K.R. MANGALAM UNIVERSITY

THE COMPLETE WORLD OF EDUCATION

Department: CSE

Session: 2025-26

Programme: BTech, BCA

Semester: 3rd

Course Code: ENCS205, ENCA 201

Course: Data Structures

Assignment Details

Theme: Linear Data Structures

Assignment Number: 02

Total Marks: 5

Contribution: 16% of internal evaluation

CO Mapping: CO 1, CO 3

Submission Requirements:

- Submit individually via GitHub and provide the link of submission by **30th September 2025**.
- Late submissions, copied assignments, or AI-generated text will not be accepted.
- Use the dedicated assignment format shared in classes.

Problem: Develop the Hospital Patient Record Management System

Description: The Hospital Patient Record Management System is a software solution designed to efficiently manage patient records, emergency queues, billing calculations, and inventory operations in a hospital setting. This system leverages linear data structures such as linked lists, stacks, and queues to dynamically handle patient admissions, prioritize emergency cases, track changes, and perform calculations for billing and inventory management. The assignment focuses on implementing these data structures to address real-world hospital management scenarios.

Implementation Sub-Problems

1. Use linked lists to manage admitted patient records dynamically.
2. Implement a stack-based undo mechanism for recent admission changes.
3. Use queues to manage an emergency patient queue using circular and priority queues.

4. Perform polynomial operations to calculate patient billing based on treatment duration and services.
5. Demonstrate postfix expression evaluation for hospital inventory calculations.

Evaluation Metrics

Metric	Marks	Excellent	Very Good	Satisfactory	Poor
Problem Understanding	1	1	0.75	0.5	0.25
Problem Solving Approach	1	1	0.75	0.5	0.25
Completion of the Problem	1	1	0.75	0.5	0.25
Participation during Class/Lab	1	1	0.75	0.5	0.25
Explanation of the Assignment	0.5	0.5	0.4	0.25	0.1
Attendance	0.5	0.5	0.4	0.25	0.1

Assignment Objectives

a) Develop a foundational understanding of C++/Python programming and linear data structures.
b) Gain practical experience with linked lists, stacks, and queues.
c) Implement real-world applications for patient record and queue management.
d) Understand the application of stacks and queues in undo mechanisms and priority-based systems.
e) Apply polynomial operations and postfix expression evaluation for computational tasks.

Assignment Instructions

1. Patient Record ADT Design

Attributes:

a) **PatientID**: Integer, unique identifier for the patient.
b) **PatientName**: String, name of the patient.
c) **AdmissionDate**: String or custom structure (e.g., day/month/year).
d) **TreatmentDetails**: String, details of treatments or services provided.

Methods:

a) **insertPatient(data)**: Insert a new patient record into the linked list.
b) **deletePatient(PatientID)**: Remove a patient record based on the PatientID.
c) **retrievePatient(PatientID)**: Retrieve patient details for a given PatientID.

2. Patient Management System

Attributes:

a) **LinkedList**: A linked list to store patient records dynamically.
b) **UndoStack**: A stack to store recent admission changes for undo functionality.
c) **EmergencyQueue**: A queue (circular or priority) to manage emergency patients.

Methods:

a) **addPatientRecord()**: Add a new patient record to the linked list and update the undo stack.
b) **undoAdmission()**: Revert the most recent admission change using the stack.
c) **manageEmergencyQueue()**: Enqueue and dequeue patients in the emergency queue using circular or priority queue logic.
d) **calculateBilling()**: Perform polynomial operations to compute billing based on treatment duration and services.
e) **evaluateInventory()**: Use postfix expression evaluation to calculate hospital inventory requirements.

Implementation Steps

1. Define the Patient Record ADT with the specified attributes and methods using a linked list structure.
2. Implement a stack-based undo mechanism to track and revert patient admission changes.
3. Develop circular and priority queue implementations to manage the emergency patient queue.
4. Implement polynomial operations for billing calculations based on treatment duration and services.
5. Demonstrate postfix expression evaluation for inventory calculations, ensuring accurate computation.

Evaluation Criteria

Criteria	**Marks**	**Description**
Problem Understanding	1	Demonstrates clear understanding of the problem requirements and objectives.
Problem Solving Approach	1	Effective approach to designing and implementing linked lists, stacks, and queues.
Completion of the Problem	1	Complete implementation of all required components (linked lists, stacks, queues, polynomial operations, postfix evaluation).
Participation during Class/Lab	0.5	Active engagement in class/lab discussions and activities related to the assignment.
Explanation of the Assignment	1	Clear and detailed explanation of the implemented solution in the submission report.
Attendance	0.5	Consistent attendance in classes/labs relevant to the assignment.

Prerequisites

- Knowledge of linked lists, stacks, and queues.
- Basic understanding of queueing logic.

Learning Outcomes

1. Apply linear data structures in practical queue and patient management scenarios.
2. Understand the use of stacks and queues in real-world problems like undo mechanisms and priority-based systems.
3. Gain proficiency in implementing polynomial operations and postfix expression evaluation for computational tasks.

Submission Guidelines

1. Upload the assignment on GitHub and provide the link of submission by the specified deadline.
2. Ensure code is original and follows the provided format.
3. Include a detailed report covering:

   a) Description of the Patient Record ADT.
   b) Strategy for implementing linked lists, stacks, and queues.
   c) Approach to polynomial operations and postfix expression evaluation.
   d) Analysis of the system's efficiency and functionality.

Helpful References

1. **Books**:
   a."Data Structures and Algorithm Analysis in C++" by Mark Allen Weiss
   b."Python Data Structures and Algorithms" by Benjamin Baka
2. **Tutorials**:
   a. GeeksforGeeks: Linked List | Set 1 (Introduction)
   b. GeeksforGeeks: Stack Data Structure