Stephen Hogeman

Professor Connamacher

CSDS 132

December 5, 2021

Project 4 Testing Report

Purpose of the Project:

The purpose of this project was to utilize linked lists, iterators, and generic types, to create a main method that could sequence DNA segments that consisted of the base types A, G, C, and T. To do this, methods needed to be added to the DoubleLinkedList class created in Lab 11, to manipulate and traverse (effectively) linked lists. Also, the iterator() method of the Iterable interface was overridden to return a ListIterator, with overwritten methods. Finally, a DNA class was created, to create methods necessary to add specific Base types, to a DNA segment, which was a DoubleLinkedList, that only allowed the Base Types.

New Methods of DoubleLinkedList

testEquals():

- The testEquals() method of the DoubleLinkedListTester class was created to test the equals() method of the DoubleLinkedList class. This evaluates whether 2 linked lists contained the same elements in the same order. The tests are as follows:
    - Test 1: Tests if the last element is not equals.
    - Test 2: Tests if a middle element is not equals.
    - Test 3: Tests if the first element is not equals.
    - Test 4: Tests if an empty list is compared.
    - Test 5: Tests if a list that is one longer is not equals.
    - Test 6: Tests if a list that is many elements longer is not equals.
    - Test 7: Tests 2 equal lists.

testAppend():

- The testAppend() method of the DoubleLinkedListTester class was created to test the append() method of the DoubleLinkedList class. The append() method appends 1 list to the end of another.
    - Test 1: Tests the element of list1 (empty appended to empty)(Returns null).
    - Test 2: Tests the first element of list3 (List of 1 appends empty list).
    - Test 3: Tests the next element of list3 (Returns null).
    - Test 4: Tests the front element of list4 (List of 2 appends list of 1).
    - Test 5: Tests the second element of list4.
    - Test 6: Tests the third element of list4.
    - Test 7: Tests the next element of list4 (Returns null).
    - Test 8: Tests the 1st element of list5 (list of many appends list of 3).
    - Test 9: Tests the switch element of list5 (Switch refers to location where lists appended).
    - Test 10: Tests the end element of list4.

- Test 11: Tests the next element of list5 (Returns null).

testListIterator():

- The testListIterator() method of the DoubleLinkedListTester class tests the iterator() method of the DoubleLinkedList class. This method overrides the element of the Iterable interface, and returns a ListIterator.
  - Test 1: Tests whether or not an iterator has been created.

iterator() Anonymous Class:

The following methods are methods of the iterator() method in DoubleLinkedList, which is an anonymous class that overrides the iterator() method of the Iterable interface.

testAdd()

- The testAdd() method of the DoubleLinkedListTester class tests the add method of the iterator() anonymous class. This method adds an element to an iterator.
  - Test 1: Tests an empty iterator adding 2 values. Front test.
  - Test 2: Tests an empty iterator adding 2 values. Back test.
  - Test 3: Tests an iterator with 1 adding 1 element. Original element.
  - Test 4: Tests an iterator with 1 adding 1 element. Added element.
  - Test 5: Tests an iterator adding 3 elements. First element.
  - Test 6: Tests an iterator adding 3 elements. Last element.

testHasNext():

- The testHasNext() method of the DoubleLinkedListTester class tests the hasNext method of the iterator() anonymous class. This method determines if there is another element in the iterator after next() is called.
  - Test 1: Testing an empty iterator.
  - Test 2: Testing an iterator with 1 element.
  - Test 3: Testing an iterator with 2 elements. First.
  - Test 4: Testing an iterator with 2 elements. Second.
  - Test 5: Testing an iterator with 2 elements. Outside.

testHasPrevious():

- The testHasPrevious() method of the DoubleLinkedListTester class tests the hasPrevious method of the iterator() anonymous class. This method determines if there is a previous element in the iterator once previous() is called.
  - Test 1: Tests an empty iterator.
  - Test 2: Tests an iterator with 1 element.
  - Test 3: Tests an iterator with 2 elements. Last.
  - Test 4: Tests an iterator with 2 elements. First.
  - Test 5: Tests an iterator with 2 elements. Outside.

testNext()

- The testNext() method of the DoubleLinkedListTester class tests the next method of the iterator() anonymous class. This method gets the next element in the iterator after the current one.

- Test 1: Test an empty iterator.
- Test 2: Testing the only element of an iterator.
- Test 3: Testing that the previous iterator has been iterated.
- Test 4: Testing an iterator with 2 elements, 1st to 2nd.
- Test 5: Testing an iterator with 2 elements, 2nd to end.
- Test 6: Testing an iterator with 2 elements, end of iterator.

testPrevious()

- The testPrevious() method of the DoubleLinkedListTester class tests the previous method of the iterator() anonymous class. This method gets the previous element in the iterator before the current one.
  - Test 1: Test an empty iterator.
  - Test 2: Testing the only element of an iterator.
  - Test 3: Testing that the previous iterator has been iterated.
  - Test 4: Testing an iterator with 2 elements, 2nd to 1st.
  - Test 5: Testing an iterator with 2 elements, 1st to front.

testSet()

- The testSet() method of the DoubleLinkedListTester class tests the set method of the iterator() anonymous class. This method sets the current value of the iterator node. This will in turn set the value of the current node of the list to be the element input.
  - Test 1: Testing the list before the element is set.
  - Test 2: Testing the list once the element is set.
  - Test 3: Tests one element before it is set.
  - Test 4: Tests next element before it is set.
  - Test 5: Tests next element before it is set.
  - Test 6: Tests one element after it is set.
  - Test 7: Tests next element once it is set.
  - Test 8: Tests next element once it is set.

testRemove()

- The testRemove() method of the DoubleLinkedListTester class tests the remove method of the iterator() anonymous class. This method only throws an UnsupportedOperationException.
  - Test 1: Test that exception is thrown.

testPreviousIndex():

- The testPreviousIndex() method of the DoubleLinkedListTester class tests the previousIndex method of the iterator() anonymous class. This method only throws an UnsupportedOperationException.
  - Test 1: Test that exception is thrown.

testNextIndex():

- The testNextIndex() method of the DoubleLinkedListTester class tests the nextIndex method of the iterator() anonymous class. This method only throws an UnsupportedOperationException.

- Test 1: Test that exception is thrown.

DNA Class Tester

testToString()

- The testToString() method of the DNATester class tests the toString() method of the DNA class. This method returns the string value of this DNA sequence.
  - Test 1: Testing a list that has no bases.
  - Test 2: Testing a list that has multiple bases.

testString2DNA()

- The testString2DNA() method of the DNATester class tests the string2DNA method of the DNA class. This method takes a string input and returns the DNA representation of the string. If an illegal argument is used as input, an exception is thrown.
  - Test 1: Testing one element of a string to DNA.
  - Test 2: Testing multiple elements in a string to DNA.
  - Test 3: Testing if an illegal input is given.

testSplice()

- The testSplice() method of the DNATester class tests the splice method of the DNA class. This method removes the input int from the input DNA segment and appends it to this segment.
  - Test 1: Testing a segment splicing a strand of 2. Front.
  - Test 2: Testing a segment splicing a strand of 2. Middle.
  - Test 3: Testing a segment splicing a strand of 2. End.
  - Test 4: Testing a segment splicing a full strand, 3 bases. Front.
  - Test 5: Testing a segment splicing a full strand, 3 bases. Middle.
  - Test 6: Testing a segment splicing a full strand, 3 bases. End.

testOverlaps():

- The testOverlaps() method of the DNATester class tests the overlaps method of the DNA class. This method determines whether the end int n values of the first input DNA list are equivalent to the first int n values of the second input DNA list. If this is true, it returns true.
  - Test 1: Testing overlaps on sequences of 2.
  - Test 2: Testing overlaps on sequences of many.
  - Test 3: Testing overlaps on sequences of many (False).
  - Test 4: Testing overlaps on sequences of 2 (False).

testCompareTo():

- The testCompareTo() method of the DNATester class tests the compareTo method of the DNA class. This method overrides the compareTo method of the Comparable interface, which allows two objects to be compared. The overridden method determines whether a list is longer, and in alphabetical order.
  - Test 1: Tests a sequence of 1 compared to a sequence of 0.

- Test 2: Tests a sequence of 0 compared to a sequence of 1.
- Test 3: Tests a sequence of many compared to a sequence with the same number of bases.

testMain()

- The main method of the DNA class doesn't contain a Junit test; however, its purpose is to take 2 input strings, determine where the greater overlap lies between the 2, and then splice the 2 strings. To run the method, one must type the 2 strings out in the interactions pane.

Test 1:

- The first test involves creating no input, or 1 input in the run method.

> run DNA

2 Strings were not input!

or

> run DNA ACATCAT

2 Strings were not input!

Test 2:

- The second test involves inputting 2 string inputs that don't contain base arguments.

> run DNA acuie 789112

Input included an input that is not a base!

Test 3:

- The third test involves inputting 2 strings of 1 that don't interlap. This will not return a value, as there is nothing to splice.

> run DNA T G

>

Test 4:

- The fourth test involves inputting 2 strings of 1 that interlap. This will not return a value, as there is nothing to splice.

> run DNA T T

>

Test 5:

- The fifth test involves inputting 2 strings of 3 bases, that have 1 base that overlaps. This will return ATCGG

> run DNA ATC CGG

ATCGG

Test 6:

- The sixth test involves inputting 2 longer strings, that have multiple bases that overlap. This will return

> run DNA ATGCAT CATTGCA

ATGCATTGCA