EECS 114

Name: Tram Le

Student ID: 48109508

Date: 05/28/2023

Homework #3

Question 3:

To achieve O(1) time complexity for the insert, delete, replace, and right operations, a suitable data structure can be designed using a combination of a doubly linked list and a hash table.

Data structure design:

Each element in the doubly linked list represents a character in the string.
The nodes of the doubly linked list contain the character value and pointers to the previous and next nodes.
In addition to the doubly linked list, a hash table is utilized to map each character to its corresponding node in the doubly linked list. This enables quick access to any character in the list.

Explanation of operations:

Insert: To insert a new character at a specific position, a new node is created with the given character. The necessary pointers in the doubly linked list are updated to include the new node. Additionally, the hash table entry is updated to map the character to the newly inserted node. This operation can be completed in O(1) time.

Delete: To remove a character at a specific position, the corresponding node is located using the hash table. The neighboring nodes' pointers in the doubly linked list are updated to bypass the node being deleted. The hash table entry for the character is also removed. This operation can be completed in O(1) time.

Replace: To update a character at a specific position, the corresponding node in the doubly linked list is located using the hash table. The value of the node is updated with the new character. This operation can be completed in O(1) time.

Right: To move to the next node in the doubly linked list from a given position, the next pointer of the current node is followed. This allows traversing through the linked list efficiently and accessing the characters in a forward direction. This operation can be completed in O(1) time.

By combining the doubly linked list and the hash table, the data structure provides constant time complexity (O(1)) for all four operations.