1 Explain at least one benefit and one drawback for using a linked list in place of a traditional array in the `Stack` and `Queue` data structures. (3)

2 Implement the `trimN()` method in our `LinkedList` class. This method should take `N` as a parameter and remove the last `N` nodes from the list. (3)

3 Implement `Queue` using a linked list. Ensure you maintain the use of generics as appropriate. (5)

4 Implement the `find()` method in our `LinkedList` class. This method should take some data, `key`, as a parameter and return the index where `key` exists as the data for a particular `Node`. This method should return the value -1 if `key` does not exist in the linked list. (5)

5 *Doubly-Linked List.* Make all necessary changes to our `LinkedList` in order to implement a *doubly-linked list* in which each node has both `Next` and `Previous` attributes. In particular, ensure that `add()` and `remove()` work as intended. (10)

6 *Deque.* A double-ended queue or *deque* (pronounded "deck") is like a stack of a queue but supports adding and removing items from both ends. A deque stores a collection of items and supports the following operations: (20)

`isEmpty()`	is the deque empty?
`size()`	how many items are in the deque?
`pushLeft(item)`	add an item to the left end
`pushRight(item)`	add an item to the right end
`popLeft()`	remove and return an item from the left end
`popright()`	remove and return an item from the right end

Implement a *deque* in Java using a doubly-linked list.