# Lab 2 Looking at Java’s ArrayList and LinkedList classes

# Week beginning 19/09/2016

### The following table is from TIJ3 in Choosing between Lists in Chapter 11. It gives the time in milliseconds for multiple runs of various methods.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Type** | **Get** | **Iteration** | **Insert** | **Remove** |
| array | 172 | 516 | na | na |
| **ArrayList** | 281 | 1375 | 328 | 30484 |
| **LinkedList** | 5828 | 1047 | 109 | 16 |
| **Vector** | 422 | 1890 | 360 | 30781 |

# It is comparing get, iteration, insert, remove for ArrayList, LinkedList and Vector.

Try to do this yourself for just ArrayList and LinkedList.

There are datasets available to download from <http://www.cs.princeton.edu/introcs/data/>

Populate your ArrayList and your LinkedList with data from one of these datasets e.g. words.txt.

Sample code for reading from a file is given in filehandlingcode.txt.

To test get, use code like:

public static void testGetForList(List list) {

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

// reps is the number of repetitions

// declare reps as something large e.g. 10000

for(int j = 0; j < quantity; j++) // declare quantity // as reps/10

list.get(j); // list is your list

// (ArrayList or LinkedList)

}

}

Write a method for each list type e.g. testGetForArrayList and testGetForLinkedList – each of these method calls testGetForList with the relevant list as argument.

To test iteration, use code like:

public static void testIterateForList(List list) {

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

Iterator it = list.iterator();

while(it.hasNext())

it.next();

}

}

To test insert, use code like:

public static void testInsertForList(List list) {

int half = list.size()/2;

String s = "test";

ListIterator it = list.listIterator(half);

for(int i = 0; i < reps \* 10; i++)

it.add(s);

}

To test remove, use code like:

public static void testRemoveForList(List list) {

ListIterator it = list.listIterator(3);

while(it.hasNext()) {

it.next();

it.remove();

}

}

Time the tests on the two types of list. Look at System class for a method that give you the current time.

Check if your results are comparable to Eckels.

To produce more accurate results, use Java’s profiling tool to see the CPU time for each method.

**Profiling in Java**

See the following website:

<https://netbeans.org/kb/docs/java/profiler-intro.html>

### Look at the section Using the Profiler for the First Time.

This will show you how to run the calibration process.

### Then look at the section Analyzing CPU Performance

### This will show you how to look at CPU performance of the methods in your code. Use Advanced (Instrumented) option here.