KWLinkedList

We will define a KWLinkedList class which implements some of the methods of the List interface (AbstractList class implements other methods).

KWLinkedList class is for demonstration purposes only; Java provides a standard LinkedList class in java.util which you should use in your programs.

Graphical user interface, text, application

Description automatically generated

Node 🡪 private inner class

Text

Description automatically generated with medium confidence

Text, letter

Description automatically generatedAdd Method

First choice (traverse from head yourself):

1. Obtain a reference, nodeRef, to the node at position index
2. Insert a new Node containing obj before the node referenced by nodeRef

Second choice (implemented at right):

To use a ListIterator object to implement add:

1. Obtain an iterator that is positioned just before the Node at position index
2. Insert a new Node containing obj before the Node currently referenced by this iterator

listIterator will return an iterator and we call add method of that iterator.

It is not necessary to declare a local ListIterator reference; the method call listIterator returns an anonymous ListIterator object.

Creating an iterator takes linear time (worst case 🡪 index is middle), then add will take constant time.

You can start from head, traverse through the list and add new node instead of using iterators. It takes same time.

Text

Description automatically generatedGet Method

First choice (traverse from head yourself):

1. Obtain a reference, nodeRef, to the node at position index
2. Return the contents of the Node referenced by nodeRef

Second choice (implemented at right)

Text, letter

Description automatically generatednext 🡪 constant time

creating iterator 🡪 linear time

total 🡪 linear time

Other Add and Get Methods

Graphical user interface, application, Teams

Description automatically generated

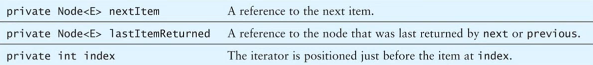
You can use get(0) for getFirst and get(size-1) for getLast.

get(0) takes constant time. If size-1 is assumed as special case in listIterator, then get(size-1) also takes constant time.

IMPLEMENTING THE LISTITERATOR INTERFACE

KWListIter is an inner class of KWLinkedList which implements the ListIterator interface.

We said iterator refers in between the elements but there is no such thing. So we are gonna simulate it with keeping reference to nextItem.



index is for simplicity. We can also calculate index value if we know the nextItem, we can start from beginning and calculate the index where we are at. To make implementation faster we keep index.

index and nextItem should be consistent.

If index is 2, last item return may be blue or red according to use of next or prev methods.

index 🡪

Diagram

Description automatically generated with medium confidence



Diagram

Description automatically generated



Red numbers are index.

set and remove operations are based on the previous operations performed. lastItemReturned is for learning the last operation. You can also keep a boolean that means 1 for next 0 for prev but there would be a problem if none operations has performed. It is easier to keep lastItemReturned.

You can also keep an integer for it. 0 for 0 operation has been done, 1 for former operation is next and 2 for former operation is prev. Then you can find the lastItem with using prev or next references in the node.

Graphical user interface, application

Description automatically generated with medium confidence

KWListIter is inner class as Node but it is not static as Node. It shouldn’t be static since the add, remove, and set methods has to refer to KWLinkedList (outer) class. Node class doesn’t need to access to KWLinkedList.

Constructor

Graphical user interface, text, application, email

Description automatically generated

size comes from KWLinkedList. So as we mentioned, we have to access KWLinkedList.

Time 🡪 constant if i = 0 or size, linear for same cases as i = n/2 🡪 overall is O(n)

hasNext()

Tests to see if nextItem is null

Text

Description automatically generated

Advancing the Iterator - requires constant time

Graphical user interface, diagram

Description automatically generated

Diagram

Description automatically generated

Graphical user interface, text, application, email

Description automatically generatedPrevious Methods - require constant time

nextItem is null in 2 cases:

1. If we are at the end of the list
2. If the list is empty

ListIterator Interface

Table

Description automatically generated



Add Method

When adding, there are 4 cases to address:

1. Add to an empty list
2. Add to the head of the list
3. Add to the tail of the list
4. Add to the middle of the list

Adding to an empty list (constant time):

Diagram

Description automatically generated

Adding to the head of the list (constant time):

We can also control as “if (index == 0)” since we are sure that head isn’t null because we controlled it in adding to empty list part. So “index == 0” means list is not empty and we are at the beginning.

tail, nextItem, and lastItemReturned don’t need to be updated.

Graphical user interface, application

Description automatically generated

Graphical user interface, diagram, application

Description automatically generated

Adding to the tail of the list (constant time):

Graphical user interface, diagram, application

Description automatically generated

Adding to the middle of the list (constant time):

Graphical user interface, diagram

Description automatically generated

Inner Classes: Static and Nonstatic

KWLinkedList contains 2 inner classes:

1. Node<E> is declared static: there is no need for it to access the data fields of its parent class, KWLinkedList
2. KWListIter cannot be declared static because its methods access and modify data fields of KWLinkedList’s parent object which created it

An inner class which is not static contains an implicit reference to its parent object and can reference the fields of its parent object

Since its parent class is already defined with the parameter <E>, KWListIter cannot be declared as KWListIter<E>; if it were, an incompatible types syntax error would occur