Merchant Monetary System

Data Structure

Project Supervisor

Mr. Samyan Qayyum Wahla

Group ID ($G11$)

Project Member

Syed Hashir	2021-CS-1
Kabir Ahmed	2021-CS-4
M. Hamad Hassan	2021-CS-33

Department of Computer Science
University of Engineering and Technology, Lahore
Pakistan

Data Strucuture

The following section shows the reason for choosing the data structure in the particular use case with a brief explanation.

Use Case IDs	U01,U02,U03,U04,U05,U06,U07,U08,U09,U010,U11,U12,U13,U14, U15,U16,U17,U18,U21,U22,U25,U26,U30,U31
Data Structure Used	Linked List
Time Complexity	In Worst Case: Search: O(n), Insertion: O(1), Deletion: O(n)
Space Complexity	O(n)
Justification for the use of data structure	In mentioned use case required a linear-dynamic data structure. Doubly LinkedList provides an efficient way to search the specific information from a large amount of data and then compare it with input information to produce the required result. It helps to store and delete the data fastly. It allows you to move back and forth in the list to get the required result.
Available choices	Array List,Hash Table
Comparison	The array list worst and average case time complexity is O(n). It takes contiguous memory. The hash table is best in the average case, but in the worst case time, complexity rise to O(n). It takes contiguous memory for storing the hash function value. In the average and worst case, the linked list insertion and deletion take O(1) time. In the average and worst case, it takes O(n) time for deletion. It did not require contiguous memory allocation.Array list, hash table, and linked list space complexity O(n) are the same.

Use Case IDs	U19
Data Structure Used	Queue
Time Complexity	In Worst Case: Search: O(n), Insertion: O(1), Deletion: O(n)
Space Complexity	O(n)
Justification for the use of data structure	In mentioned use case required a linear-dynamic data structure. Queue provides an efficient way to search the specific information from a large amount of data and then compare it with input information to produce the required result. It helps to store the data of orders in specific pattern . It allows to get the ordered pattern of incoming and outgoing orders data and shows the required result.
Available choices	Array List,Hash Table, Linked List
Comparison	The array list worst and average case time complexity is O(n). It takes contiguous memory. The hash table is best in the average case, but in the worst case time, complexity rise to O(n). It takes contiguous memory for storing the hash function value. In the average and worst case, the linked list insertion and deletion take O(1) time. In the average and worst case, Queue takes O(n) time for deletion. It gives the specific ordered pattern to store and Dequeue reguired data.Array list, hash table, and linked list space complexity O(n) are the same.

Use Case IDs	U11,U12
Data Structure Used	Array List
Time Complexity	In Worst Case: Search: O(n), Insertion: O(1), Deletion: O(n)
Space Complexity	O(n)
Justification for the use of data structure	In mentioned use case required a linear-dynamic data structure. Queue provides an efficient way to search the specific information from a large amount of data and then compare it with input information to produce the required result.IT allows to get specific data and shows the required result.Only a specific detail of the data is required to store the specific information in this use case.
Available choices	Linked List
Comparison	The array list worst and average case time complexity is O(n). In the average and worst case, the linked list insertion and deletion take O(1) time. IN the average and worst case, List takes O(n) time for deletion. It did not require contiguous memory allocation.Array list and linked list space complexity O(n) are the same therefore for the small data Array list used.

Use Case IDs	U23,U24
Data Structure Used	Heap
Time Complexity	In Worst Case: Search: O(n), Insertion: O(n), Deletion: O(n)
Space Complexity	O(n)
Justification for the use of data structure	In mentioned use case required a Heap data structure. Heap provides an efficient way to search the specific information from a moderate and huge amount of data and then compare it with input information to produce the required result. It allows to get specific data of some specific data, it allows to apply specific operation on that and shows the required result.
Available choices	Linked List
Comparison	The Heap worst and average case time complexity is O(n). In the average and worst case, the linked list insertion and deletion take O(1) time. IN the average and worst case, List takes O(n) time for deletion. It did not require contiguous memory allocation.Heap and linked list space complexity O(n) are the same therefore for the Detailed data Heap used.

Use Case IDs	U20
Data Structure Used	Tree
Time Complexity	In Worst Case: Search: O(lg n), Insertion: O(lg n), Deletion: O(lg n)
Space Complexity	O(n)
Justification for the use of data structure	In mentioned use case required a Tree data structure. Tree provides an efficient way to search the specific information from a moderate and huge amount of data and then compare it with input information to produce the required result. It allows to get specific detailed ordered data in a specfic manner and point out the points to some reference .
Available choices	Linked List
Comparison	The Tree worst and average case time complexity is O(lg n). In the average and worst case, the linked list insertion and deletion take O(1) time. IN the average and worst case, List takes O(n) time for deletion. It did not require contiguous memory allocation.Tree and linked list space complexity O(n) are not same therefore, for the Detailed data Tree is prefered to be used.

Use Case IDs	U29
Data Structure Used	Graph
Time Complexity	In Worst Case: Search: O(lg n), Insertion: O(lg n), Deletion: O(lg n)
Space Complexity	O(n)
Justification for the use of data structure	In mentioned use case required a Tree data structure. Tree provides an efficient way to search the specific information from a moderate and huge amount of data and then compare it with input information to produce the required result. It allows to plot the data and apply operation on the data to show the required result.
Available choices	Tree
Comparison	The Graph worst and average case time complexity is O(lg n). In the average and worst case, the Tree insertion and deletion take O(lg n) time. IN the average and worst case, Tree takes O(n) time for deletion. It did not require contiguous memory allocation.Tree and Graph space complexity O(n) are same therefore, for the Plotting data graph is prefered to be used.