Alex Honeygosky (alexhoneygosky)

Tommy Bednar (TommyBednar)

<https://github.com/alexhoneygosky/MonkeySim>

CS 1632 - DELIVERABLE 4: Performance Testing Using VisualVM

**Summary**

In order to determine which functions we needed to refactor for this project, we used the Visual VM profiler provided to us in the write up of this deliverable. Upon running the program with Visual VM open, we determined through the CPU profiler in the software that the getFirstMonkey, stringifyResults, and generateId functions were the worst three functions based on performance. After pinpointing the “hotspots” of the program, we opened up the code to find the three functions chosen to have unnecessary code running. In each of them, the value being returned was not being affected based on the big loops, calculations, and concatenations happening.

First, the getFirstMonkey function used a long running for loop and inner for loop to return the first monkey in the monkey list. Therefore, we changed the function to just take the monkey list passed in and find and return the first monkey in the list through the use of the list get function.

Next, in the stringifyResults function, there was a very large 50000 iteration loop running that added 50000 @ symbols only to be wiped out from the addition of the printed out current round of the program. Thus, not only did we remove the pointless loop, but we also used String.format to put the string together because that also sped the program up.

Finally, in the generateId function, there were nested loops used to come up with a random base id value before adding the n int passed in. To speed this function up, we determined the base value being used by the nested for loops and just simply returned that calculated value + n and the program’s speed went up again.

When we created the pinning tests, all of them passed with the original functions, but when we used the performance-enhanced functions, two of our functions failed, but on purpose. When the getFirstMonkey was changed to return ml.get(1), the method automatically assumes the monkey in position 0 in the list is not a valid monkey, so therefore the tests that check that a valid monkey is return with one monkey in the list, fails. The other method that fails is the empty monkey list test for the getFirstMonkey function. The test is supposed to check that the observed and expected behavior both equal null, but since the list is empty, an IndexOutOfBounds exception is thrown and that is fine considering the sped up function performs correctly. In both of these cases, the tests would need to be rewritten because they are now incorrect, not the methods.

**Initial/Final Times (w/ parameter of 23)**

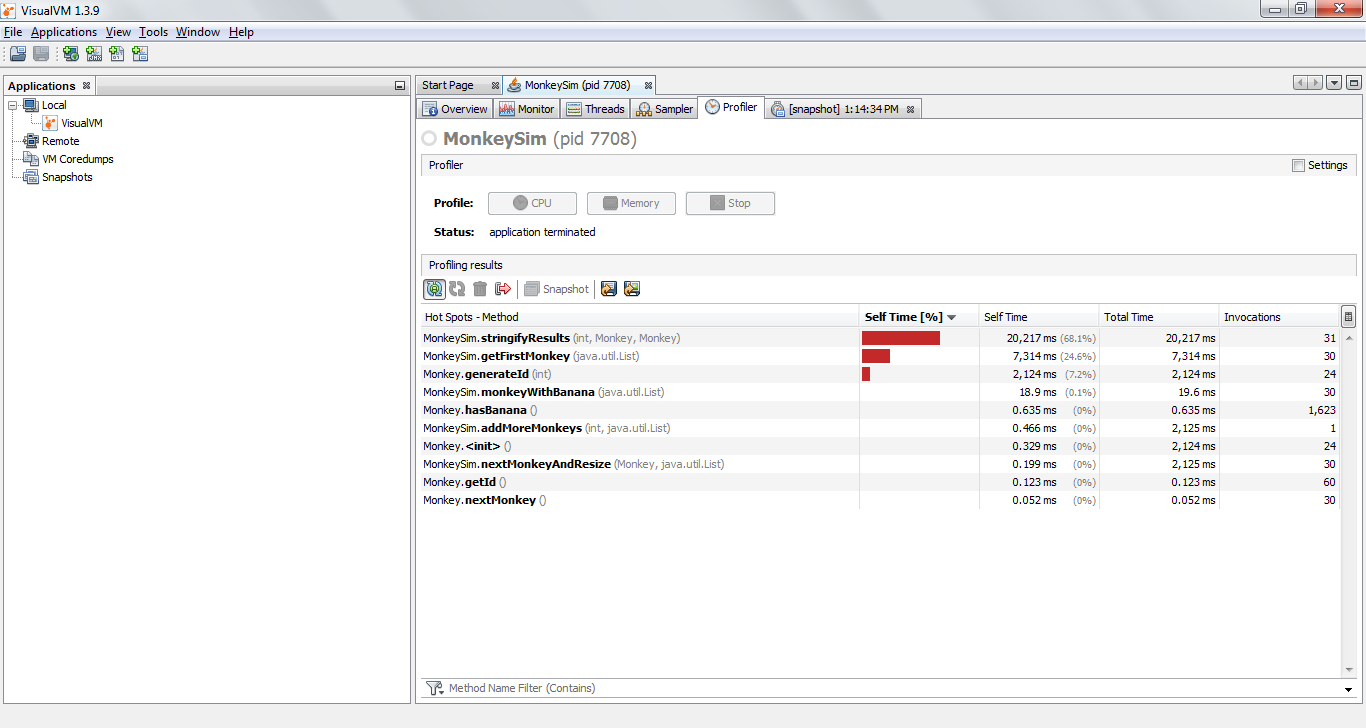
Initial Mean Time: 14.5758963 seconds

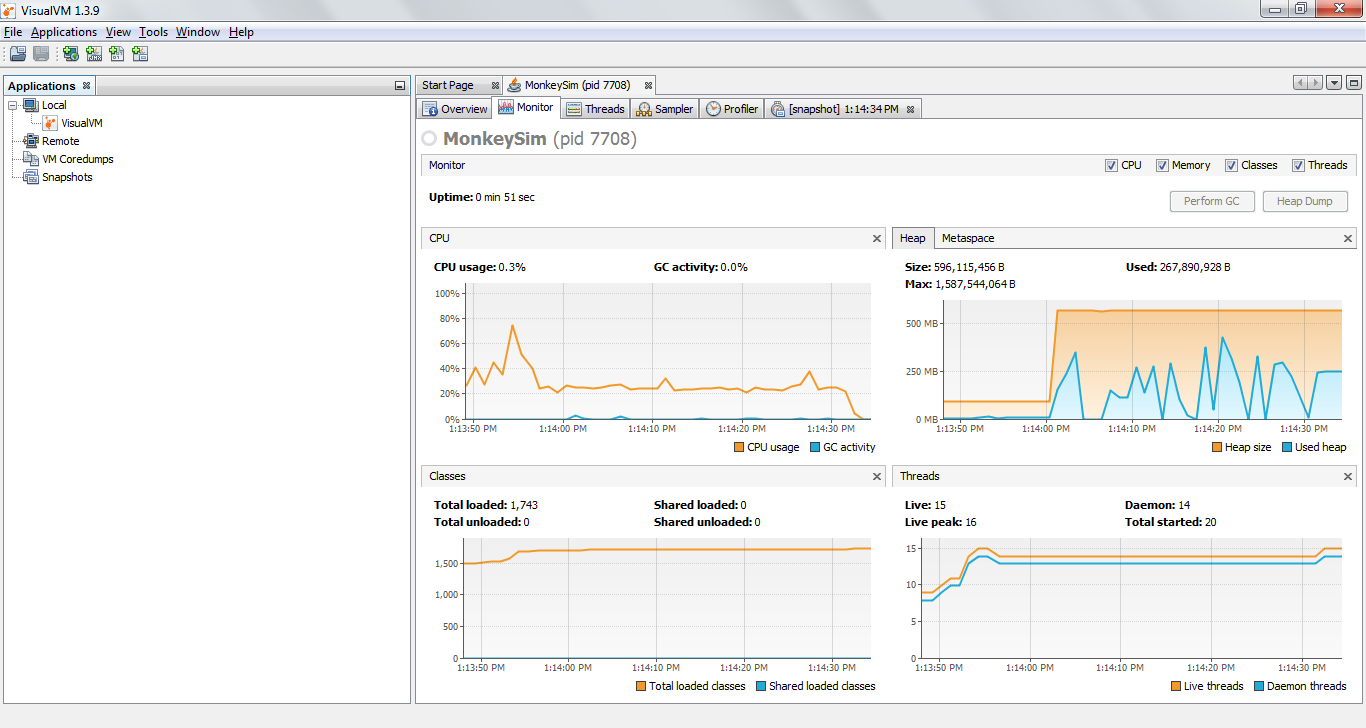
Initial Max Time: 14.7346586 seconds

Final Mean Time: 0.19983703 seconds

Final Max Time: 0.1975829 seconds

**Before Screenshots**





**After Screenshots**

