**DOUBLE-LINKED LISTS**

Limitations of single linked list:

* Insertion at the front is O(1); insertion at other positions is O(n)
* Insertion is convenient only after a referenced node
* Removing a node requires a reference to the previous node
* We can traverse the list only in the forward direction

We can overcome these limitations:

* Add a reference in each node to the previous node, creating a double-linked list

Diagram

Description automatically generated

Node Class

Diagram

Description automatically generatedprivate static class Node<E> {

private E data;

private Node<E> next = null;

private Node<E> prev = null;

private Node(E dataItem) {

data = dataItem;

}

}

Diagram

Description automatically generated

Diagram

Description automatically generated

A Double-Linked List Class

Graphical user interface

Description automatically generatedSo far we have worked only with internal nodes

As with the single-linked class, it is best to access the internal nodes with a double-linked list object

A double-linked list object has data fields:

* head: a reference to the first list Node
* tail: a reference to the last list Node
* size

Insertion at either end is O(1); insertion elsewhere is still O(n)

Removal from either end is also O(1)

In single linked list, even if you keep tail information, removing the last element cannot be done in constant time since tail should point to second last element afterwards but I cannot find that element in constant time by using single linked list. I have to traverse from the beginning and it is linear time.

Line chart

Description automatically generated with medium confidence

**CIRCULAR LISTS**

Circular double-linked list:

* Link last node to the first node, and
* Link first node to the last node

We can also build singly-linked circular lists:

* Traverse in forward direction only

Advantages:

* Continue to traverse even after passing the first or last node
* Visit all elements from any starting point
* Never fall of the end of a list

Disadvantage:

* Code must avoid an infinite loop

Diagram

Description automatically generated