**Exercise 5: Task Management System**

Linked List Types:

- Singly Linked List: Nodes have a reference to the next node only.

- Doubly Linked List: Nodes have references to both next and previous nodes.

Linked List vs Arrays:

- Linked Lists are dynamic in size; arrays are static.

- Insertion/deletion is faster in linked lists (O(1) if at head), but slower access (O(n)).

- Arrays allow O(1) access via index; linked lists do not.

public class TaskManagementSystem {

static class Task {

int taskId;

String taskName;

String status;

public Task(int taskId, String taskName, String status) {

this.taskId = taskId;

this.taskName = taskName;

this.status = status;

}

@Override

public String toString() {

return "[" + taskId + "] " + taskName + " (" + status + ")";

}

}

static class Node {

Task task;

Node next;

public Node(Task task) {

this.task = task;

this.next = null;

}

}

static class TaskLinkedList {

Node head;

// Add task to end (Time Complexity: O(n))

public void addTask(Task task) {

Node newNode = new Node(task);

if (head == null) {

head = newNode;

return;

}

Node current = head;

while (current.next != null) {

current = current.next;

}

current.next = newNode;

}

// Traverse and display tasks (Time Complexity: O(n))

public void traverseTasks() {

if (head == null) {

System.out.println("No tasks available.");

return;

}

Node current = head;

while (current != null) {

System.out.println(current.task);

current = current.next;

}

}

// Search task by taskId (Time Complexity: O(n))

public Task searchTask(int taskId) {

Node current = head;

while (current != null) {

if (current.task.taskId == taskId) {

return current.task;

}

current = current.next;

}

return null;

}

// Delete task by taskId (Time Complexity: O(n))

public boolean deleteTask(int taskId) {

if (head == null) return false;

if (head.task.taskId == taskId) {

head = head.next;

return true;

}

Node current = head;

while (current.next != null && current.next.task.taskId != taskId) {

current = current.next;

}

if (current.next == null) return false;

current.next = current.next.next;

return true;

}

}

public static void main(String[] args) {

TaskLinkedList taskList = new TaskLinkedList();

taskList.addTask(new Task(1, "Design database schema", "Pending"));

taskList.addTask(new Task(2, "Implement login feature", "In Progress"));

taskList.addTask(new Task(3, "Write unit tests", "Pending"));

System.out.println("All Tasks:");

taskList.traverseTasks();

System.out.println("\nSearching for Task ID 2:");

Task foundTask = taskList.searchTask(2);

System.out.println(foundTask != null ? foundTask : "Task not found");

System.out.println("\nDeleting Task ID 1...");

boolean isDeleted = taskList.deleteTask(1);

System.out.println(isDeleted ? "Deleted successfully." : "Task not found.");

System.out.println("\nAll Tasks After Deletion:");

taskList.traverseTasks();

}

}

