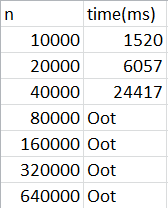
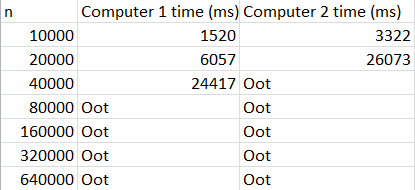


Activity 1. [Make a table reflecting the execution times of the PythonA1.py module for the exposed values of n (10000, 20000, 40000, 80000, 160000, 320000, 640000). If for any n it takes more than 60 seconds you can indicate OoT (“Out of Time”), both in this section and in subsequent sections.]



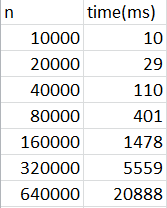
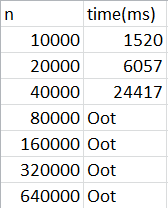
Activity 2. [Make a table that reflects, at least for two computers to which you have access, the execution times of the PythonA1.py module for the exposed values of n (10000,20000, …, 640000). Clearly indicate which CPU and RAM memory you are using in each test.]



Computer 1 : Intel core i5-12600KF, 32GB DDR5

Computer 2: Intel core i5-8300U, 8GB DDR3

Activity 3. [Program a class named JavaA1.java, which uses the same A1 algorithm to find out if a number is a primer number, as in PythonA1.py. Then, a table must be made reflecting the execution times of JavaA1.java for the same values of n (10000, 20000, ..., 640000). Finally, compare these times with those obtained in Python (in a previous section) for that same algorithm A1.]

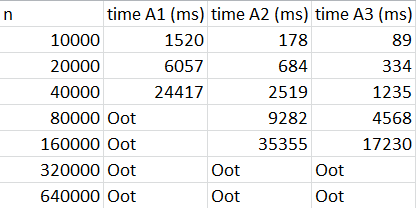
 

Times in Java Times in python

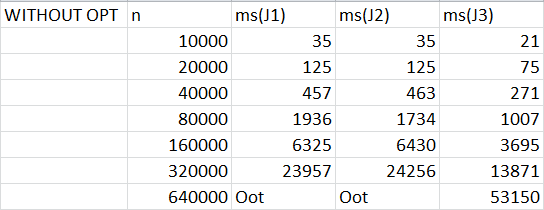
We can see that Java is much faster than python.

Activity 4. [Make a table reflecting the execution times of the modules PythonA1.py, PythonA2.py and PythonA3.py, for the same values of n (10000, 20000, …, 640000). Codify those same algorithms A2 and A3 in Java, in two classes named respectively JavaA2.java and JavaA3.java. Make a table that reflects the execution times of the classes JavaA1.java, JavaA2.java and JavaA3.java, for the values of n (10000, 20000, ..., 640000) WITHOUT OPTIMIZATION of the Java program. Make a table that reflects the execution times of the classes JavaA1.java, JavaA2.java and JavaA3.java, for the values of n (10000, 20000, ..., 640000) WITH OPTIMIZATION of the Java program. Finally, draw final conclusions by comparing the times previously obtained: with Python, with Java WITHOUT OPTIMIZATION and with Java WITH OPTIMIZATION. Optionally, an A4 algorithm can be implemented in Java, which should improve the previous algorithms.]

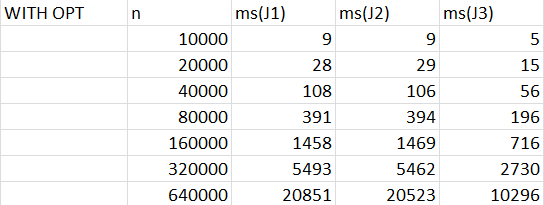
Times for Python A1, A2, A3



Times for Java A1, A2, A3 without optimization:



Times for Java A1, A2, A3 with optimization:



Having studied all the algorithms we can assume that A3 is the best algorithm as it takes the lowest time considering the other two. It works reducing the problem size of the problem to its half, knowing that dividing a number by other one that is higher than its half the resulting quotient will be 0.

From the times taken in Java we can see that both, with optimization and without, times of A1 and A2 are very similar, this is because Java does not take many time taking classes outside the main class.

We can easily appreciate that times with optimization in Java are much faster than without it, with optimization we don´t get any out of time measure. The biggest time we get with optimization in any of the three programs is 20 seconds while in the ones without it are higher than 60 seconds.

In conlusion, Java is much faster than python.