-D array * Dynamically allocation of 2-D array * Use of pointers in context of Arrays * Application of 2-D arrays in Mathematics and basic arithmetic operations on Matrices |
| **3** | Stacks | Working on the Stack ADT |
| **4** | Application of Stacks | Working on Applications of Stack - Infix, Postfix and Prefix notations |
| **5** | Queues | Working on the Queue ADT |
| **6** | Applications of Queues | Working on Applications of Queues – Priority Queues (Bank, Motorway e-tag) |
| **7** | Linked Lists | Implementation of Linked Lists ADT |
| **8** | Doubly and Doubly Circular Linked Lists | * Implementation of Doubly and Doubly Circular Linked Lists * Implementation of Stack and Queues using linked lists |
| **9** | Recursion | Tower of Hanoi, basic problems (factorial, Fibonacci, decimal to binary, number of digits in an integer) |
| **10** | Binary Trees | Implementation of Binary Search Trees – different traversals, height of the tree, searching |
| **11** | Binary Trees | Applications of BST |
| **12** | Self-Balancing Trees | Implantation of AVL or R-B trees |
| **13** | Hash Tables | Applications using unordered maps |
| **14** | Hash Tables | Applications using unordered maps continued |
| **15** | Heaps | Implementation of Min and Max Heaps |
| **16** | Revision | Revision Lab |