/\*\*\* JUnit imports \*\*\*/

// We will use the BeforeEach and Test annotation types to mark methods in

// our test class.

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.Test;

// The Assertions class that we import from here includes assertion methods like assertEquals()

// which we will used in test1000Inserts().

import static org.junit.jupiter.api.Assertions.assertEquals;

// More details on each of the imported elements can be found here:

// https://junit.org/junit5/docs/current/api/org.junit.jupiter.api/org/junit/jupiter/api/package-summary.html

/\*\*\* JUnit imports end \*\*\*/

public class TestMyList {

protected ListADT<Integer> \_instance = null;

//BeforeEach annotation makes a method invocked automatically

//before each test

@BeforeEach

public void createInstane() {

\_instance = new MyList<Integer>();

}

//The @Test annotation allows JUnit to recognize its following

//method as a test method

@Test

public void test1000Inserts() {

// This tests inserts 1000 integers into the list and then

// checks if they're in the list at the expected index

for (int i = 0; i < 1000; i++) {

\_instance.add(i);

}

for (int i = 0; i < 1000; i++) {

assertEquals(i, \_instance.get(i));

}

}

@Test

public void testInsertSize() {

// TODO: Complete this test. The test should insert 10 integers, and check if the .size()

// method of ListADT returns the correct result after inserting each one of them into the list.

for (int i = 0; i < 10; i++) {

\_instance.add(i);

assertEquals(i + 1, \_instance.size());

}

}

// TODO: Write a third test method that

// 1) inserts 10 integers into the list

// 2) after all insertions are done, removes them

// and checks after each remove if the .size() method of ListADT returns the correct result.

@Test

public void testInsertRemoveSize(){

for (int i = 0; i < 10; i++) {

\_instance.add(i);

}

for (int i = 0; i < 10; i++) {

assertEquals(10 - i, \_instance.size());

\_instance.remove(0);

}

assertEquals(0, \_instance.size());

}

}