INSTITUTE OF TECHNOLOGY

COMPUTER SCIENCE AND ENGINEERING PROGRAMME (2023-26)

Data Structures

(2CS501)

Assignment - II

Submitted to: Prof. - Jitali Patel

On: 18-12-2023

Submitted By: Dhruvi Tanna(23BCE507)

TASK MANAGEMENT SYSTEM

Introduction:

An advanced task management system is a sophisticated software solution designed to efficiently organize, track, and manage tasks within an organization or for personal use. It goes beyond basic to-do lists and offers a comprehensive set of features to enhance productivity, collaboration, and project management.

Assignment Definition:

In this assignment, you are tasked with creating an advanced task management system, allowing users to efficiently manage their tasks, deadlines, dependencies, and priorities. However, the catch is that this system is designed using multiple data structures without explicitly revealing which ones to use. Your challenge is to identify and implement the appropriate data structures for each aspect of the system.

Requirements:

1. Task Storage: Create a mechanism to store tasks efficiently. Tasks have attributes such as a unique ID, title, description, deadline, priority, and dependencies on other tasks.
2. Dependency Management: Allow users to specify dependencies between tasks, ensuring that a task with dependencies cannot be marked as complete until all its prerequisites are completed.
3. Priority Queues: Implement a priority queue to manage task priorities, ensuring that users can efficiently retrieve and complete the highest-priority tasks first.
4. Deadlines: Implement a data structure to efficiently track and manage task deadlines, providing timely notifications and ensuring that overdue tasks are easily identified.
5. Graph Representation: Model the task dependencies as a graph structure. Identify the appropriate data structure for storing the task graph, and implement algorithms to efficiently traverse and validate it.
6. User Interface: Create a user-friendly interface to interact with the task management system, allowing users to add, modify, complete, and view tasks seamlessly.

Data Structure Used:

1. HASH-MAP:

   In Java, a HashMap is a part of the Java Collections Framework and is used to store key-value pairs. It provides efficient retrieval and storage of elements based on the keys.

   Time Complexity :

   The time complexity for inserting a key-value pair into a HashMap is usually O(1) on average. However, in rare cases (due to hash collisions), the time complexity for insertion can degrade to O(n)

   Space Complexity:

   The space complexity of a HashMap in Java is O(n), where n is the number of key-value pairs stored in the map. The space complexity analysis includes both the space required for the keys and the space required for the values.

2. PRIORITY QUEUE:

   In Java, a PriorityQueue is part of the Java Collections Framework and is used to represent a priority queue. A priority queue is a data structure where elements are retrieved based on their priority. The element with the highest (or lowest, depending on the implementation) priority is served before others.

   Time Complexity :

   The time complexity for an element into a binary heap (priority queue) is logarithmic in the number of elements. This is because the element needs to be added to the appropriate position in the heap, and the height of a binary heap is log(n), where n is the number of elements.

   Space Complexity:

   The space complexity of a binary heap is linear in the number of elements. which is O(n) in terms of the number of elements.

3. ARRAY LIST:

In Java, ArrayList is a part of the Java Collections Framework and is an implementation of a dynamic array.

Time Complexity:

Inserting an element at a specific index in the middle of the ArrayList requires shifting elements to make room for the new one. This operation takes O(n) time because, in the worst case, all elements after the insertion point need to be shifted.

Space Complexity:

The space complexity of an ArrayList is linear O(n) with respect to the number of elements it contains. The underlying array needs to be allocated with a size that accommodates the elements. If the array needs to be resized, a new array is created, and the elements are copied, resulting in additional space usage.

4. TREE SET:

In Java, TreeSet is part of the Java Collections Framework and is an implementation of the Set interface. It is a NavigableSet based on a TreeMap

Time Complexity:

The time complexity for adding an element to a TreeSet is logarithmic in the number of elements. O(Log n).

Space Complexity:

The space complexity of a TreeSet is linear with respect to the number of elements. Each element is stored along with additional data structures to maintain the binary search tree structure

Method Used In The Program :

1. Add Task
2. Modify Task
3. View Task
4. Add Dependency
5. Show Overdue Tasks
6. Complete Task
7. Display Tasks by Priority
8. Exit

Conclusion:

Thus we Can Conclude That:

The advanced task management system outlined in the project definition aims to provide a comprehensive solution for efficiently organizing, tracking, and managing tasks within an organization or for personal use.

Using Different Data Struture and its different Time and Space Complexity we are able to learn the different concept of how this Data struture can be used in developing a task management and how it works internally.