**Exercise 5: Task Management System**

Because linked lists allow for the flexible addition and removal of members, they are dynamic data structures that are perfect for job management. Each node in a single linked list points to the next, allowing data to be stored sequentially. For adding and navigating tasks in a linear order, this is an efficient method. By appending a previous pointer to every node, a doubly linked list improves this and enables efficient deletion of any node without requiring a traversal of the entire list. While there are benefits to both structures for managing tasks, the decision comes down to particular needs such as frequency of deletion and traversal direction.

Especially at the beginning, linked lists are efficient for insertions and deletions. Single-linked lists require a fixed amount of time, while doubly-linked lists require a constant amount of time for both head and tail operations. On the other hand, both forms of linked lists have a linear time complexity of O(n) for searching and accessing elements. Although doubly linked lists allow greater flexibility in traversal and deletion, they do require additional space for the preceding pointer.

Linked lists are more adaptable and effective at managing insertions and deletions than arrays, they are a better choice for handling dynamic data. In contrast to fixed-size arrays, linked lists are able to dynamically expand and contract to meet changing data needs. It is usually faster to adjust pointers when adding or removing members from a linked list than it is to relocate elements within an array. Because of this, linked lists are better suited for systems like task management where data sizes shift regularly. Linked lists can also be effectively used to implement other data structures, such as stacks and queues.However, the random access is slower in linked lists compared to arrays.