FND23

Queue and a linked lists are considered ADTs. Queues implement a FIFO (First in first out) approach to process data. To add an element to a queue, we enqueue it. When we want to remove an element from a queue, we dequeue it. What gets dequeued is the first element that was enqueued. When we peek an element (view the element that when we dequeue will be dequeued) we peek the element that was first enqueued. The functionalities of queues are enqueue, dequeue, and peek. Due to these functionalities, we can only manipulate the first element that was enqueued (by dequeuing) and change and look at the last enqueued element (by enqueuing and peeking). When we store an item in a linked list, it will always be stored in the first free and available memory location. When we add an element to a linked list, we insert it. When we want to remove an element from a linked list, we delete it. This could be done by giving the key or the position of the element (node) that we want to delete. Most importantly, unlike a queue, we can search a linked list. We can read and manipulate any element in a linked list regardless of the time it was inserted and its position in the list, which gives a lot more freedom.