Activity 1. Measuring execution times

CurrentTimesMillis ()

Since the currentTimeMillis () method, uses as type long the amount of milliseconds we can use it would be the MAX\_LONG value in java that is: 9,223,372,036,854,775,807. If we convert that number to years it would be approximately: 2924594114 years. So the answer is we do not have to worry about this method functionality at all.

Vector2 class

The time is 0 sometimes because the currentTimeMillis() method is run both times in the less than a millisecond. The number n where we start getting times above 50ms is approximately 13000000(13 million).

Activity 2. Taking small execution times(<50ms)

Vector4 class

When we multiply the problem size by 2 the times above 50ms double its time.

When multiplying the times by 3 and 4 the result is the expected, the times increase accordingly. So taking this into account we got the results expected for a program with complexity O(n).

Table 1: T sum and T maximum WITHOUT OPTIMIZATION

The specs of the pc for this table are: RAM (16, 0 GB (15, 8 GB usable) and CPU (12th Gen Intel(R) Core(TM) i5-12400 2.50 GHz)

|  |  |  |
| --- | --- | --- |
| N | Tsum | Tmaximum |
| 10000 | 42 / 1000 = 0,042 ms | 59 / 1000 = 0,059 ms |
| 20000 | 82 / 1000 = 0,082 ms | 115 / 1000 = 0,115 ms |
| 40000 | 1270 / 1000 = 1,27 ms | 229 / 1000 = 0,229 ms |
| 80000 | 2535 / 1000 = 2,35 ms | 459 / 1000 = 0,459 ms |
| 160000 | 5073 / 1000 = 5,073ms | 917 / 1000 = 0,917 ms |
| 320000 | 10171 / 1000 = 10,171 ms | 1853 / 1000 = 1,853 ms |
| 640000 | 20609 / 1000 = 20,609 ms | 3691 / 1000 = 3,691 ms |
| 1280000 | 41056 / 1000 = 41,056 ms | 7411 / 1000 = 7,411 ms |
| 2560000 | OoT | 14885 / 1000 = 14,885 ms |
| 5120000 | Oot | 29284 / 1000 = 29,284 ms |