Package utility

Class ArrayList<E>

java.lang.Object
    utility.ArrayList<E>

All Implemented Interfaces:

`List<E>`

```
public class ArrayList<E>
extends java.lang.Object
implements List<E>
```

Field Summary

Fields

Modifier and Type	Field	Description
`static int`	`DEFAULT_CAPACITY`	

Constructor Summary

Constructors

Constructor	Description
`ArrayList()`	creates array list object
`ArrayList(int capacity)`	creates array list object for a specific capacity

Method Summary

All Methods **Instance Methods** **Concrete Methods**

Modifier and Type	Method	Description
`void`	`add(int index, E item)`	inserts the item at the given index in the list.
`boolean`	`add(E item)`	appends the item specified to the end of the list.
`void`	`clear()`	clears list of all elements, return size back to zero.
`boolean`	`contains (E item)`	searches for an item and returns true if in the array,

Modifier and Type	Method	Description
`void`	`ensureCapacity(int capacity)`	doubles the capacity of the underlying array, to ensure that it can hold the number of elements specified by the capacity requested.
`E`	`get(int index)`	returns the item at the specified position in the list.
`int`	`indexOf(E item)`	searches for an item and returns the first occurrence in the array, otherwise returns -1, if NOT found.
`boolean`	`isEmpty()`	returns true, if the list is empty,
`Iterator<E>`	`iterator()`	returns an object used to traverse the elements in list
`E`	`remove(int index)`	removes the item at the given index in the list.
`boolean`	`remove(E item)`	removes the first occurrence of the specified item from the list, if present.
`E`	`set(int index, E item)`	replaces the item at the specified position with the one passed.
`int`	`size()`	returns the number of the elements in the list.
`java.lang.String`	`toString()`	displays the contents of the list.

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Field Details

DEFAULT_CAPACITY

```
public static final int DEFAULT_CAPACITY
```

See Also:

Constant Field Values

Constructor Details

ArrayList

```
public ArrayList()
```

creates array list object

ArrayList

```
public ArrayList(int capacity)
```

creates array list object for a specific capacity

Parameters:

`capacity` - of the maximum list.

Method Details

size

```
public int size()
```

returns the number of the elements in the list.

Specified by:

`size` in interface `List<E>`

Returns:

size of the list.

add

```
public boolean add(E item)
```

appends the item specified to the end of the list.

Specified by:

`add` in interface `List<E>`

Parameters:

`item` - in the list

Returns:

boolean value if successful.

add

```
public void add(int index,
                E item)
```

inserts the item at the given index in the list.

Specified by:

`add` in interface `List<E>`

Parameters:

`index` - given in the list.

`item` - in the list.

clear

```
public void clear()
```

clears list of all elements, return size back to zero.

Specified by:

`clear` in interface `List<E>`

get

```
public E get(int index)
```

returns the item at the specified position in the list.

Specified by:

`get` in interface `List<E>`

Parameters:

`index` - of item in list.

Returns:

item at index.

remove

```
public E remove(int index)
```

removes the item at the given index in the list.

Specified by:

`remove` in interface `List<E>`

Parameters:

`index` - of item in list.

Returns:

old value in list.

remove

```
public boolean remove(E item)
```

removes the first occurrence of the specified item from the list, if present.

Specified by:

`remove` in interface `List<E>`

Parameters:

`item` - to remove from list.

Returns:

boolean value.

set

```
public E set(int index,
             E item)
```

replaces the item at the specified position with the one passed.

Specified by:

`set` in interface `List<E>`

Parameters:

`index` - to replace list item.

`item` - that replaces one in list.

Returns:

old item.

indexOf

```
public int indexOf(E item)
```

searches for an item and returns the first occurrence in the array, otherwise returns `-1`, if NOT found.

Specified by:

`indexOf` in interface `List<E>`

Parameters:

`item` - to search for in list.

Returns:

location of item, if found.

isEmpty

```
public boolean isEmpty()
```

returns true, if the list is empty,

Specified by:

`isEmpty` in interface `List<E>`

Returns:

boolean value

iterator

```
public Iterator<E> iterator()
```

returns an object used to traverse the elements in list

Specified by:

`iterator` in interface `List<E>`

Returns:

iterator for list

contains

```
public boolean contains(E item)
```

searches for an item and returns true if in the array,

Specified by:

`contains` in interface `List<E>`

Parameters:

`item` - to search for in list.

Returns:

boolean value.

ensureCapacity

```
public void ensureCapacity(int capacity)
```

doubles the capacity of the underlying array, to ensure that it can hold the number of elements specified by the capacity requested.

Parameters:

`capacity` -

toString

```
public java.lang.String toString()
```

displays the contents of the list.

Overrides:

`toString` in class `java.lang.Object`

Returns:

list

Package utility

Class LinkedList<E>

java.lang.Object
    utility.LinkedList<E>

All Implemented Interfaces:

`List<E>`

```
public class LinkedList<E>
extends java.lang.Object
implements List<E>
```

Nested Class Summary

Nested Classes

Modifier and Type	Class	Description
`class`	`LinkedList.LinkedIterator`	

Constructor Summary

Constructors

Constructor	Description
`LinkedList()`	creates linked list object

Method Summary

All Methods **Instance Methods** **Concrete Methods**

Modifier and Type	Method	Description
`void`	`add(int index, E item)`	inserts the item at the given index in the list.
`boolean`	`add(E item)`	appends the item specified to the end of the list.
`void`	`clear()`	clears list of all elements, return size back to zero.
`boolean`	`contains(E item)`	searches for an item and returns true if in the array,

Modifier and Type	Method	Description
`E`	`get(int index)`	returns the item at the specified position in the list.
`int`	`indexOf(E item)`	searches for an item and returns the first occurrence in the array, otherwise returns -1, if NOT found.
`boolean`	`isEmpty()`	returns true, if the list is empty,
`Iterator<E>`	`iterator()`	returns an object used to traverse the elements in list
`E`	`remove(int index)`	removes the item at the given index in the list.
`boolean`	`remove(E item)`	removes the first occurrence of the specified item from the list, if present.
`E`	`set(int index, E item)`	replaces the item at the specified position with the one passed.
`int`	`size()`	returns the number of the elements in the list.
`java.lang.String`	`toString()`	displays the contents of the list.

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Details

LinkedList

`public LinkedList()`

creates linked list object

Method Details

add

`public boolean add(E item)`

appends the item specified to the end of the list.

Specified by:

`add` in interface `List<E>`

Parameters:

`item` - in the list

Returns:

boolean value if successful.

add

```
public void add(int index,
                E item)
```

inserts the item at the given index in the list.

Specified by:

`add` in interface `List<E>`

Parameters:

`index` - given in the list.

`item` - in the list.

clear

```
public void clear()
```

clears list of all elements, return size back to zero.

Specified by:

`clear` in interface `List<E>`

contains

```
public boolean contains(E item)
```

searches for an item and returns true if in the array,

Specified by:

`contains` in interface `List<E>`

Parameters:

`item` - to search for in list.

Returns:

boolean value.

get

```
public E get(int index)
```

returns the item at the specified position in the list.

Specified by:

`get` in interface `List<E>`

Parameters:

`index` - of item in list.

Returns:

item at index.

indexOf

```
public int indexOf(E item)
```

searches for an item and returns the first occurrence in the array, otherwise returns -1, if NOT found.

Specified by:

`indexOf` in interface `List<E>`

Parameters:

`item` - to search for in list.

Returns:

location of item, if found.

isEmpty

```
public boolean isEmpty()
```

returns true, if the list is empty,

Specified by:

`isEmpty` in interface `List<E>`

Returns:

boolean value

iterator

```
public Iterator<E> iterator()
```

returns an object used to traverse the elements in list

Specified by:

`iterator` in interface `List<E>`

Returns:

iterator for list

remove

```
public E remove(int index)
```

removes the item at the given index in the list.

Specified by:

`remove` in interface `List<E>`

Parameters:

`index` - of item in list.

Returns:

old value in list.

remove

```
public boolean remove(E item)
```

removes the first occurrence of the specified item from the list, if present.

Specified by:

`remove` in interface `List<E>`

Parameters:

`item` - to remove from list.

Returns:

boolean value.

set

```
public E set(int index,
             E item)
```

replaces the item at the specified position with the one passed.

Specified by:

`set` in interface `List<E>`

Parameters:

`index` - to replace list item.

`item` - that replaces one in list.

Returns:

old item.

size

```
public int size()
```

returns the number of the elements in the list.

Specified by:

`size` in interface `List<E>`

Returns:

size of the list.

toString

```
public java.lang.String toString()
```

displays the contents of the list.

Overrides:

`toString` in class `java.lang.Object`

Returns:

string representation of list

Package utility

Class MyQueue<E>

java.lang.Object
    utility.MyQueue<E>

```
public class MyQueue<E>
extends java.lang.Object
```

Constructor Summary

Constructors

Constructor	Description
`MyQueue()`	

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method	Description
`boolean`	`add(E item)`	
`boolean`	`isEmpty()`	returns true, if the list is empty,
`E`	`peek()`	
`E`	`remove()`	
`int`	`size()`	returns the number of the elements in the list.
`java.lang.String`	`toString()`	displays the contents of the list.

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Details

MyQueue

```
public MyQueue()
```

Method Details

add

```
public boolean add(E item)
```

isEmpty

```
public boolean isEmpty()
```

returns true, if the list is empty,

Returns:

boolean value

peek

```
public E peek()
```

remove

```
public E remove()
```

size

```
public int size()
```

returns the number of the elements in the list.

Returns:

size of queue.

toString

```
public java.lang.String toString()
```

displays the contents of the list.

Overrides:

`toString` in class `java.lang.Object`

Returns:

queue

Package utility

Class MyStack<E>

java.lang.Object
  utility.MyStack<E>

```
public class MyStack<E>
extends java.lang.Object
```

Constructor Summary

Constructors

Constructor	Description
MyStack()	

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method	Description
boolean	isEmpty()	returns true, if the list is empty,
E	peek()	
E	pop()	
E	push(E item)	
int	size()	returns the number of the elements in the list.
java.lang.String	toString()	displays the contents of the list.

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Details

MyStack

```
public MyStack()
```

Method Details

peek

```
public E peek()
```

pop

```
public E pop()
```

push

```
public E push(E item)
```

isEmpty

```
public boolean isEmpty()
```

returns true, if the list is empty,

Returns:

boolean value

size

```
public int size()
```

returns the number of the elements in the list.

Returns:

size of stack.

toString

```
public java.lang.String toString()
```

displays the contents of the list.

Overrides:

`toString` in class `java.lang.Object`

Returns:

stack

Package utility

Interface Iterator<E>

All Known Implementing Classes:

`LinkedList.LinkedIterator`

`public interface Iterator<E>`

Method Summary

All Methods | **Instance Methods** | **Abstract Methods**

Modifier and Type	Method	Description
`boolean`	`hasNext()`	
`E`	`next()`	
`void`	`remove()`	

Method Details

hasNext

`boolean hasNext()`

next

`E next()`

remove

`void remove()`

Package utility

Interface List<E>

All Known Implementing Classes:

`ArrayList`, `LinkedList`

```
public interface List<E>
```

Method Summary

All Methods | **Instance Methods** | **Abstract Methods**

Modifier and Type	Method	Description
`void`	`add(int index, E item)`	
`boolean`	`add(E item)`	
`void`	`clear()`	
`boolean`	`contains(E item)`	
`E`	`get(int index)`	
`int`	`indexOf(E item)`	
`boolean`	`isEmpty()`	
`Iterator<E>`	`iterator()`	
`E`	`remove(int index)`	
`boolean`	`remove(E item)`	
`E`	`set(int index, E item)`	
`int`	`size()`	

Method Details

add

```
boolean add(E item)
```

add

```
void add(int index,
         E item)
```

clear

```
void clear()
```

contains

```
boolean contains(E item)
```

get

```
E get(int index)
```

indexOf

```
int indexOf(E item)
```

isEmpty

```
boolean isEmpty()
```

remove

```
E remove(int index)
```

remove

```
boolean remove(E item)
```

set

```
E set(int index,
      E item)
```

size

```
int size()
```

iterator

```
Iterator<E> iterator()
```

Package testing

Class ArrayListTest

java.lang.Object
    testing.ArrayListTest

```
public class ArrayListTest
extends java.lang.Object
```

Constructor Summary

Constructors

Constructor	Description
`ArrayListTest()`	

Method Summary

All Methods **Static Methods** **Concrete Methods**

Modifier and Type	Method	Description
`static void`	`intro()`	
`static void`	`libraryVersionTest()`	
`static void`	`main(java.lang.String[] args)`	
`static void`	`myVersionTest()`	

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Details

ArrayListTest

```
public ArrayListTest()
```

Method Details

intro

```
public static void intro()
```

libraryVersionTest

```
public static void libraryVersionTest()
```

myVersionTest

```
public static void myVersionTest()
```

main

```
public static void main(java.lang.String[] args)
```

Package testing

Class LinkedListTest

java.lang.Object
    testing.LinkedListTest

```
public class LinkedListTest
extends java.lang.Object
```

Constructor Summary

Constructors

Constructor	Description
`LinkedListTest()`	

Method Summary

All Methods **Static Methods** **Concrete Methods**

Modifier and Type	Method	Description
`static void`	`intro()`	
`static void`	`main(java.lang.String[] args)`	
`static void`	`myVersionTest()`	

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Details

LinkedListTest

```
public LinkedListTest()
```

Method Details

intro

```
public static void intro()
```

myVersionTest

```
public static void myVersionTest()
```

main

```
public static void main(java.lang.String[] args)
```

Package testing

Class LinkedTestProgram

java.lang.Object
    testing.LinkedTestProgram

```
public class LinkedTestProgram
extends java.lang.Object
```

Nested Class Summary

Nested Classes

Modifier and Type	Class	Description
`static class`	`LinkedTestProgram.Node<E>`	

Constructor Summary

Constructors

Constructor	Description
`LinkedTestProgram()`	

Method Summary

All Methods **Static Methods** **Concrete Methods**

Modifier and Type	Method	Description
`static void`	`intro()`	
`static void`	`main(java.lang.String[] args)`	
`static java.lang.String`	`print (LinkedTestProgram.Node<java.lang.String> first)`	
`static void`	`testNode()`	

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Details

LinkedTestProgram

```
public LinkedTestProgram()
```

Method Details

intro

```
public static void intro()
```

testNode

```
public static void testNode()
```

print

```
public static java.lang.String print(LinkedTestProgram.Node<java.lang.String> first)
```

main

```
public static void main(java.lang.String[] args)
```

Package testing

Class LinkedTestProgram.Node<E>

java.lang.Object
    testing.LinkedTestProgram.Node<E>

Enclosing class:

LinkedTestProgram

```
public static class LinkedTestProgram.Node<E>
extends java.lang.Object
```

Constructor Summary

Constructors

Constructor	Description
`Node(E data)`	
`Node(LinkedTestProgram.Node<E> next, E data)`	

Method Summary

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Details

Node

```
public Node(E data)
```

Node

```
public Node(LinkedTestProgram.Node<E> next,
            E data)
```

Package testing

Class MyQueueTest

java.lang.Object
    testing.MyQueueTest

```
public class MyQueueTest
extends java.lang.Object
```

Constructor Summary

Constructors

Constructor	Description
`MyQueueTest()`	

Method Summary

All Methods | Static Methods | Concrete Methods

Modifier and Type	Method	Description
`static void`	`intro()`	
`static void`	`libraryVersionTest()`	
`static void`	`main(java.lang.String[] args)`	
`static void`	`myVersionTest()`	

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Details

MyQueueTest

```
public MyQueueTest()
```

Method Details

intro

```
public static void intro()
```

libraryVersionTest

```
public static void libraryVersionTest()
```

myVersionTest

```
public static void myVersionTest()
```

main

```
public static void main(java.lang.String[] args)
```

Package testing

Class MyStackTest

java.lang.Object
    testing.MyStackTest

```
public class MyStackTest
extends java.lang.Object
```

Constructor Summary

Constructors

Constructor	Description
`MyStackTest()`	

Method Summary

All Methods **Static Methods** **Concrete Methods**

Modifier and Type	Method	Description
`static void`	`intro()`	
`static void`	`libraryVersionTest()`	
`static void`	`main(java.lang.String[] args)`	
`static void`	`myVersionTest()`	

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Details

MyStackTest

```
public MyStackTest()
```

Method Details

intro

```
public static void intro()
```

libraryVersionTest

```
public static void libraryVersionTest()
```

myVersionTest

```
public static void myVersionTest()
```

main

```
public static void main(java.lang.String[] args)
```

Package tests

Class ArrayListUnitTest

java.lang.Object
    tests.ArrayListUnitTest

```
public class ArrayListUnitTest
extends java.lang.Object
```

Constructor Summary

Constructors

Constructor	Description
`ArrayListUnitTest()`	

Method Summary

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Details

ArrayListUnitTest

```
public ArrayListUnitTest()
```

Package tests

Class LinkedListUnitTest

java.lang.Object
    tests.LinkedListUnitTest

```
public class LinkedListUnitTest
extends java.lang.Object
```

Constructor Summary

Constructors

Constructor	Description
`LinkedListUnitTest()`	

Method Summary

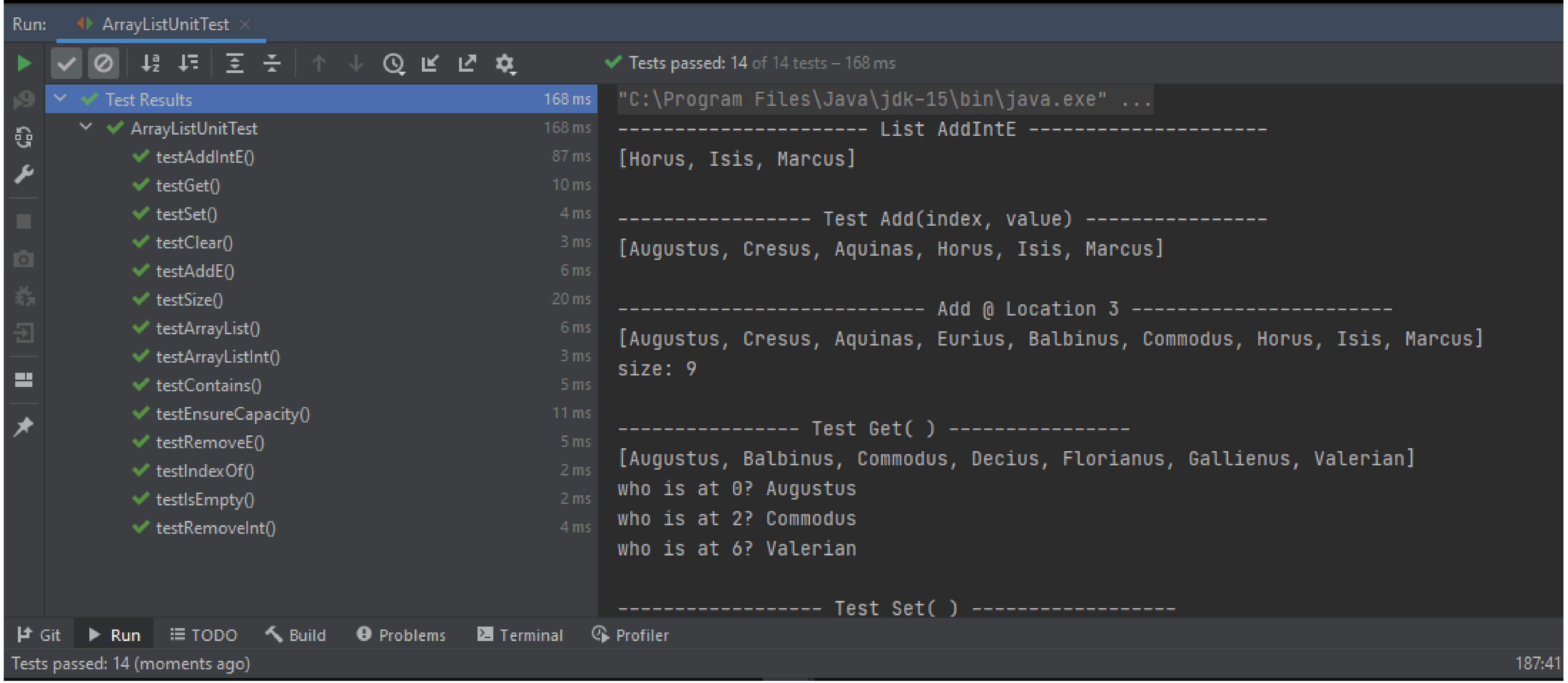
Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Details

LinkedListUnitTest

```
public LinkedListUnitTest()
```

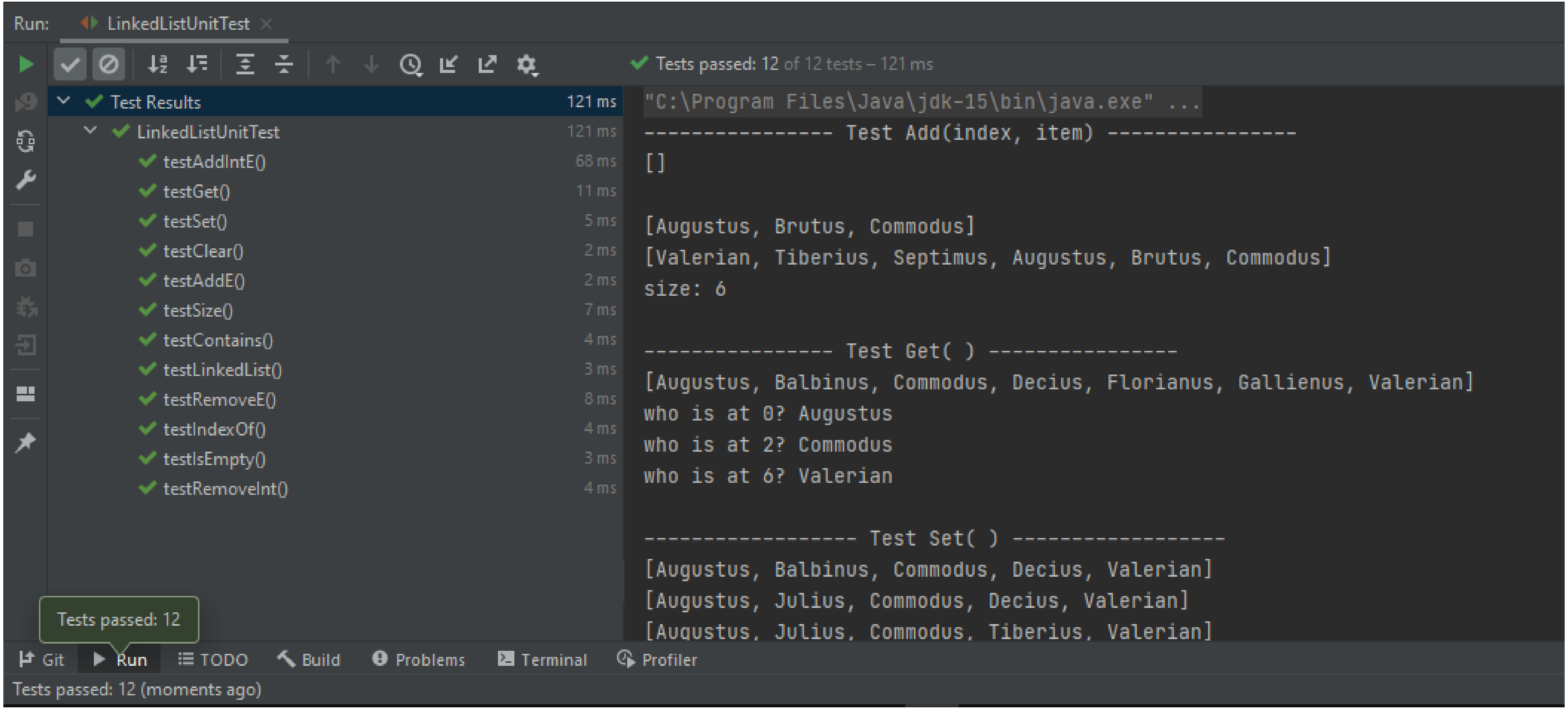


✓ Tests passed: 14 of 14 tests – 168 ms

```
✓ Test Results               168 ms
  ✓ ArrayListUnitTest        168 ms
    ✓ testAddIntE()           87 ms
    ✓ testGet()               10 ms
    ✓ testSet()                4 ms
    ✓ testClear()              3 ms
    ✓ testAddE()               6 ms
    ✓ testSize()              20 ms
    ✓ testArrayList()          6 ms
    ✓ testArrayListInt()       3 ms
    ✓ testContains()           5 ms
    ✓ testEnsureCapacity()    11 ms
    ✓ testRemoveE()            5 ms
    ✓ testIndexOf()            2 ms
    ✓ testIsEmpty()            2 ms
    ✓ testRemoveInt()          4 ms
```

```
"C:\Program Files\Java\jdk-15\bin\java.exe" ...
---------------------- List AddIntE ---------------------
[Horus, Isis, Marcus]

----------------- Test Add(index, value) ----------------
[Augustus, Cresus, Aquinas, Horus, Isis, Marcus]

--------------------------- Add @ Location 3 -----------------------
[Augustus, Cresus, Aquinas, Eurius, Balbinus, Commodus, Horus, Isis, Marcus]
size: 9

---------------- Test Get( ) ----------------
[Augustus, Balbinus, Commodus, Decius, Florianus, Gallienus, Valerian]
who is at 0? Augustus
who is at 2? Commodus
who is at 6? Valerian

------------------ Test Set( ) ------------------
```

Tests passed: 14 (moments ago)   187:41

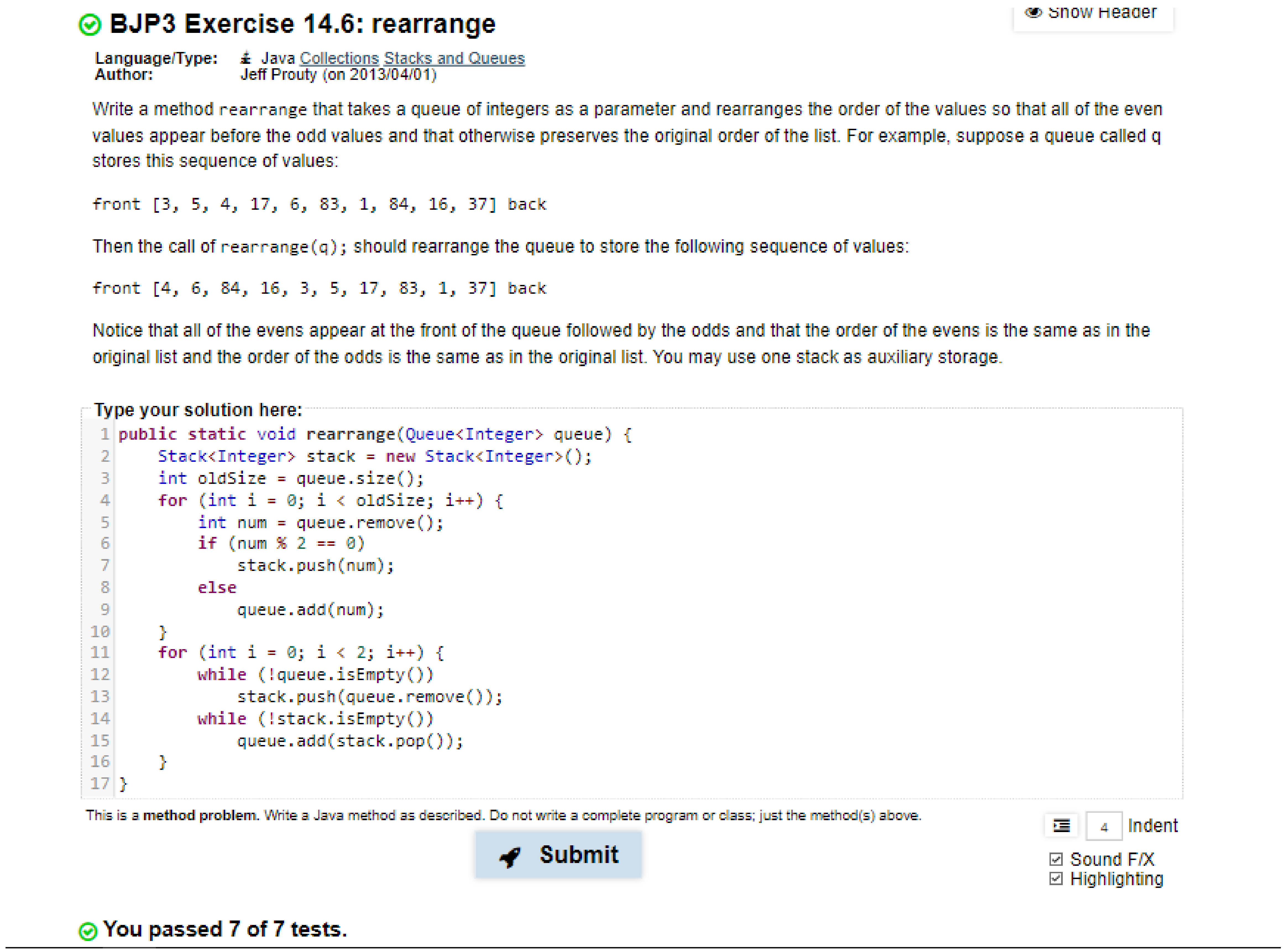


```
✓ Tests passed: 12 of 12 tests – 121 ms

✓ Test Results            121 ms
  ✓ LinkedListUnitTest    121 ms
    ✓ testAddIntE()        68 ms
    ✓ testGet()            11 ms
    ✓ testSet()             5 ms
    ✓ testClear()           2 ms
    ✓ testAddE()            2 ms
    ✓ testSize()            7 ms
    ✓ testContains()        4 ms
    ✓ testLinkedList()      3 ms
    ✓ testRemoveE()         8 ms
    ✓ testIndexOf()         4 ms
    ✓ testIsEmpty()         3 ms
    ✓ testRemoveInt()       4 ms
```

```
"C:\Program Files\Java\jdk-15\bin\java.exe" ...
---------------- Test Add(index, item) ----------------
[]

[Augustus, Brutus, Commodus]
[Valerian, Tiberius, Septimus, Augustus, Brutus, Commodus]
size: 6

---------------- Test Get( ) ----------------
[Augustus, Balbinus, Commodus, Decius, Florianus, Gallienus, Valerian]
who is at 0? Augustus
who is at 2? Commodus
who is at 6? Valerian

------------------ Test Set( ) ------------------
[Augustus, Balbinus, Commodus, Decius, Valerian]
[Augustus, Julius, Commodus, Decius, Valerian]
[Augustus, Julius, Commodus, Tiberius, Valerian]
```

Tests passed: 12

Tests passed: 12 (moments ago)



Show Header

BJP3 Exercise 14.6: rearrange

Language/Type: Java Collections Stacks and Queues
Author: Jeff Prouty (on 2013/04/01)

Write a method `rearrange` that takes a queue of integers as a parameter and rearranges the order of the values so that all of the even values appear before the odd values and that otherwise preserves the original order of the list. For example, suppose a queue called q stores this sequence of values:

```
front [3, 5, 4, 17, 6, 83, 1, 84, 16, 37] back
```

Then the call of `rearrange(q);` should rearrange the queue to store the following sequence of values:

```
front [4, 6, 84, 16, 3, 5, 17, 83, 1, 37] back
```

Notice that all of the evens appear at the front of the queue followed by the odds and that the order of the evens is the same as in the original list and the order of the odds is the same as in the original list. You may use one stack as auxiliary storage.

Type your solution here:

```java
public static void rearrange(Queue<Integer> queue) {
    Stack<Integer> stack = new Stack<Integer>();
    int oldSize = queue.size();
    for (int i = 0; i < oldSize; i++) {
        int num = queue.remove();
        if (num % 2 == 0)
            stack.push(num);
        else
            queue.add(num);
    }
    for (int i = 0; i < 2; i++) {
        while (!queue.isEmpty())
            stack.push(queue.remove());
        while (!stack.isEmpty())
            queue.add(stack.pop());
    }
}
```

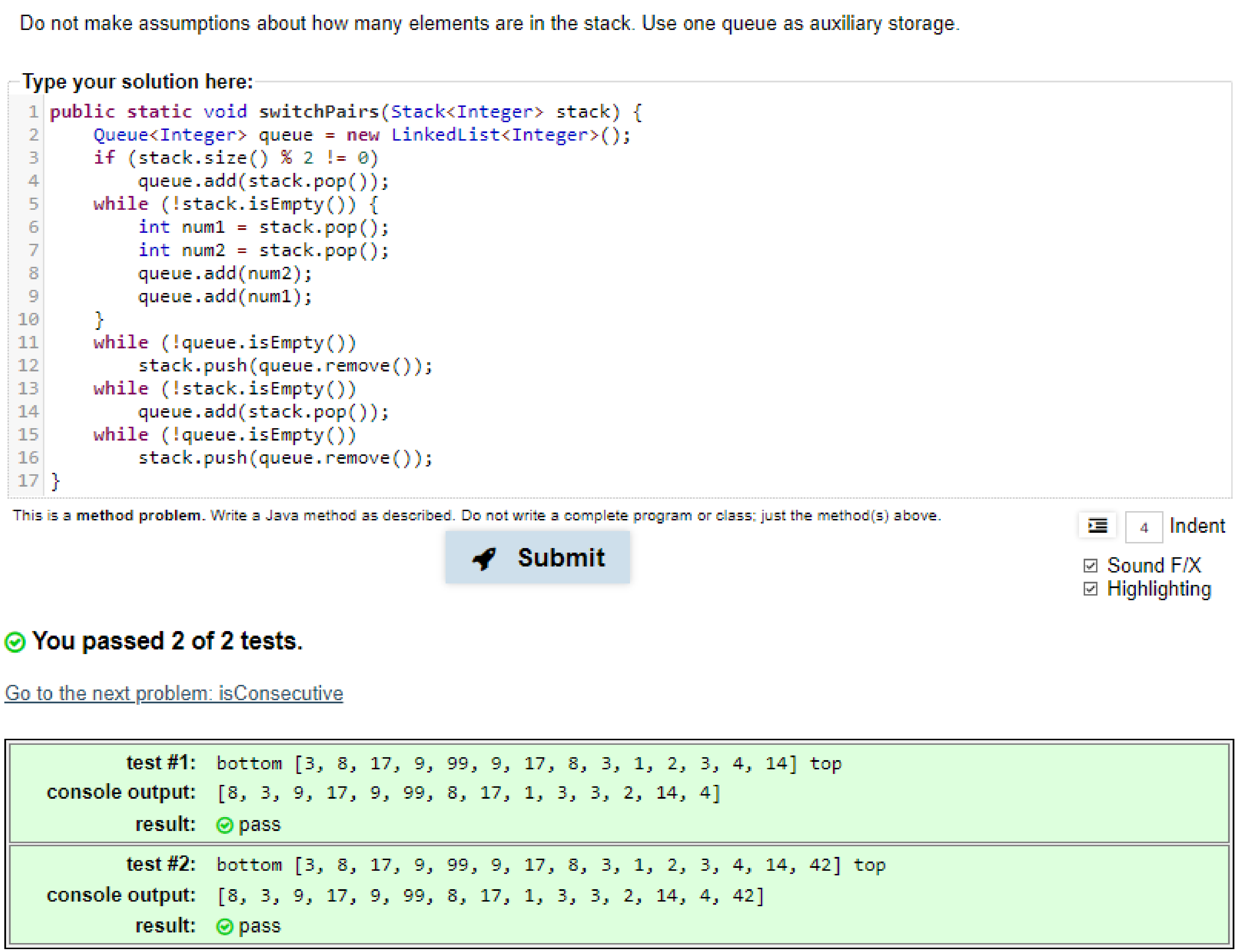
This is a **method problem**. Write a Java method as described. Do not write a complete program or class; just the method(s) above.

4 Indent

Submit

☑ Sound F/X
☑ Highlighting

You passed 7 of 7 tests.



Do not make assumptions about how many elements are in the stack. Use one queue as auxiliary storage.

Type your solution here:

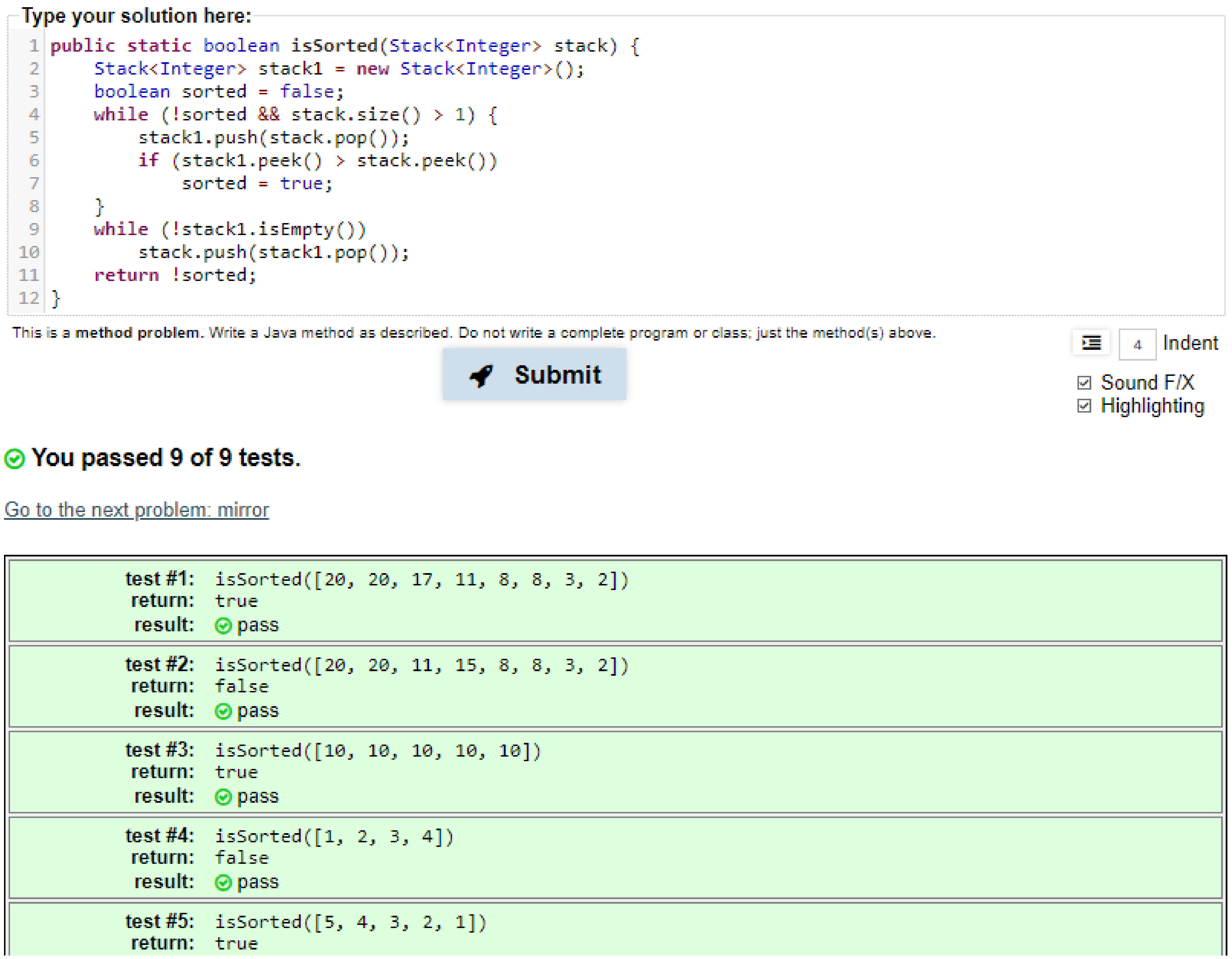
```java
public static void switchPairs(Stack<Integer> stack) {
    Queue<Integer> queue = new LinkedList<Integer>();
    if (stack.size() % 2 != 0)
        queue.add(stack.pop());
    while (!stack.isEmpty()) {
        int num1 = stack.pop();
        int num2 = stack.pop();
        queue.add(num2);
        queue.add(num1);
    }
    while (!queue.isEmpty())
        stack.push(queue.remove());
    while (!stack.isEmpty())
        queue.add(stack.pop());
    while (!queue.isEmpty())
        stack.push(queue.remove());
}
```

This is a **method problem.** Write a Java method as described. Do not write a complete program or class; just the method(s) above.

You passed 2 of 2 tests.

Go to the next problem: isConsecutive

test #1:	`bottom [3, 8, 17, 9, 99, 9, 17, 8, 3, 1, 2, 3, 4, 14] top`
console output:	`[8, 3, 9, 17, 9, 99, 8, 17, 1, 3, 3, 2, 14, 4]`
result:	✓ pass
test #2:	`bottom [3, 8, 17, 9, 99, 9, 17, 8, 3, 1, 2, 3, 4, 14, 42] top`
console output:	`[8, 3, 9, 17, 9, 99, 8, 17, 1, 3, 3, 2, 14, 4, 42]`
result:	✓ pass



Type your solution here:

```java
public static boolean isSorted(Stack<Integer> stack) {
    Stack<Integer> stack1 = new Stack<Integer>();
    boolean sorted = false;
    while (!sorted && stack.size() > 1) {
        stack1.push(stack.pop());
        if (stack1.peek() > stack.peek())
            sorted = true;
    }
    while (!stack1.isEmpty())
        stack.push(stack1.pop());
    return !sorted;
}
```

This is a **method problem.** Write a Java method as described. Do not write a complete program or class; just the method(s) above.

✓ **You passed 9 of 9 tests.**

Go to the next problem: mirror

```
test #1:  isSorted([20, 20, 17, 11, 8, 8, 3, 2])
 return:  true
 result:  ✓ pass

test #2:  isSorted([20, 20, 11, 15, 8, 8, 3, 2])
 return:  false
 result:  ✓ pass

test #3:  isSorted([10, 10, 10, 10, 10])
 return:  true
 result:  ✓ pass

test #4:  isSorted([1, 2, 3, 4])
 return:  false
 result:  ✓ pass

test #5:  isSorted([5, 4, 3, 2, 1])
 return:  true
```