KEY: Orange highlight = Test Added during this milestone
Red highlight = Impossible test case

<table>
<tr><td>Test</td><td>Equivalence class partitioning(Wect, Sect, Wrect, Srect)</td></tr>
<tr><td>testAdd</td><td>

Variables:

List length L
Object O

Equivalence classes

L1:L=0
L2:L>0
O1: O does not already exist in list
O2: O already exists in list

Wect
<table>
<tr><td>ID</td><td>L</td><td>O</td></tr>
<tr><td>WECT1</td><td>L1</td><td>O1</td></tr>
<tr><td>WECT2</td><td>L2</td><td>O2</td></tr>
</table>
Sect
<table>
<tr><td>ID</td><td>L</td><td>I</td></tr>
<tr><td>SECT1</td><td>L1</td><td>O1</td></tr>
<tr><td>SECT2</td><td>L2</td><td>O1</td></tr>
<tr><td>SECT3</td><td>L1</td><td>O2</td></tr>
<tr><td>SECT4</td><td>L2</td><td>O2</td></tr>
</table>
Wrect and Srect would involve L < 0 but this is not possible
Wrect and srect are covered by testing of a null variable list

</td></tr>
<tr><td>testAddAll</td><td>

Variables:

List length L
Object list O

Equivalence classes

L1:L=0
L2:L>0
O1: O can be fully inserted into list
O2: O can be partially inserted into list
O3: O cannot be inserted at all into list

Wect

</td></tr>
</table>

<table>
<tr>
<td></td>
<td>
<table>
<tr><td>ID</td><td>L</td><td>O</td></tr>
<tr><td>WECT1</td><td>L1</td><td>O1</td></tr>
<tr><td>WECT2</td><td>L2</td><td>O2</td></tr>
<tr><td>WECT3</td><td>L1</td><td>O3</td></tr>
</table>

Sect

<table>
<tr><td>ID</td><td>L</td><td>I</td></tr>
<tr><td>SECT1</td><td>L1</td><td>O1</td></tr>
<tr><td>SECT2</td><td>L2</td><td>O1</td></tr>
<tr><td>SECT3</td><td>L1</td><td>O2</td></tr>
<tr><td>SECT4</td><td>L2</td><td>O2</td></tr>
<tr><td>SECT5</td><td>L1</td><td>O3</td></tr>
<tr><td>SECT6</td><td>L2</td><td>O3</td></tr>
</table>

Wrect and Srect would involve L < 0 but this is not possible
Wrect and srect are covered by testing of a null variable list

</td>
</tr>
<tr>
<td>testEquals</td>
<td>

Variables:

List length L

Equivalence classes

L1:L=0
L2:L>0

Thus, wect and sect are both covered by using an empty list and a nonempty

This is already done

Wrect and Srect would involve L < 0 but this is not possible
Wrect and srect are covered by testing of a null variable list

</td>
</tr>
<tr>
<td>testHashCode</td>
<td>

Variables:

List length L

Equivalence classes

L1:L=0
L2:L>0

Thus, wect and sect are both covered by using an empty list and a nonempty

This is already done

Wrect and Srect would involve L < 0 but this is not possible
Wrect and srect are covered by testing of a null variable list

</td>
</tr>
</table>

<table>
<tr>
<td>testGet</td>
<td>

Variables:

List length L
Index I

Equivalence classes

L1: L=0
L2: L>0
I1: I<0
I2: I>=0&&I<L
I3: I>=L

Wect
<table>
<tr><td>ID</td><td>L</td><td>I</td></tr>
<tr><td>WECT1</td><td>L1</td><td>I1</td></tr>
<tr><td>WECT2</td><td>L2</td><td>I2</td></tr>
<tr><td>WECT3</td><td>L1</td><td>I3</td></tr>
</table>

Sect
<table>
<tr><td>ID</td><td>L</td><td>I</td></tr>
<tr><td>SECT1</td><td>L1</td><td>I1</td></tr>
<tr><td>SECT2</td><td>L2</td><td>I1</td></tr>
<tr><td>SECT3</td><td>L1</td><td>I2</td></tr>
<tr><td>SECT4</td><td>L2</td><td>I2</td></tr>
<tr><td>SECT5</td><td>L1</td><td>I3</td></tr>
<tr><td>SECT6</td><td>L2</td><td>I3</td></tr>
</table>

Wrect and Srect would involve L < 0 but this is not possible
Wrect and srect are covered by WECT1, WECT3, SECT1, SECT2, SECT5, and SECT6
Wrect and srect are also covered by testing of a null variable list

</td>
</tr>
<tr>
<td>testIndexOf</td>
<td>

Variables:

List length L
Object O

Equivalence classes

L1: L=0
L2: L>0
O1: O is in list
O2: O is not in list

</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td>
● Wect
<table>
<tr><td>ID</td><td>L</td><td>O</td></tr>
<tr><td>WECT1</td><td>L1</td><td>O1</td></tr>
<tr><td>WECT2</td><td>L2</td><td>O2</td></tr>
</table>
● Sect
<table>
<tr><td>ID</td><td>L</td><td>I</td></tr>
<tr><td>SECT1</td><td>L1</td><td>O1</td></tr>
<tr><td>SECT2</td><td>L2</td><td>O1</td></tr>
<tr><td>SECT3</td><td>L1</td><td>O2</td></tr>
<tr><td>SECT4</td><td>L2</td><td>O2</td></tr>
</table>

Wrect and Srect would involve L < 0 but this is not possible
Wrect and srect are covered by WECT1, SECT1
Wrect and srect are also covered by testing of a null variable list

</td>
</tr>
<tr>
<td>testLastIndexOf</td>
<td>

Variables:

List length L
Object O

Equivalence classes

L1: L=0
L2: L>0
O1: O is in list
O2: O is not in list

● Wect
<table>
<tr><td>ID</td><td>L</td><td>O</td></tr>
<tr><td>WECT1</td><td>L1</td><td>O1</td></tr>
<tr><td>WECT2</td><td>L2</td><td>O2</td></tr>
</table>
● Sect
</td>
</tr>
</table>

<table>
<tr><td></td><td>
<table>
<tr><td>ID</td><td>L</td><td>I</td></tr>
<tr><td>SECT1</td><td>L1</td><td>O1</td></tr>
<tr><td>SECT2</td><td>L2</td><td>O1</td></tr>
<tr><td>SECT3</td><td>L1</td><td>O2</td></tr>
<tr><td>SECT4</td><td>L2</td><td>O2</td></tr>
</table>

Wrect and Srect would involve L < 0 but this is not possible
Wrect and srect are covered by WECT1, SECT1
Wrect and srect are also covered by testing of a null variable list

</td></tr>
<tr><td>testIterator</td><td>

Variables:

List length L

Equivalence classes

L1: L=0
L2: L>0

Thus, wect and sect are both covered by using an empty list and a nonempty

This is already done

Wrect and Srect would involve L < 0 but this is not possible
Wrect and Srect are covered by testing of a null variable list

</td></tr>
<tr><td>testRemove</td><td>

Variables:

List length L

Equivalence classes

L1: L=0
L2: L>0

Thus, wect and sect are both covered by using an empty list and a nonempty

This is already done

Wrect and Srect would involve L < 0 but this is not possible
Wrect and rect are covered by testing of a null variable list

</td></tr>
<tr><td>testSublist</td><td>

Variables:

List length L

Equivalence classes

L1: L=0
L2: L>0

Thus, wect and sect are both covered by using an empty list and a nonempty

This is already done

Wrect and Srect would involve L < 0 but this is not possible
Wrect and Srect are covered by testing of a null variable list

</td></tr>
</table>

testSubListAddBegin	● Variables: ○ List length L ● Equivalence classes ○ L1: L=0 ○ L2: L>0 ● Thus, wect and sect are both covered by using an empty list and a nonempty ○ This is already done ● Wrect and Srect would involve L < 0 but this is not possible ● Wrect and Srect are covered by testing of a null variable list
testSubListAddEnd	● Variables: ○ List length L ● Equivalence classes ○ L1: L=0 ○ L2: L>0 ● Thus, wect and sect are both covered by using an empty list and a nonempty ○ This is already done ● Wrect and Srect would involve L < 0 but this is not possible ● Wrect and Srect are covered by testing of a null variable list
testSubListAddMiddle	● Variables: ○ List length L ● Equivalence classes ○ L1: L=0 ○ L2: L>0 ● Thus, wect and sect are both covered by using an empty list and a nonempty ○ This is already done ● Wrect and Srect would involve L < 0 but this is not possible ● Wrect and Srect are covered by testing of a null variable list
testSubListRemove	● Variables: ○ List length L ● Equivalence classes ○ L1: L=0 ○ L2: L>0 ● Thus, wect and sect are both covered by using an empty list and a nonempty ○ This is already done ● Wrect and Srect would involve L < 0 but this is not possible ● Wrect and Srect are covered by testing of a null variable list
testTransformedList	● Variables: ○ List length L ● Equivalence classes ○ L1: L=0 ○ L2: L>0

	● Thus, wect and sect are both covered by using an empty list and a nonempty ○ This is already done ● Wrect and Srect would involve L < 0 but this is not possible ● Wrect and Srect are covered by testing of a null variable list

Test	Boundary Value Analysis
testGet	● Variables: ○ List length L ○ Index I ● Test Set ○ BVL:{nom:L=0,nom:L>0} ○ BVI:{min-:I=-1,min:I=0,min+:I=1,nom:1<I<L, max-: I = L - 2, max: I = L - 1, max+: I = L} ● BVA (basic)

ID	L	I
BVA (basic) 1	L>0	0
BVA (basic) 2	L>0	1
BVA (basic) 3	L>0	nom
BVA (basic) 4	L>0	L-2
BVA (basic) 5	L > 0	L - 1

● BVA (robust)

ID	L	I
BVA (robust) 1	L>0	-1
BVA (robust) 2	L>0	0
BVA (robust) 3	L>0	1
BVA (robust) 4	L>0	nom
BVA (robust) 5	L>0	L-2
BVA (robust) 6	L>0	L-1
BVA (robust) 7	L>0	L

<table>
<tr><th>Test</th><th>Decision Table</th></tr>
<tr><td>testAdd</td><td>

Conditions

C1:L<=0?
C2: Object can be inserted?

Actions

A1: Object is inserted

Table

<table>
<tr><th>Test Case ID</th><th>C1</th><th>C2</th><th>Expected Output</th></tr>
<tr><td>TC1</td><td>Y/T</td><td>-</td><td>-</td></tr>
<tr><td>TC2</td><td>N/F</td><td>Y/T</td><td>Object inserted</td></tr>
<tr><td>TC3</td><td>N/F</td><td>N/F</td><td>No change</td></tr>
</table>
</td></tr>
<tr><td>testEquals</td><td>

Conditions

C1:L<=0?
C2: Objects are equal?

Actions

A1: Objects are equal

Table

<table>
<tr><th>Test Case ID</th><th>C1</th><th>C2</th><th>Expected Output</th></tr>
<tr><td>TC1</td><td>Y/T</td><td>-</td><td>-</td></tr>
<tr><td>TC2</td><td>N/F</td><td>Y/T</td><td>True</td></tr>
<tr><td>TC3</td><td>N/F</td><td>N/F</td><td>False</td></tr>
</table>
</td></tr>
<tr><td>testGet</td><td>

Conditions

C1:L<=0?
C2: Index in range?

Actions

A1: Object is gotten

Table

<table>
<tr><th>Test Case ID</th><th>C1</th><th>C2</th><th>Expected Output</th></tr>
<tr><td>TC1</td><td>Y/T</td><td>-</td><td>-</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td>
<table>
<tr><td>TC2</td><td>N/F</td><td>Y/T</td><td>Object at index</td></tr>
<tr><td>TC3</td><td>N/F</td><td>N/F</td><td>Error</td></tr>
</table>
</td></tr>
<tr><td>testIndexOf</td><td>

Conditions

C1:L<=0?
C2: Object in list?

Actions

A1: Index is gotten

Table

<table>
<tr><td>Test Case ID</td><td>C1</td><td>C2</td><td>Expected Output</td></tr>
<tr><td>TC1</td><td>Y/T</td><td>-</td><td>-</td></tr>
<tr><td>TC2</td><td>N/F</td><td>Y/T</td><td>Index of object</td></tr>
<tr><td>TC3</td><td>N/F</td><td>N/F</td><td>-1</td></tr>
</table>
</td></tr>
<tr><td>testRemove</td><td>

Conditions

C1:L<=0?
C2: Object in list?

Actions

A1: Object is removed

Table

<table>
<tr><td>Test Case ID</td><td>C1</td><td>C2</td><td>Expected Output</td></tr>
<tr><td>TC1</td><td>Y/T</td><td>-</td><td>-</td></tr>
<tr><td>TC2</td><td>N/F</td><td>Y/T</td><td>Object removed</td></tr>
<tr><td>TC3</td><td>N/F</td><td>N/F</td><td>No change</td></tr>
</table>
</td></tr>
<tr><td>testSublist</td><td>

Conditions

C1:L<=0?
C2: Start index in range?
C3: End index in range?

Actions

A1: Sublist is made

Table

<table>
<tr><td>Test Case ID</td><td>C1</td><td>C2</td><td>C3</td><td>Expected Output</td></tr>
<tr><td>TC1</td><td>Y/T</td><td>-</td><td>-</td><td>-</td></tr>
</table>
</td></tr>
</table>

	TC2	N/F	Y/T	Y/T	Sublist
	TC3	N/F	Y/T	N/F	Error
	TC4	N/F	N/F	Y/T	Error
	TC5	N/F	N/F	N/F	Error