23CSE203 - DATA STRUCTURES AND ALGORITHM

CUSTOMISED LEARNING EXPERIENCE PLATFORM FOR ADAPTIVE EDUCATION

TEAMMATES:

YUVA HASINI (CB.SC.U4CSE23412)

JASWITA.M(CB.SC.U4CSE23430)

SRIJA.K(CB.SC.U4CSE23448)

VAMSI VVS(CB.SC.U4CSE23454)

HARSHINI.V(CB.SC.U4CSE23455)

AVK BHAVAN SURYA(CB.SC.U4CSE23467)

PROBLEM STATEMENT

ADT'S USED:

- GRAPHS
- TRIE
- HASHMAP
- HEAP
- QUEUE
- ARRAY

GRAPHS

WHAT ARE GRAPHS?

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras arcu metus, feugiat vitae ante ac, aliquet tempus ante. In euismod nibh eget lacinia imperdiet. Maecenas tristique risus felis, ut ultrices lectus ultrices lobortis.

WHY DID WE USE IT IN OUR CODE?

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras arcu metus, feugiat vitae ante ac, aliquet tempus ante. In euismod nibh eget lacinia imperdiet. Maecenas tristique risus felis, ut ultrices lectus ultrices lobortis.

TRIE

WHAT IS TRIE?

A Trie is a tree-like data structure that is used to efficiently store and retrieve strings, particularly useful for scenarios involving words, prefixes, or dictionary-type data.It enables O(m) search time for words of length m

WHY DID WE USE IT IN OUR CODE?

Tries support fast dynamic operations, such as adding or deleting words, making them ideal for managing large dictionaries of strings with shared prefixes.

HASHMAP

WHAT ARE HASHMAPS?

A hash map is a data structure that stores data in key-value pairs and enables fast access using a unique key. It uses a hash function to assign each key to a specific index, allowing quick lookups, insertions, and deletions (typically O(1) time complexity).

WHY DID WE USE IT IN OUR CODE?

Fast access: We can quickly retrieve each student's data using their unique student_id.
Unique keys: Hash maps are ideal for organizing data with unique identifiers.
Easy updates: Hash maps allow efficient updates without rearranging or searching through elements.

Min-HEAP

WHAT IS MIN-HEAP?

The key of each node is greater than or equal to the key of its parent. This means the smallest element is always at the root.

WHY DID WE USE IT IN OUR CODE?

We’re using a Min-Heap, as elements with the smallest scores (calculated from difficulty and relevance) should be at the root.Fast Access: Min-heaps provide quick access to the smallest element, ideal for priority queues or recommendations.
Efficient Insert/Delete: Both insertions and deletions are $O(\log n)$.O(logn), faster than keeping a sorted list which is $O(n)$.

QUEUE

WHAT ARE QUEUES?

A Queue is a data structure that operates in a First-In-First-Out (FIFO) order, meaning elements are processed in the order they are added.

WHY DID WE USE IT IN OUR CODE?

Here, it's used to manage and serve recommended learning tasks for each student, ensuring that tasks are handled sequentially based on priority and prerequisites.It has time complexity of O(n).

ARRAY

WHAT IS ARRAY?

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras arcu metus, feugiat vitae ante ac, aliquet tempus ante. In euismod nibh eget lacinia imperdiet. Maecenas tristique risus felis, ut ultrices lectus ultrices lobortis.

WHY DID WE USE IT IN OUR CODE?

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras arcu metus, feugiat vitae ante ac, aliquet tempus ante. In euismod nibh eget lacinia imperdiet. Maecenas tristique risus felis, ut ultrices lectus ultrices lobortis.

Thank You