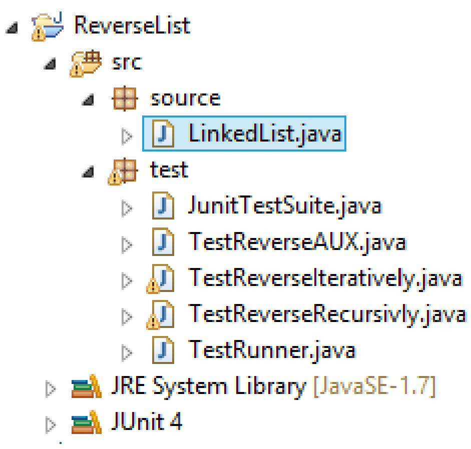
To run the test execute

TestRunner.java



Implementation of a simple singly-linked list, and two functions to reverse the order of the list Using Java and Junit 4.0.

1. An iterative reverse.
2. A recursive reverse.
3. A full suite of automated tests.

LinkedList.java

STRECTURE OF LINKEDLIST

Constructor

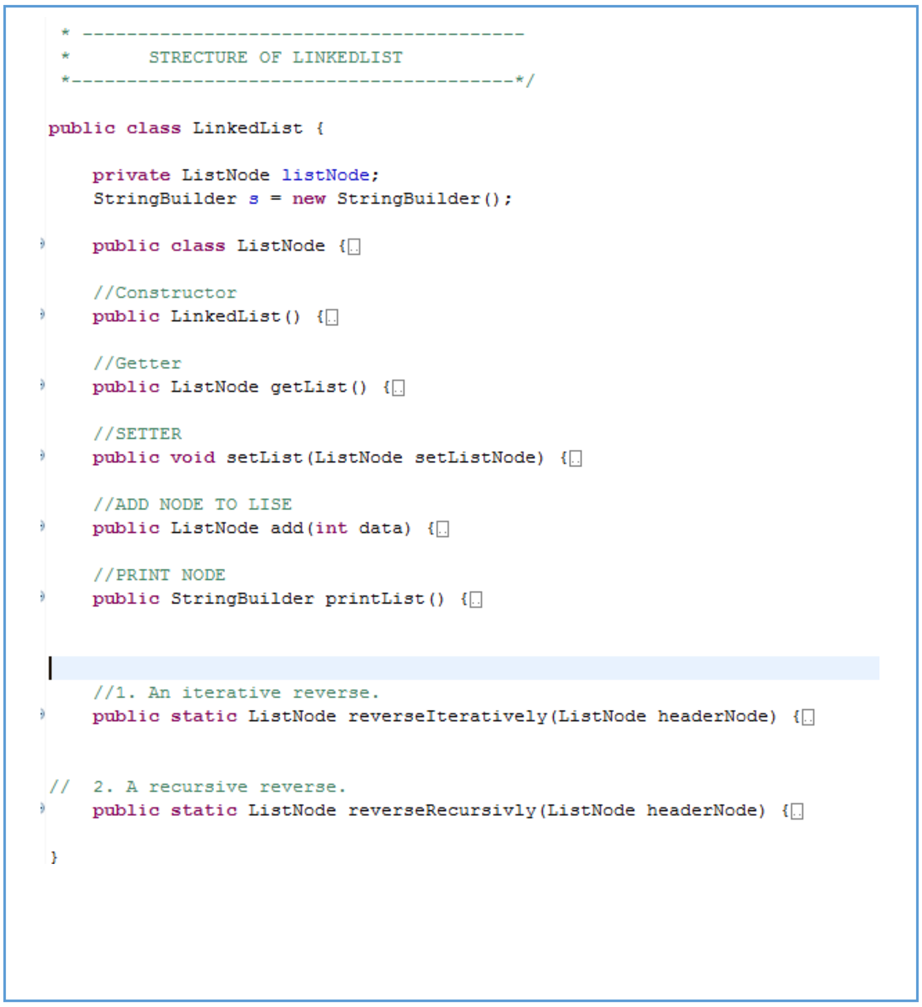
LinkedList()

Getter and Setter

ListNode getList()
void setList(ListNode setListNode)

Helpers methods

ListNode add(int data)
StringBuilder printList()



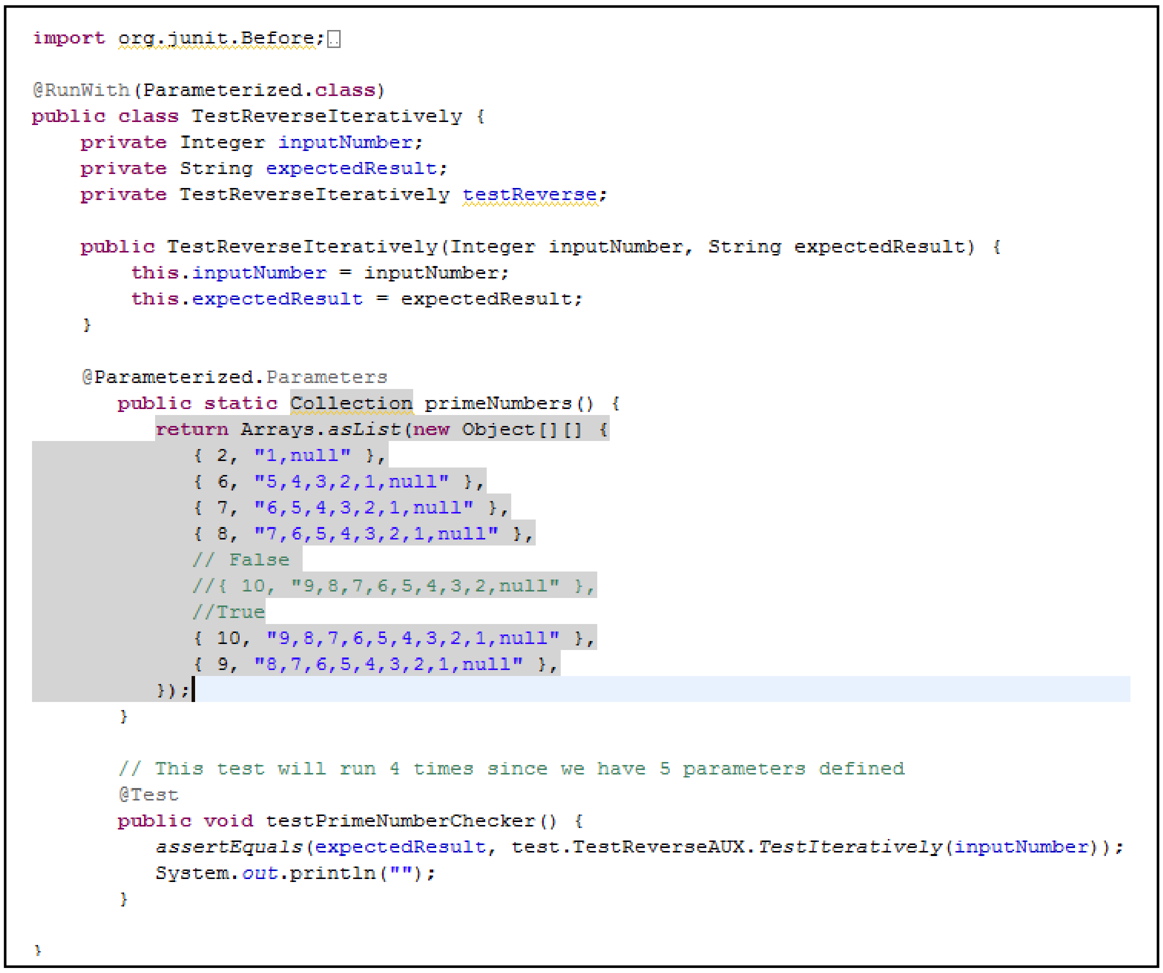
```
 * ------------------------------------------
 *       STRECTURE OF LINKEDLIST
 *------------------------------------------*/

public class LinkedList {

    private ListNode listNode;
    StringBuilder s = new StringBuilder();

    public class ListNode {

    //Constructor
    public LinkedList() {

    //Getter
    public ListNode getList() {

    //SETTER
    public void setList(ListNode setListNode) {

    //ADD NODE TO LISE
    public ListNode add(int data) {

    //PRINT NODE
    public StringBuilder printList() {


    //1. An iterative reverse.
    public static ListNode reverseIteratively(ListNode headerNode) {


//  2. A recursive reverse.
    public static ListNode reverseRecursivly(ListNode headerNode) {

}
```

Additional Functions

Reverse Functions:

An iterative reverse.
A recursive reverse.

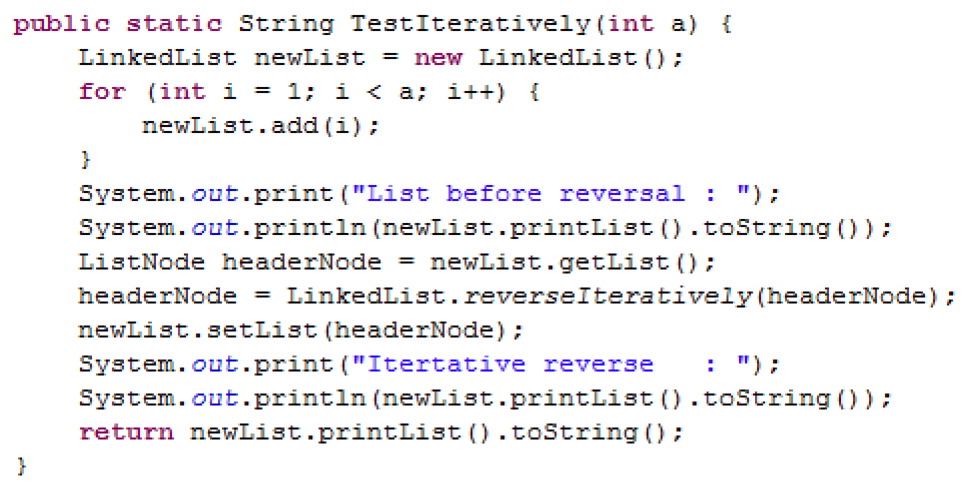
TestReverseIteratively.java



```java
import org.junit.Before;

@RunWith(Parameterized.class)
public class TestReverseIteratively {
    private Integer inputNumber;
    private String expectedResult;
    private TestReverseIteratively testReverse;

    public TestReverseIteratively(Integer inputNumber, String expectedResult) {
        this.inputNumber = inputNumber;
        this.expectedResult = expectedResult;
    }

    @Parameterized.Parameters
       public static Collection primeNumbers() {
          return Arrays.asList(new Object[][] {
             { 2, "1,null" },
             { 6, "5,4,3,2,1,null" },
             { 7, "6,5,4,3,2,1,null" },
             { 8, "7,6,5,4,3,2,1,null" },
             // False
             //{ 10, "9,8,7,6,5,4,3,2,null" },
             //True
             { 10, "9,8,7,6,5,4,3,2,1,null" },
             { 9, "8,7,6,5,4,3,2,1,null" },
          });
       }

       // This test will run 4 times since we have 5 parameters defined
       @Test
       public void testPrimeNumberChecker() {
          assertEquals(expectedResult, test.TestReverseAUX.TestIteratively(inputNumber));
          System.out.println("");
       }

}
```

Int InputNumber: the length of the list

Example in the TestReverseAUX.Java:



```
public static String TestIteratively(int a) {
    LinkedList newList = new LinkedList();
    for (int i = 1; i < a; i++) {
        newList.add(i);
    }
    System.out.print("List before reversal : ");
    System.out.println(newList.printList().toString());
    ListNode headerNode = newList.getList();
    headerNode = LinkedList.reverseIteratively(headerNode);
    newList.setList(headerNode);
    System.out.print("Itertative reverse   : ");
    System.out.println(newList.printList().toString());
    return newList.printList().toString();
}
```

The **inputNumber it's** : **a**

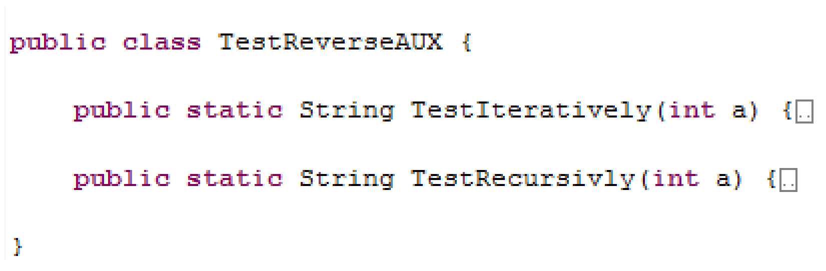
String ExpectedResult: the correct form of the reverse list.

The **ExpectedResult it's** : **"5,4,3,2,1,null"** **etc. …**

TestReverseRecursivly.java

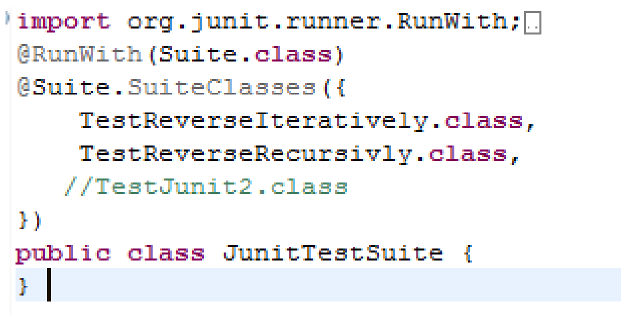
It's the same for **TestReverseRecursivly.java**

TestReverseAUX.java



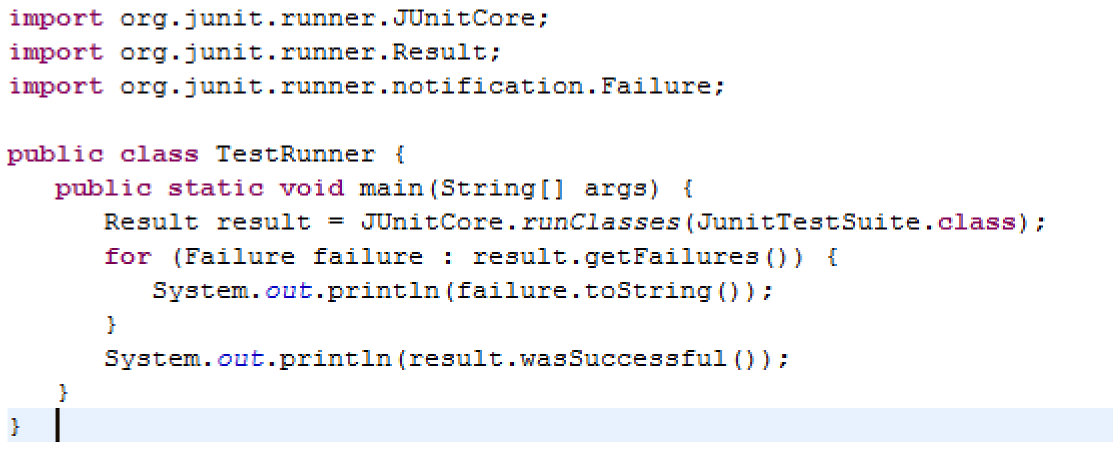
```
public class TestReverseAUX {

    public static String TestIteratively(int a) {

    public static String TestRecursivly(int a) {

}
```

JunitTestSuite.java



```java
import org.junit.runner.RunWith;
@RunWith(Suite.class)
@Suite.SuiteClasses({
    TestReverseIteratively.class,
    TestReverseRecursivly.class,
   //TestJunit2.class
})
public class JunitTestSuite {
}
```

TestRunner.java



```java
import org.junit.runner.JUnitCore;
import org.junit.runner.Result;
import org.junit.runner.notification.Failure;

public class TestRunner {
   public static void main(String[] args) {
      Result result = JUnitCore.runClasses(JunitTestSuite.class);
      for (Failure failure : result.getFailures()) {
         System.out.println(failure.toString());
      }
      System.out.println(result.wasSuccessful());
   }
}
```

References:

JUnit 4.0

http://www.mkyong.com/tutorials/junit-tutorials/

http://www.vogella.com/articles/JUnit/article.html#junit_intro

http://www.tutorialspoint.com/junit/junit_suite_test.htm