Suhyoon Bae

Lab 02

CIS 120 Data Structures

2020-02-06

Lab 02

StringLog ADT

Page(s) File

1 Code printouts – ArrayStringLog.java

2 Code printouts - LinkedStringLog.java

3-5 Junit Test

6 Extra Credit (delete method) – ArrayStringLog.java

7 Method type and the table of Big-O efficiencies.

­­

**1. Code Printouts**

**1.1 ArrayStringLog.java**

**public** **boolean** isEmpty()

// Precondition: StringLog is empty or not.

// Postcondition: Returns true if this StringLog is empty.

{

**if** (lastIndex == (log.length - 0))

**return** **true**;

**else**

**return** **false**;

}

**public** **int** howMany(String element) {

// Precondition: StringLog is not empty.

// Postcondition: Returns an int value indicating how many times element occurs in the StringLog.

**int** count = 0;

**for** (**int** i = 0; i <= lastIndex; ++i) {

**if** (log[i].equals(element)) {

count++;

}

}

**return** count;

}

**public** String smallest() {

// Precondition: StringLog is not empty.

// Postcondition: Returns a smallest value.

String smallest = log[0];

**for** (**int** i = 0; i <= lastIndex; i++) {

**if** (log[i].compareTo(smallest) < 0) {

smallest = log[i];

}

}

**return** smallest;

}

**1.2. LinkedStringLog.java**

**public** **boolean** isEmpty() {

// Precondition: StringLog is empty or not.

// Postcondition: Returns true if this StringLog is empty.

**return** (**this**.size()==0);

}

**public** **int** howMany(String element) {

// Precondition: StringLog is not empty.

// Postcondition: Returns an int value indicating how many times element occurs in the StringLog.

**int** count=0;

LLStringNode node = log;

**while** (node != **null**) {

**if** (node.getInfo() == element)

count ++;

}

**return** count;

}

**public** String smallest() {

// Precondition: StringLog is not empty.

// Postcondition: Returns a smallest value.

LLStringNode node=log;

String smallest = node.getInfo();

**while** (node!=**null**) {

**if** (node.getInfo().compareTo(smallest)<0) {

smallest=node.getInfo();

}

node = node.getLink();

}

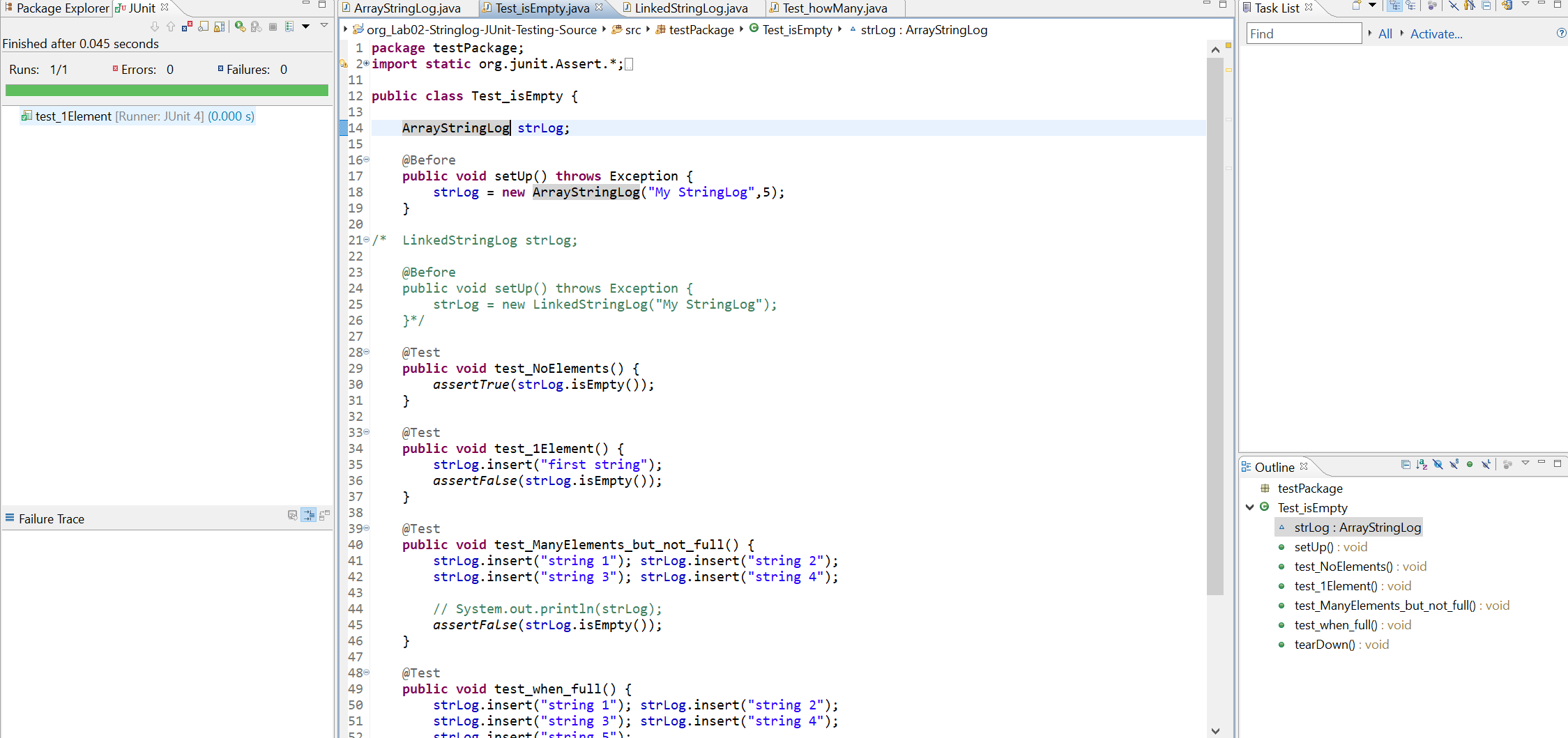
**return** smallest;

}

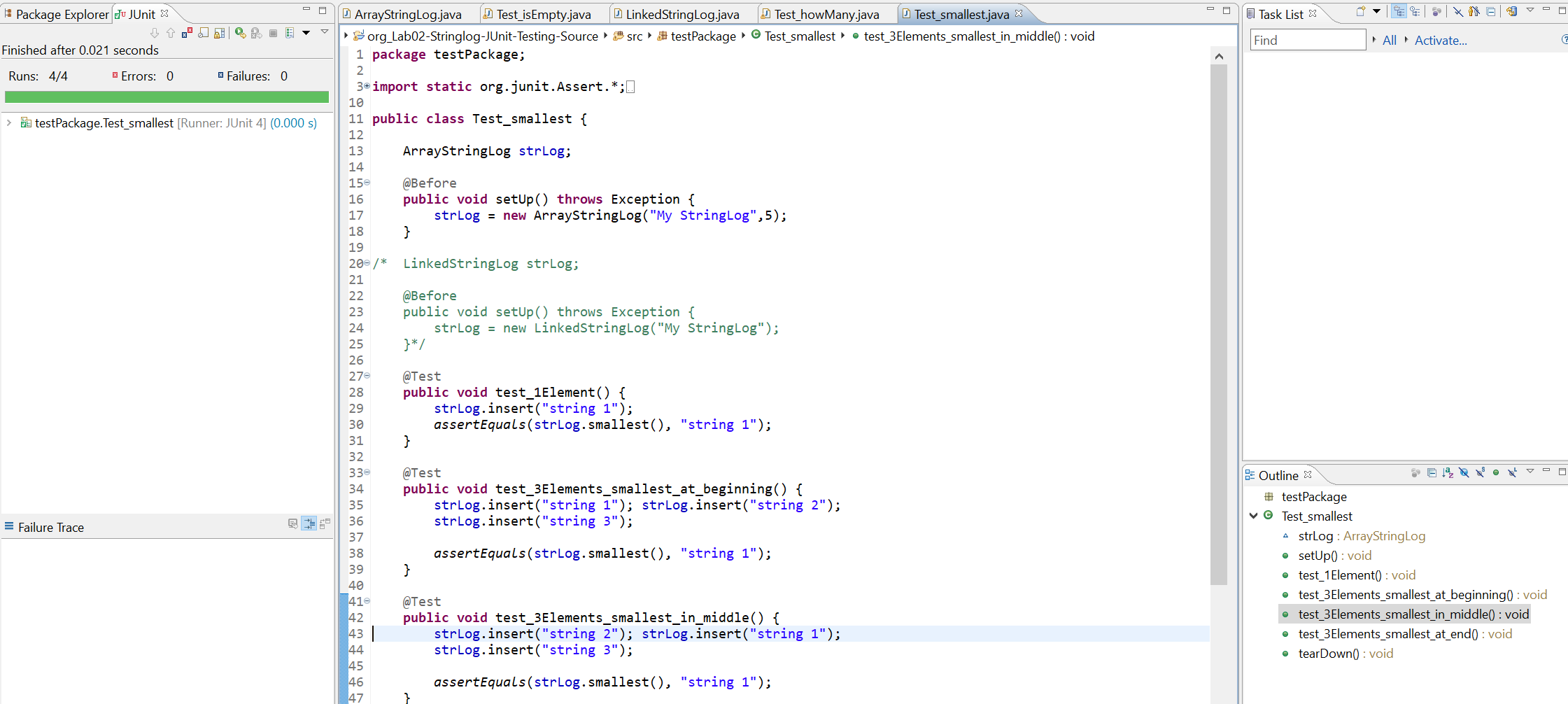
**2. Junit Test**

**2.1 ArrayStringLog.java**

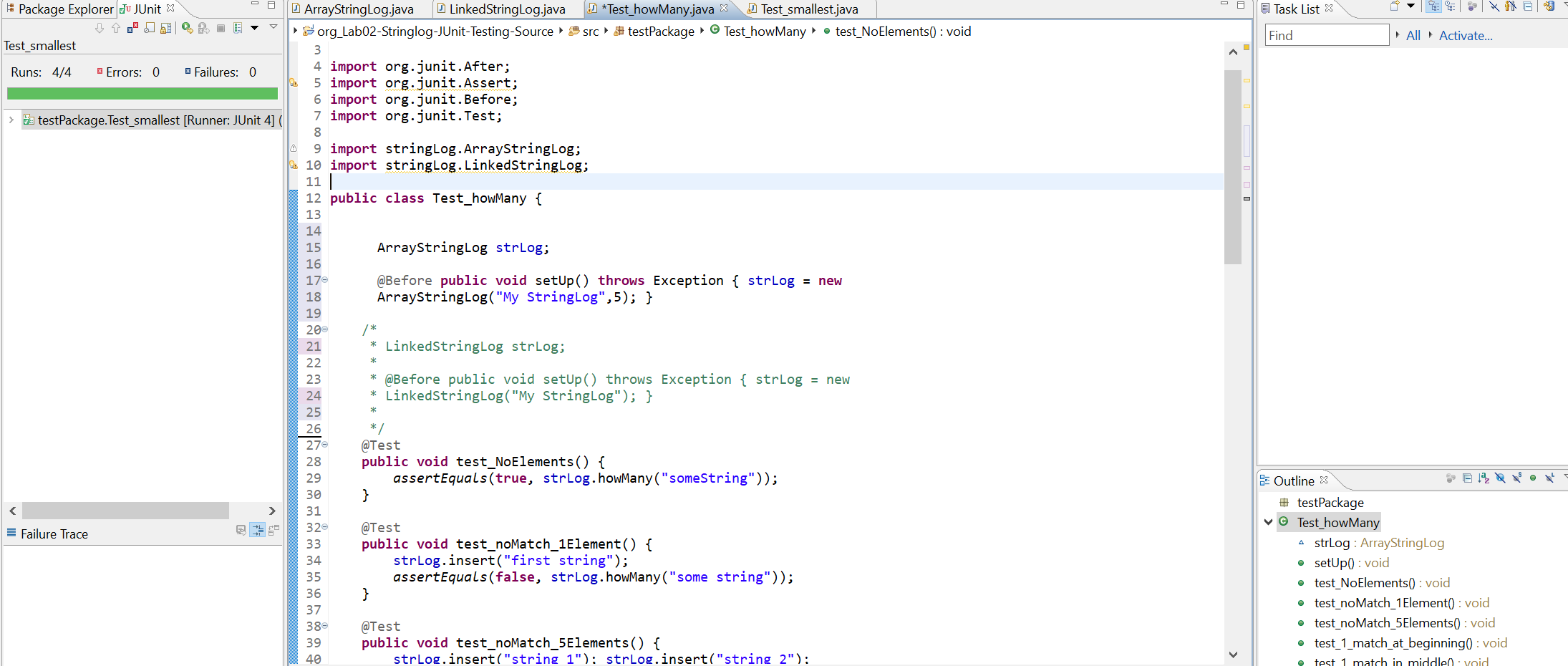
isEmpty



smallest



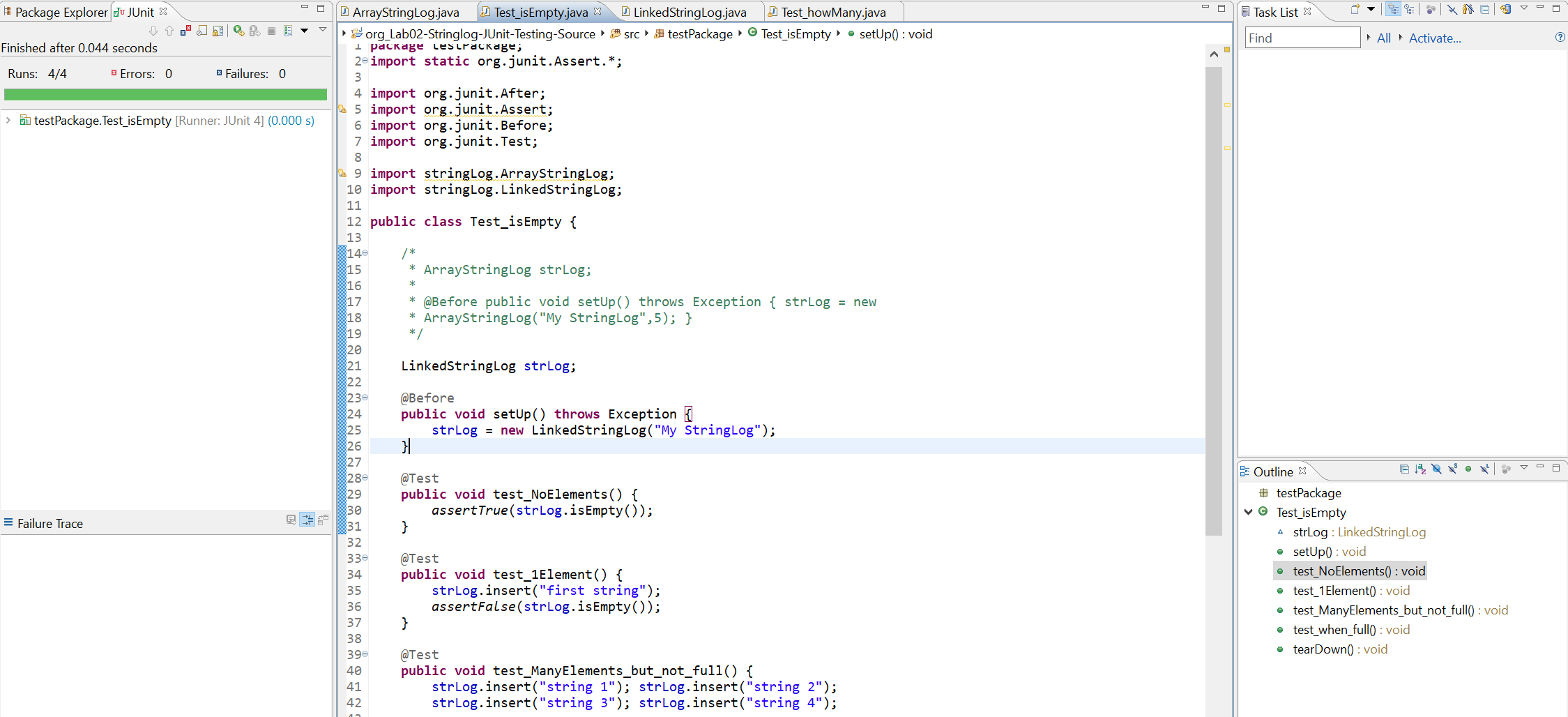
howMany



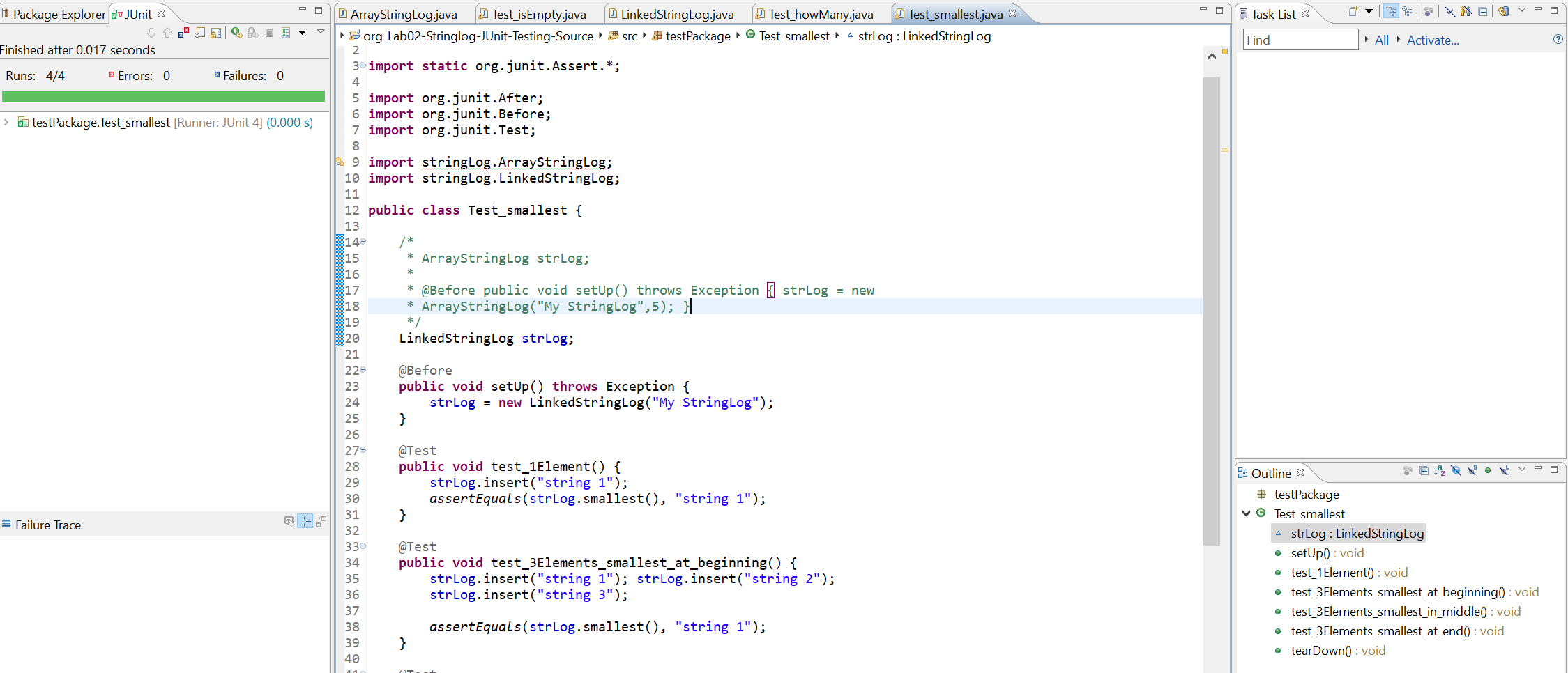


**2.1 LinkedStringLog**

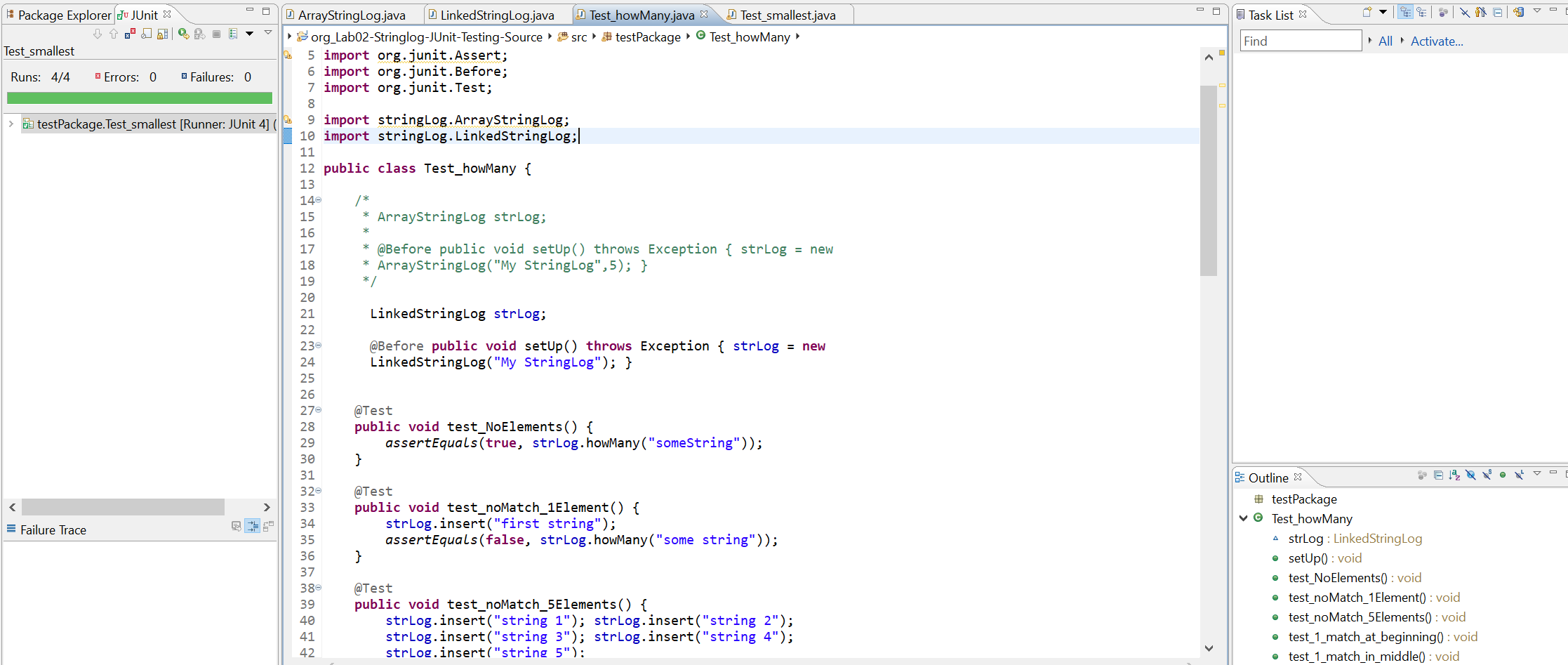
isEmpty



Smallest



howMany





**3. Extra Credit – delete**

**3.1. ArrayStringLog.java**

**public** **boolean** delete(String element)

{

**boolean** result=**false**;

**if**(!contains(element))

**return** result;

**else**

{

result=**true**;

**boolean** found=**false**;

**int** elementPos = -1;

**int** index=0;

**while** (index <= lastIndex && !found)

{

**if** (element.equalsIgnoreCase(log[index]))

{

elementPos=index;

found=**true**;

}

}

**for** (**int** i = elementPos; i < size(); i++)

{

log[i]=log[i+1];

}

lastIndex=lastIndex-1;

}

**return** result;

}

**4. method type and the table of Big-O efficiencies.**

|  |  |  |  |
| --- | --- | --- | --- |
| operation | constructor, observer, or transformer? | Big-O for Array-Based | Big-O for Reference-Based |
| constructor | constructor |  |  |
| insert | transformer | O(1) | O(1) |
| isFull | observer | O(1) | O(1) |
| size | observer | O(1) | O(N) |
| contains | observer | O(N) | O(N) |
| clear | transformer | O(N) | O(1) |
| getName | observer | O(1) | O(1) |
| toString | observer | O(N) | O(N) |
| isEmpty | observer | O(1) | O(1) |
| howMany | observer | O(N) | O(N) |
| smallest | observer | O(N) | O(N) |