# Programming Walkthrough #2

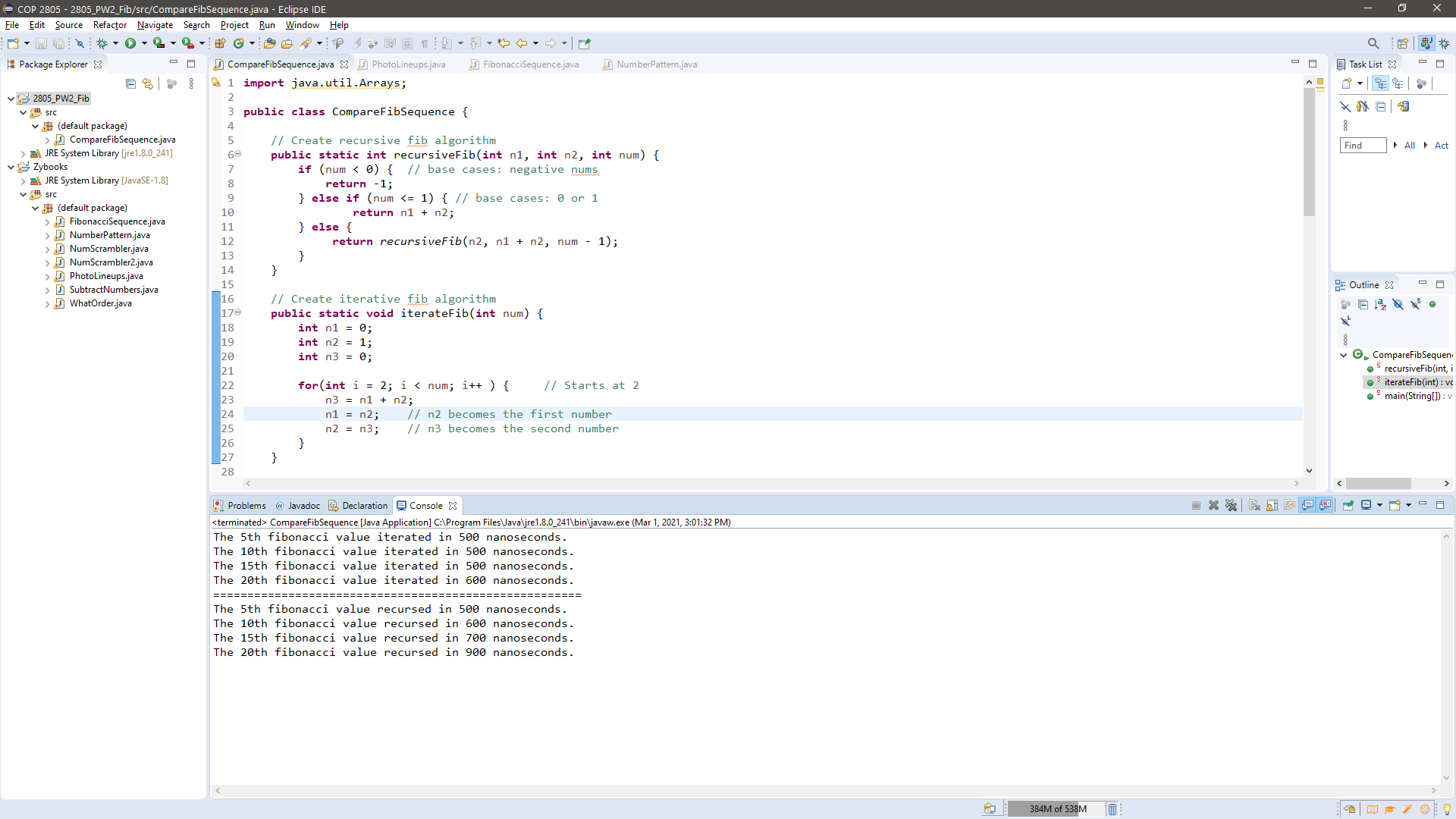
|  |  |  |
| --- | --- | --- |
| **nth Term** | **Recursive Runtime**  **(nanoseconds)** | **Iterative Runtime**  **(nanoseconds)** |
| 5 | 500 | 500 |
| 10 | 600 | 500 |
| 15 | 700 | 500 |
| 20 | 900 | 600 |

From the table, even though the recursive algorithm starts out with a comparable runtime, as the nth term increased so did the time to process the recursion.

When reading about why recursion runtimes lag behind iterative runtimes I learned that with recurision as the stack increases, so do the number of subroutine calls to memory. This doesn’t mean that recursion isn’t always the best choice. Recursion is great for problems that require simple repeated actions like factorials or complicated branching like searching for a file among all the folders on your computer.

From an answer on Stack Overflow[[1]](#footnote-1) I found this summary.

|  |  |
| --- | --- |
| **Recursion** | **Iteration** |
| *Best Use:* Compact “elegant” code, Branching “Fractal Data Structures” | *Best Use:* Simple loops, Sequential Data Structures |



*Program in use*

1. “Is Recursion Ever Faster than Looping?” *Stack Overflow*, 15 Jan. 2015, stackoverflow.com/questions/2651112/is-recursion-ever-faster-than-looping. [↑](#footnote-ref-1)