**Lab 3 : Week beginning 1/Oct/2018**

**Profiling in IntelliJ**

(Note from Computer Services: If there is a problem with lab setup to run the following software:

Choose File->Import Settings. Then choose C:\IntelliJSettings\settings.jar)

IntelliJ does not have built-in profiling capability (NetBeans does!)

VisualVM is a free profiling tool – VisualVM 1.4.1. is available at <https://visualvm.github.io/download.html>

VisualVM Launcher is a plug-in for IntelliJ which integrates VisualVM – available at <https://plugins.jetbrains.com/plugin/7115-visualvm-launcher>

The software is already set up in the labs (hopefully!).

If not, install VisualVMLauncher as follows:

Choose File->Settings. Then choose Plugins and Install plugins from disk, etc

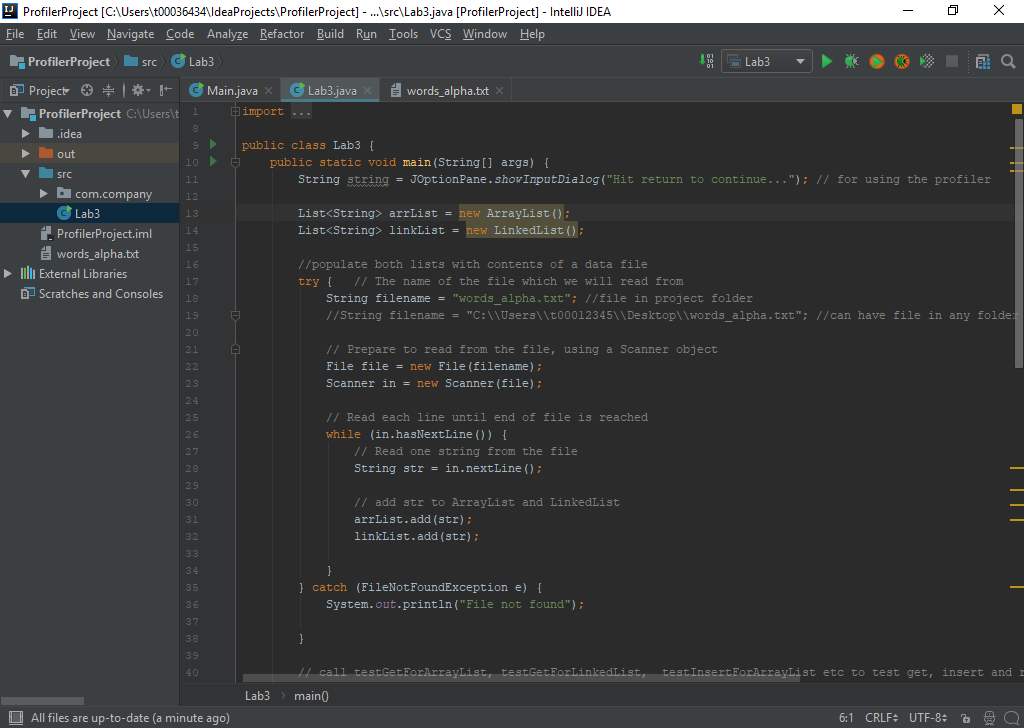
To test the profiling software, use Lab3.java (in same folder as this document). This code is code you created for Lab 1 – looking at efficiency of ArrayList and LinkedList.

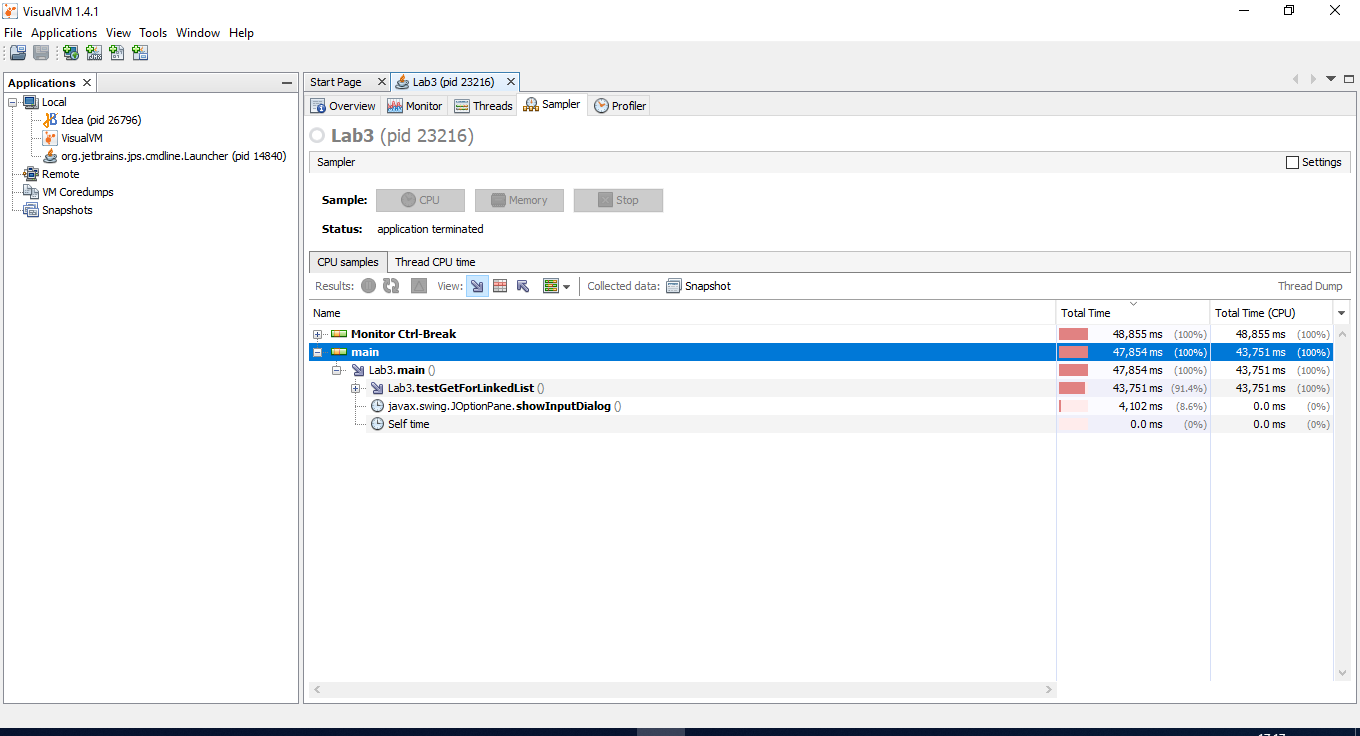
In IntelliJ, create a new project and add Lab3.java. Also add the words\_alpha.txt

Run the program in the usual way and check that it works. It should output the time for multiple “get”s for ArrayList and LinkedList. This may be **slow** to run. The time for ArrayList may be very small if you run it for less iterations.

**Running without VisualVM Launcher**

Now run with VisualVM from within IntelliJ (if VisualVM Launcher installed)



In VisualVM you will then see the time taken for testGetForArrayList (if this is very small it may not appear) and testGetForLinkedList

**Running without VisualVM Launcher**

1. Run VisualVM - visualvm\_141\bin\visualvm.exe
2. In IntelliJ, create a new project and add Lab3.java. Also add the words\_alpha.txt
3. Run the code in the usual way. Dialog box comes up. Before closing this:
   1. In VisualVM, you will now see Lab3.
   2. Right click on Lab3. Choose Sample and then choose CPU
4. Hit OK to close the dialog box.
5. In VisualVM you will then see the time taken for testGetForArrayList (if this is very small it may not appear) and testGetForLinkedList

Now add code to compare iteration, insert and remove.

Fill in the following table:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | get | | | | | iterate | | | | insert | | | | | remove | | | | |
|  | Big Oh value | Time as in Lab 1 | | Time from profiler | | Big Oh value | Time as in Lab 1 | Time from profiler | | Big Oh value | Time as in Lab 1 | | Time from profiler | | Big Oh value | Time as in Lab 1 | | Time from profiler | |
| ArrayList | O(1) | 32 ms | | 94.3 ms | | 0(n) | 610 ms | | 516 ms | 0(n) | | 297 ms | | 298 ms | 0(n) | | 10,484 ms | | 9,500 ms |
| LinkedList | O(n) | | 23,281 ms | | 16,657 ms | 0(n) | 2,265 ms | | 2,110 ms | 0(1) | | 0 ms | | 0 ms | 0(1) | | 16 ms | | 0 ms |

Are the timings consistent with the Big Oh values? Yes

Comment on results \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_