Resource Person:

Module:

Course Title: Data and File Structures

Course Code:

Course Type: Programming

Pre-Requisite: Programming Fundamentals

Counseling Hours:

Program Head: Mr. Imran Saleem

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Name** | **Signature** | **Date** |
| **Prepared By**  (Resource Person) |  |  |  |
| **Checked By**  (Program Head) |  |  |  |
| **Approved By**  (Director SPA) |  |  |  |

**Course Description& Format**

Data structures is considered as a core course in the Computer Science Curriculum. They are the basically the building blocks of computer software. The course is designed to teach students some of the basic data structures, and algorithms. Students will learn the fundamental techniques of data representation, organization, storage, searching, sorting, retrieval, and manipulation. The course will cover well-known data structures such as dynamic arrays, linked lists, stacks, queues, tree, heap, disjoint sets and table. Students will also be introduced to the notions of time and space complexities so that the students can appreciate the time and space tradeoffs.

The basic concepts, implementations, performance, and applications of the various data structures and algorithms will be integrated throughout the course. Students shall gain hands-on experience through several programming exercises. The programming language that will be followed in the course is C++.

**Course Instructional Objectives**

* Learning of different data structures
* Understanding the abstract data types.
* Learning and implementing the operations of stacks, queues and linked list.
* Understand the behavior and functionalities of different data structure.
* Learn about major issues of performance and reliability of different types of structures.

**Course Student Objectives**

* Good analytical skills and systematic program solving capabilities
* Inquisitive mind to undertake research for solving engineering problems
* Scientific skills to manage the human and technological resources optimally
* Effective decision making skills
* Develop students’ skills in analyzing data structures.
* Build up students’ capacity to evaluate different algorithmic techniques.
* Build up students’ capacity to write programs for developing simple applications.

**GENERAL INSTRUCTIONS:**

**Attendance:**

It is expected that every student will be in class for lectures. Attendance records will be kept and used to determine each person’s qualification to sit for the final examination. In case of illness or other unavoidable cause of absence, the student must communicate as soon as possible with any of the instructors, indicating the reason for the absence.

**Academic Integrity:**

Violations of academic integrity, including dishonesty in assignments, examinations, or other academic performances are prohibited. You are not allowed to make copies of another person’s work and submit it as your own; that is plagiarism. All cases of academic dishonesty will be reported to the University Management for appropriate sanctions in accordance with the guidelines for handling students’ misconduct as spelt out in the Students’ Handbook.

**Assignments and Group Work:**

Students are expected to submit assignments as scheduled. Failure to submit an assignment as at when due will earn you zero for that assignment. Only under extenuating circumstances, for which a student has notified any of the instructors in advance, will late submission of assignments be permitted.

**Code of Conduct in Lecture Rooms and Laboratories:**

Students should turn off their cell phones during lectures. Students are prohibited from engaging in other activities (such as texting, watching videos, *etc*.) during lectures. Food and drinks are not permitted in the laboratories.

**Course Contents:** Following is the session-wise breakup of the course:

**Session 1: Introduction to Data Structures**

* One-to-one introduction
* Course Introduction, Teaching & Assessment Methodology
* Distribution of Course Outlines
* Discussion on Course Outline
* Setting up of Norms
* A brief introduction to Data Structures.

**Learning Outcomes**

The main learning objective of the reading is to provide a basic and conceptual understanding of different data structures.

**Activities**

* Discussion

**Session 2: Data Types and Recursion**

* Introduction to Data Types and Abstract Data Types
* List of ADT operations
* Array, Classification, Application of array
* Recursion
* Introduction to Code Blocks and Visual Studio software

**Learning Outcomes**

The main learning objective of the reading is to provide a basic and conceptual understanding of different types (Language Defined and User Defined). Learning of Recursive functions and implementation of Array operations.

**Activities**

* Home Task for Implementation of operations on array.

**Session 3: Lists**

* Introduction to Linked List
* C++ code for Linked List
* Behavior of List by using Array
* Performance
* Implementation Details of Single Link List

**Learning Outcomes**

The main learning objective of these sessions will be to understand basics of making linked lists. How memory can be assigned at run time. How the lists shrink and grow.

**Activities**

* Discussions
* Lab Tasks with (Sample code) for building Linked List

**Session 4: Lists Continue**

* C++ code for Circular Link List
* C++ code for Doubly Linked List
* Josephus problem
* Performance and efficiency
* Implementation

**Learning Outcomes**

The main learning objective of these sessions will be to understand core Operations of Doubly and Circular linked lists. How memory can be assigned at run time. How the lists shrink and grow.

**Activities**

* Hands on implementation of Circular & Double Linked list using Code Blocks and Visual Studio.
* Home Task Given (A Problem statement for Implementation of Complete Linked List)

**Session 5: Stack and Queue**

* Stacks, Stack Operations
* Static and dynamic Stack, Applications of Stack
* Implementation with arrays\linked list
* Run-Time Memory Organization and Stack Layout
* Introduction to Queues
* Queues Operation
* Storing queue in static and dynamic Data structure

**Learning Outcomes**

The main learning objective of this session is to make students familiar with dynamic data structures such as Stacks and Queues and their memory organization.

**Activities**

* Quiz 1
* Home Task 2 submission and discussion.

**Session 6: Queues and Trees**

* Queues Implementation using Linked list and Circular Arrays
* Application of Queues
* Priority Queues
* Introduction to Trees
* Different types of Trees and their respective behaviors
* Introduction to Binary Search Trees (BST).
* Application and Implementation of BST

**Learning Outcomes**

The students will learn about the implementation of queues and trees. Similarly how different operations can be performed on trees.

**Activities**

* Application of Tree Data structure in computer Networking
* Discussions with some examples

**Session 7: Trees Continued**

* Traversing in Binary Trees (Inorder, Preorder, Postorder)
* Level order traversing
* C++ code for deleting a node in a tree
* Different Algorithms of Trees
* Functions related to performing different functions in a tree.
* Application of Trees

**Learning Outcomes**

This session will enable the students to understand the concepts of tree and its implementation details.

**Activities**

* Quiz 2
* Home task 3 submission with Viva

**Session 8: Midterm Exam**

**Session 9: AVL Trees**

* Introduction to AVL Trees
* Inserting in an AVL Tree
* Inserting in a balanced BST, tree rotation for height balancing
* Various rotation cases

**Learning Outcomes**

Students will learn different methods of balancing a binary search tree.

**Activities**

* Implementation of AVL Trees
* Lab hand on with sample code

**Session 10: AVL Trees and Hashing**

* Different concepts of Hashing
* Open and Closed Hashing
* Chaining
* Double Hashing

**Learning Outcomes**

The students will learn how to quickly access the data stored in different memory locations. How it can be accessed

**Activities**

* Return of midterm exams and a discussion of the solution.

**Session 11: Hashing and Priority Queues**

* Implementation details of Hashing
* Concept of Priority Queues
* Implementation Details

**Learning Outcomes**

Students will be able to learn implementation details of hashing and priority queues.

**Activities**

* Quiz 3
* Discussions with Lab hands on.

**Session 12: Sorting Algorithms**

* Bubble Sort
* Insertion Sort
* Selection Sort
* Shell Sort

**Learning Outcomes**

The students will learn about different sorting techniques

**Activities**

* Implementation of Sorting Algorithms
* Home Task 4 (Practice exercises for Sorting Algorithms)

**Session 13: Sorting Algorithms continued**

* Quick Sort
* Merge Sort
* Heap Sort
* Radix Sort

**Learning Outcomes**

The students will learn about efficiency of different sorting algorithms.

**Activities**

* Quiz 4
* Home task 4 Submission and Discussion

**Session 14: Graphs: Preliminaries and Representations**

* Graph Traversing
* Topological sorting
* Shortest-path algorithms and Cycle Detection
* Isomorphic Graph
* Network flow problems

**Learning Outcomes**

In this session the students will learn the basics of graphs and how it is represented.

**Activities**

* Presentations on Provided Topics

**Session 15: Graphs and Final Review**

* Spanning trees
* Splay Trees
* Wrap-up and course summary

**Learning Outcomes**

In this session the students will learn about the newest and emerging technologies related to storage and memory systems.

**Activities**

* Remaining presentations (If Left)

**Session 16: Final Term**

**Recommended Book (s):**

**Textbook:**

M. A. Weiss. Data Structures and Algorithm Analysis in C++. Pearson Education, fourth edition, 2014

**Reference Book:**

Any good C programming book such as Kernighan and Ritchie’s The C Programming Language, Deitel and Deitel’s C How to Program or Herbert Schildt’s C The Complete Reference will be helpful for the programming assignments

**ASSESSMENT METHODOLOGY**

|  |  |
| --- | --- |
| **Distribution** | |
| Home Tasks / Assignments | 10% |
| Presentations | 10% |
| Lab Tasks | 10% |
| Quizzes | 15% |
| Mid Term | 20% |
| Final Exam | 35% |
| **Total** | **100%** |

Assignments will be individual or some in group of two as per Requirement of the instructor.

**Calendar of Activities**

|  |  |  |  |
| --- | --- | --- | --- |
| **Session** | **Topic** | **Readings** | **Activities** |
| 1 | **Session 1: Introduction to Data Structures**   * One-to-one introduction * Course Introduction, Teaching & Assessment Methodology * Distribution of Course Outlines * Discussion on Course Outline * Setting up of Norms * A brief introduction to Data Structures. | Book1 Chapter 1 | Discussion. |
| 2 | **Data Types and Recursion**   * Introduction to Data Types and Abstract Data Types * List of ADT operations * Array, Classification, Application of array * Recursion * Introduction to Code Blocks and Visual Studio | Book 1 Chapter 2 | 1st Home Task for Implementation of operations on array. |
| 3 | **Lists**   * Introduction to Linked List * C++ code for Linked List * Behavior of List by using Array * Performance * Implementation Details of Single Link List | Book 1 Chapter 3 | Discussions  Lab Tasks with (Sample code) for building Linked List |
| 4 | **Lists Continue**   * C++ code for Circular Link List * C++ code for Doubly Linked List * Josephus problem * Performance and efficiency * Implementation | Book 1, Chapter 3 | Implementation of Circular & Double Linked list.  2nd Home Task Given |
| 5 | **Stack and Queue**   * Stacks, Stack Operations * Static and dynamic Stack, Applications of Stack * Implementation with arrays\linked list * Run-Time Memory Organization and Stack Layout * Introduction to Queues * Queues Operation * Storing queue in static and dynamic Data structure | Book 1, Chapter 3 | Quiz 1  Home Task 2 submission and discussion. |
| 6 | **Queues and Trees**   * Queues Implementation using Linked list and Circular Arrays * Application of Queues * Priority Queues * Introduction to Trees * Different types of Trees and their respective behaviors * Introduction to Binary Search Trees (BST). * Application and Implementation of BST | Book 1 Chapter 4 | Application of Tree  Discussions with some examples |
| 7 | **Trees Continued**   * Traversing in Binary Trees (Inorder, Preorder, Postorder) * Level order traversing * C++ code for deleting a node in a tree * Different Algorithms of Trees * Functions related to performing different functions in a tree. * Application of Trees | Book1 Chapter 4 | Quiz 2  Home task 3 |
| 8 | **MIDTERM** | | |
| 9 | **AVL Trees**   * Introduction to AVL Trees * Inserting in an AVL Tree * Inserting in a balanced BST, tree rotation for height balancing * Various rotation cases | Book1 Chapter 4 | Implementation of AVL Trees  Lab Practice |
| 10 | **AVL Trees and Hashing**   * Different concepts of Hashing * Open and Closed Hashing * Chaining * Double Hashing | Book1 Chapter 4, 5 | Midterm Discussed |
| 11 | **Hashing and Priority Queues**   * Implementation details of Hashing * Concept of Priority Queues * Implementation Details | Book1 chapter 5, 6 | Quiz 3  Discussions with Lab practice |
| 12 | **Sorting Algorithms**   * Bubble Sort * Insertion Sort * Selection Sort * Shell Sort | Book 2 chapter 7 | Implementation of Sorting Algorithms  Home Task 4 |
| 13 | **Sorting Algorithms continued**   * Quick Sort * Merge Sort * Heap Sort * Radix Sort | Book1 chapter 7 | Quiz 4  Home task 4 Submission and Discussion |
| 14 | **Graphs: Preliminaries and Representations**   * Graph Traversing * Topological sorting * Shortest-path algorithms and Cycle Detection * Isomorphic Graph * Network flow problems | Book1 Chapter 9 | Presentations on Provided Topics |
| 15 | **Graphs and Final Review**   * Spanning trees * Splay Trees * Wrap-up and course summary | Book1 Chapter 9 and Selected topics | Presentations continued |
| 16 | **FINALTERM** | | |