Activity 1. Measuring Execution Times

1. How many more years can we continue using this way9.223.372.036.854.775.807 is the maximum number that a long type of number can represent. We can obtain the number of milliseconds that have passed since Jan 1, 1970, which is the year that the count started by having in account the ms that pass in a year and sum all the years that have passed since 1970. We know that the ms in a year is 31.536.000.000 ms and 51 years have passed since the count has started so the result is 1.608.336.000.000. Now to obtain the time remaining that we must use this system we do the subtraction of the maximum number of a long and the time that have passed since 1970 and we obtain the ms remaining. In years this is 292.471.157,677536
2. What does it mean that the time measured is 0?It means that between the first execution of the method is such a small that the program didn’t even represent it
3. From what size of problem (n) do we start to get reliable times?With n = 6500 I get all the executions a time above 50 milliseconds that is the minimum number of milliseconds that make the problem reliable

Activity 2. Grow of the problem size

1. What happens with time if the size of the problem is multiplied by 5?The time to execute will grow exponentially
2. Are the times obtained those that were expected from linear complexity O(n)?The times obtained are the ones expected from a O(n) (linear complexity) because the measured algorithm has O(n) complexity.
3. Gráfico, Gráfico de líneas

   Descripción generada automáticamenteUse a spreadsheet to draw a graph with Excel. On the X axis we can put the time and on the Y axis the size of the problem.

Activity 3. Taking small execution times

|  |  |  |  |
| --- | --- | --- | --- |
| n | fillIn(t) | sum(t) | maximum(t) |
| 10 | 0 | 1 | 0 |
| 30 | 0 | 0 | 0 |
| 90 | 0 | 0 | 0 |
| 270 | 0 | 1 | 0 |
| 810 | 1 | 1 | 0 |
| 2430 | 0 | 1 | 0 |
| 7290 | 2 | 2 | 1 |
| 21870 | 3 | 6 | 1 |
| 65610 | 0 | 11 | 1 |
| 196830 | 0 | 27 | 1 |
| 590490 | 3 | 68 | 1 |
| 1771470 | 10 | 177 | 6 |
| 5314410 | 45 | 539 | 35 |
| 15943230 | 79 | 1551 | 73 |
| 47829690 | 205 | 4574 | 211 |
| 143489070 | 775 | 14249 | 917 |
| 430467210 | 1908 | 43041 | 1994 |

What are the main components of the computer in which you did the work?I think first the CPU because it’s the one which is doing all the operations like sum(t)  
and then the memory

Do the values meet the expectations?Theoretical results:

|  |  |  |  |
| --- | --- | --- | --- |
| n | fillIn(t) | sum(t) | maximum(t) |
| 10 | # | # | # |
| 30 | 0 | 3 | 0 |
| 90 | 0 | 0 | 0 |
| 270 | 0 | 0 | 0 |
| 810 | 0 | 3 | 0 |
| 2430 | 3 | 3 | 0 |
| 7290 | 0 | 3 | 0 |
| 21870 | 6 | 6 | 3 |
| 65610 | 9 | 18 | 3 |
| 196830 | 0 | 33 | 3 |
| 590490 | 0 | 81 | 3 |
| 1771470 | 9 | 204 | 3 |
| 5314410 | 30 | 531 | 18 |
| 15943230 | 135 | 1317 | 105 |
| 47829690 | 237 | 4653 | 219 |
| 143489070 | 615 | 13722 | 633 |
| 430467210 | 2325 | 42747 | 2751 |

All the results are in milliseconds. The results are similar to the obtained by the program with some little variations but in most cases are close to the result obtained

Activity 4. Operations on matrices

|  |  |  |
| --- | --- | --- |
| n | sumDiagonal(t) | sumDiagonal(t) |
| 0 | 0 | 0 |
| 0 | 2 | 0 |
| 0 | 4 | 0 |
| 70 | 2 | 0 |
| 10 | 24 | 0 |
| 430 | 175 | 2 |
| 290 | 1305 | 7 |
| 1870 | 11891 | 94 |

**What are the main components of the computer in which you did the work?**I think first the CPU because, as in the previous case we are doing operations like the sum so the CPU will be working and also the memory

**Do the values meet the expectations?**Theoretical values:

|  |  |  |
| --- | --- | --- |
| *n* | *sumDiagonal(t)* | *sumDiagonal(t)* |
| 10 | # | # |
| 30 | 0 | 0 |
| 90 | 0 | 0 |
| 270 | 0 | 0 |
| 810 | 24 | 0 |
| 2430 | 72 | 2 |
| 7290 | 216 | 6 |
| 21870 | 648 | 18 |

On this case the values vary a lot from the original, maybe it’s because when I took the values the processor is doing some other instructions, so the times increase.

Activity 5. Benchmarking

It is normal to have different execution time despite being the same program because Java and Python don’t execute the code in the same way. Probably the times on a second

measurement on each IDE would be different too.