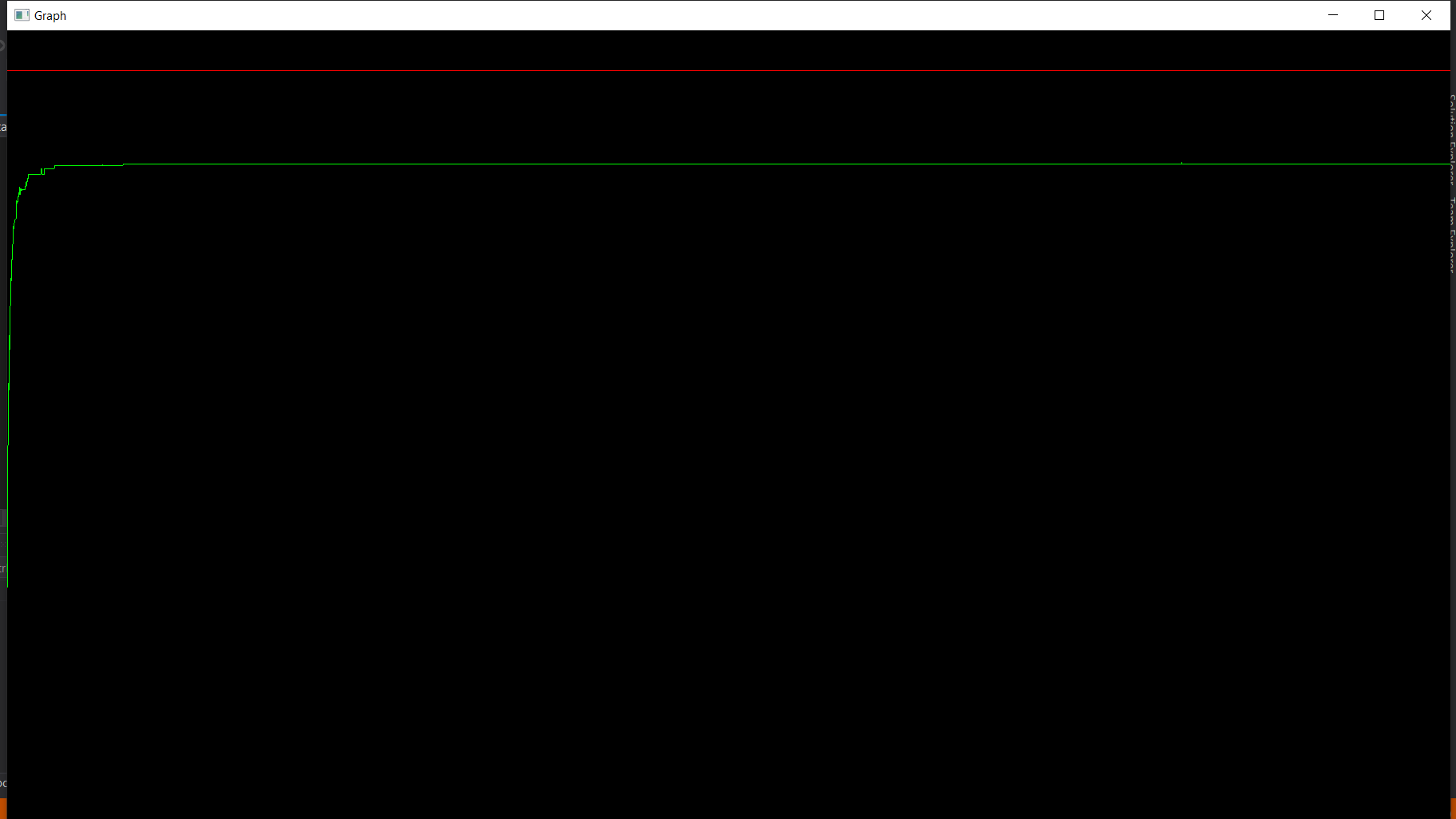
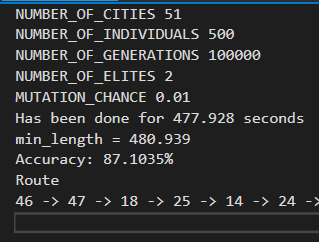
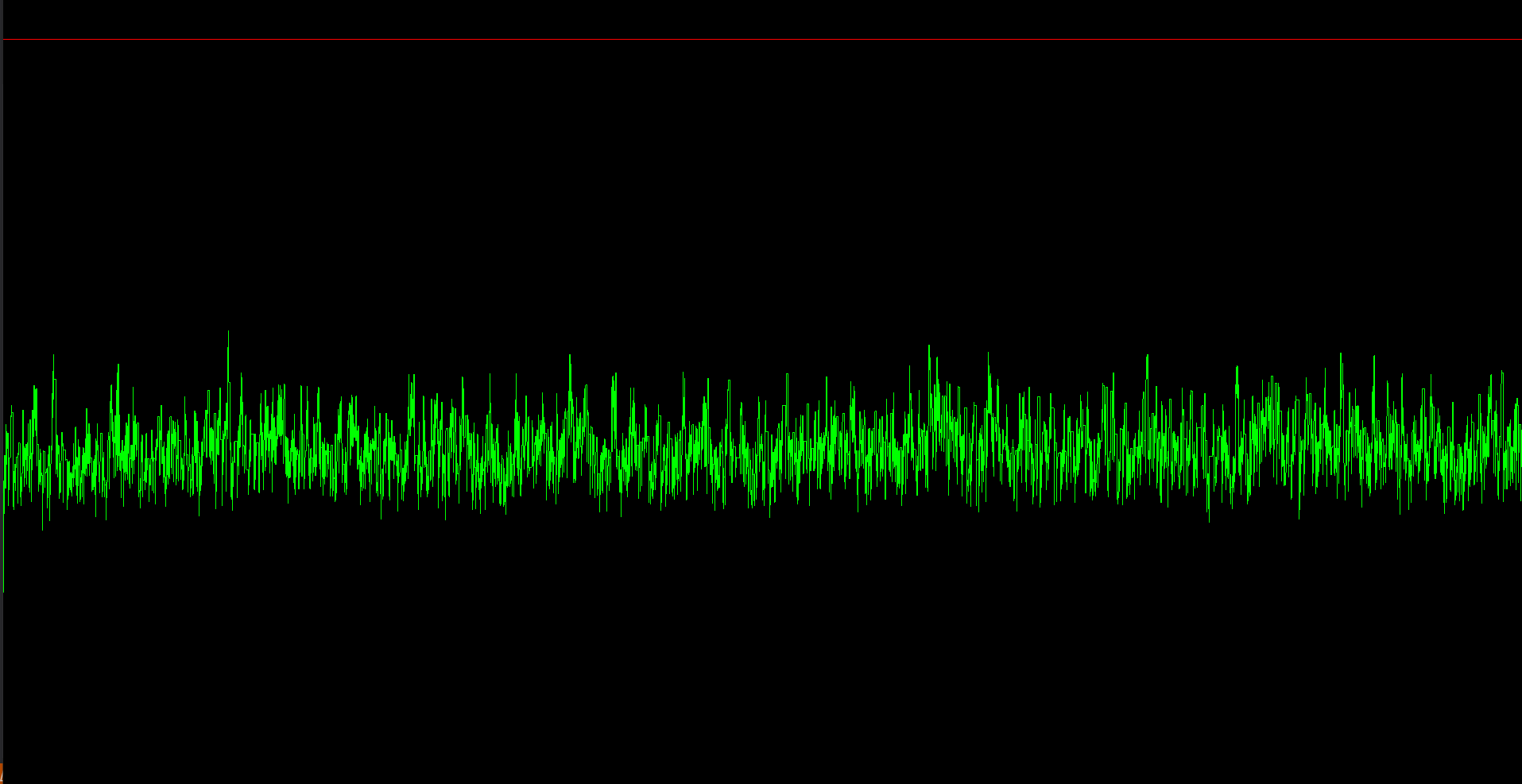
**Euclidiean TSP with 52 vertecies**

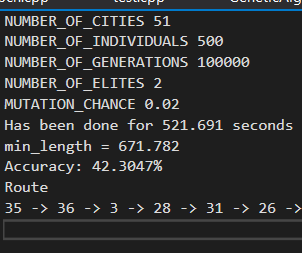
**Genetic algo Tournament selection**

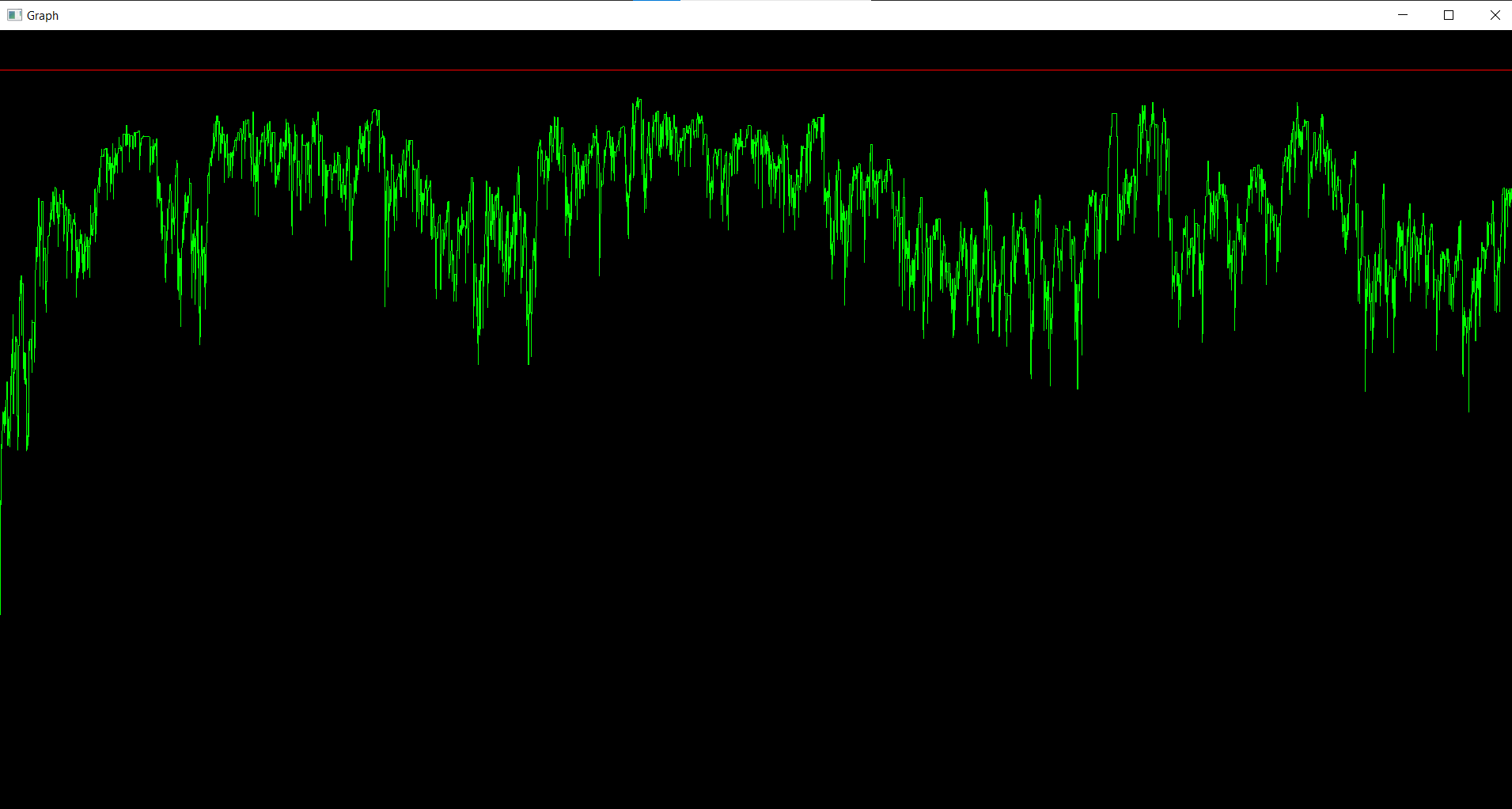
Threads and visualization

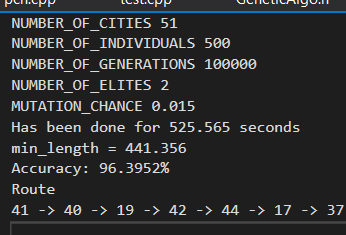


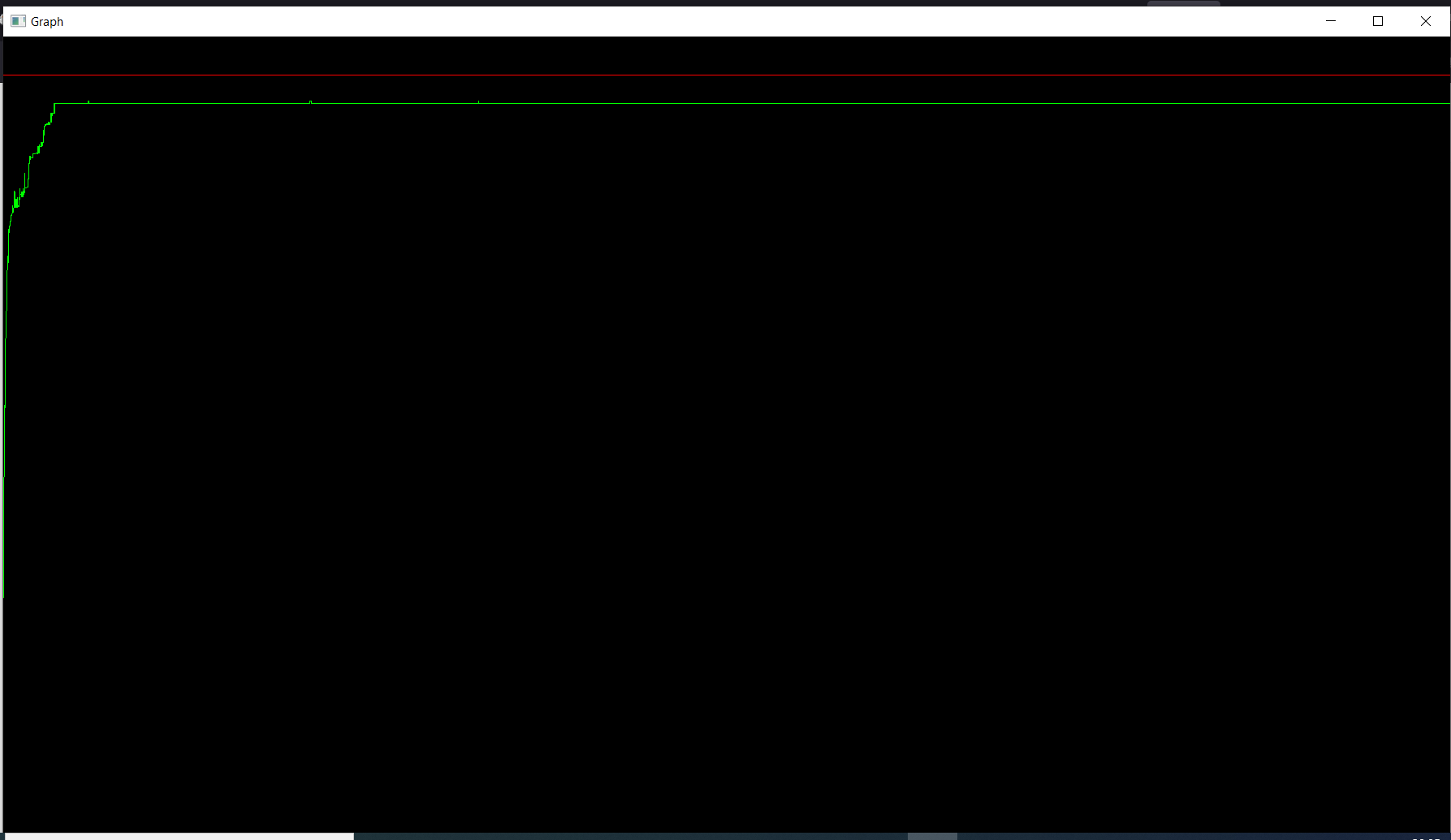


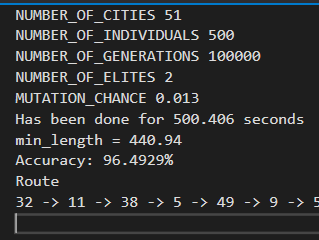


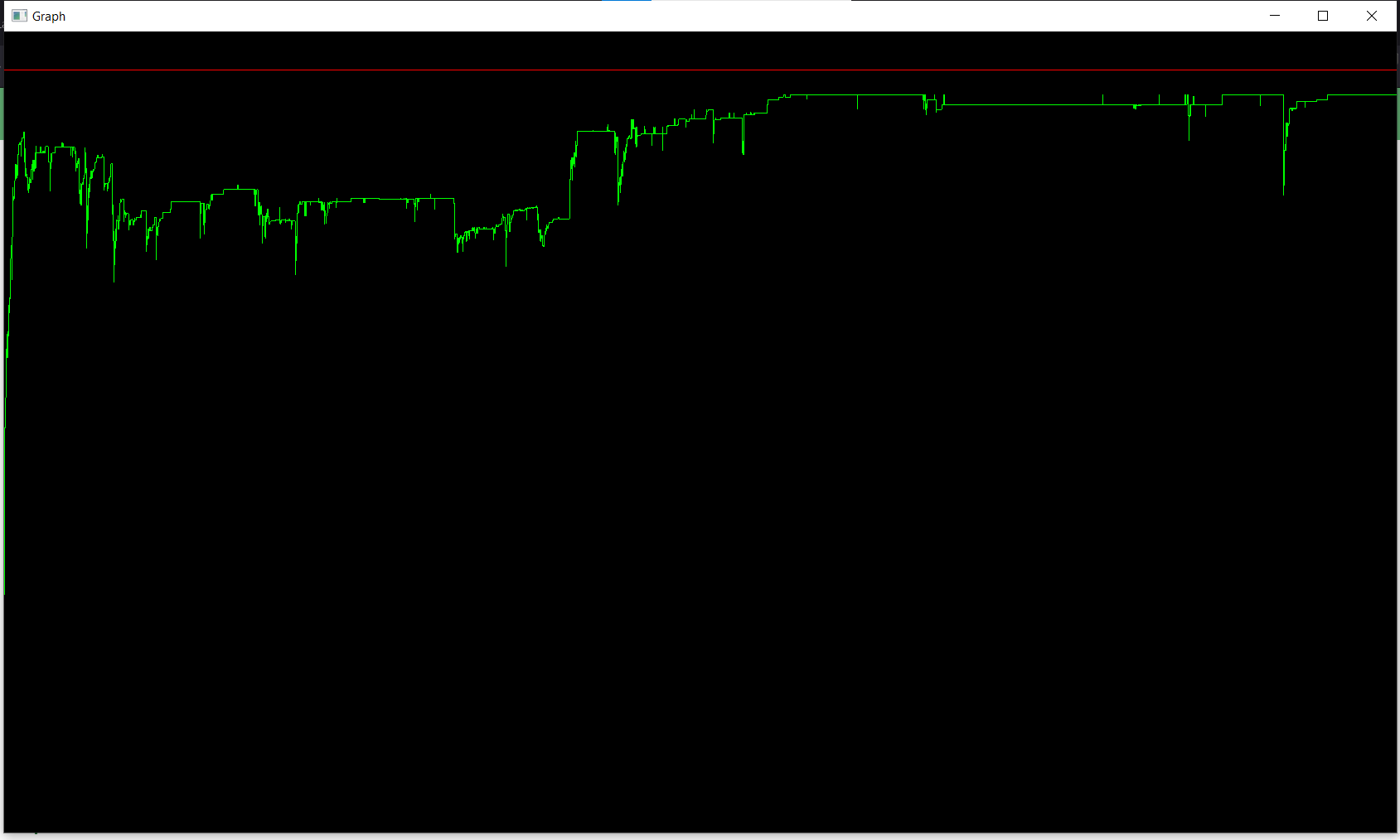


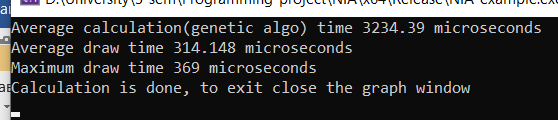


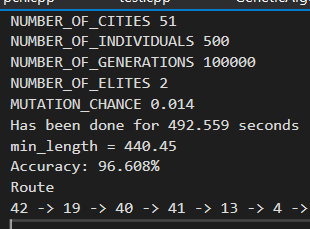






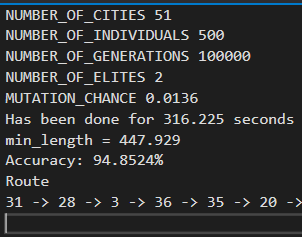




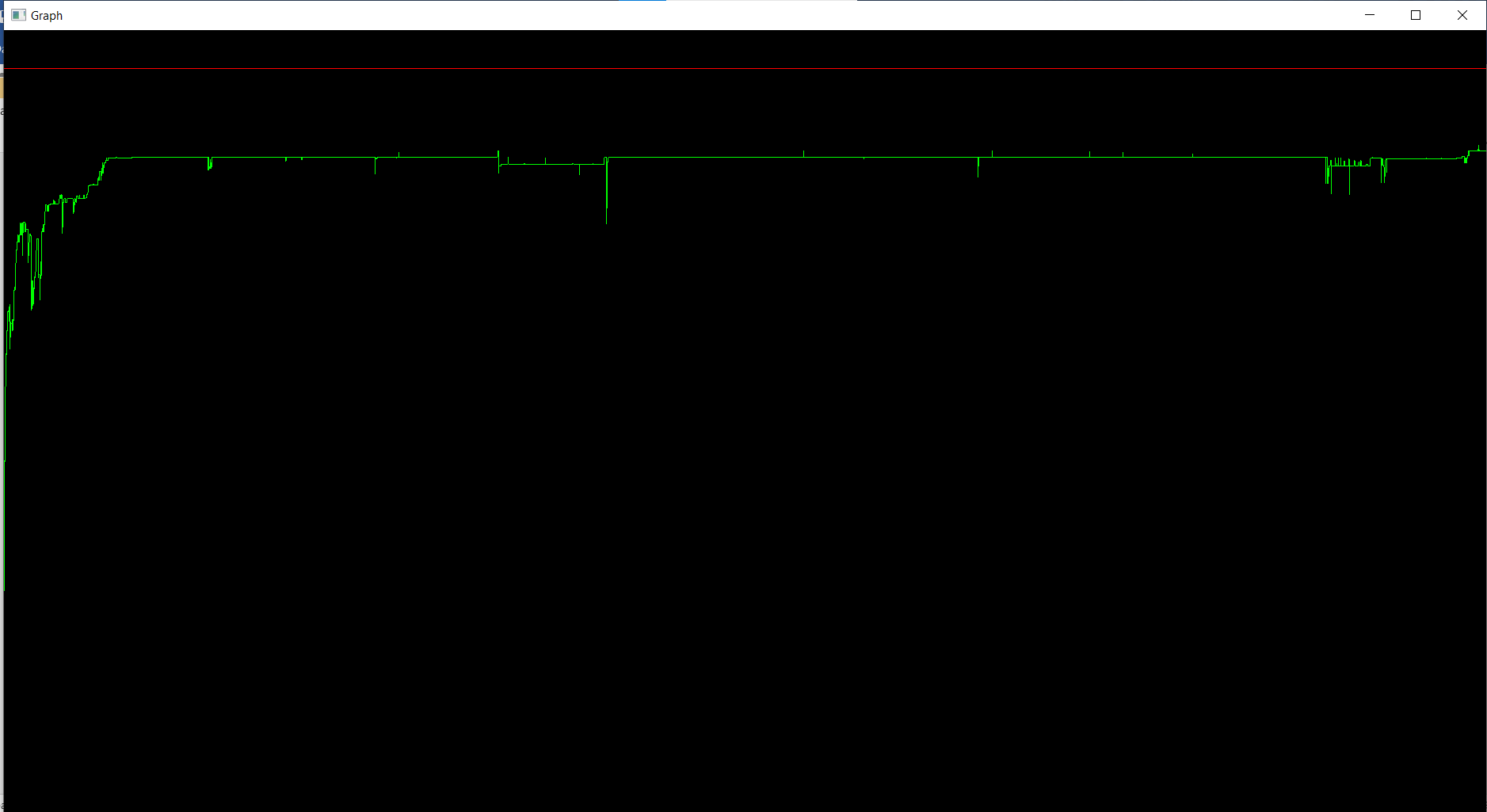


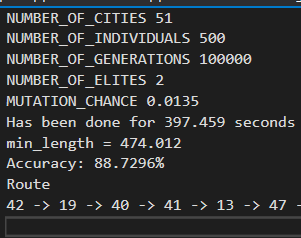
All previous was done with threads. Now let’s compare with implementation without thread.

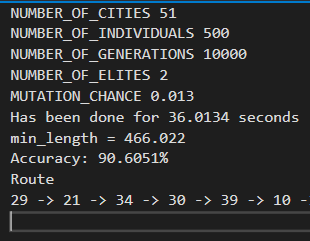
Withot visualization and without thread



With visualization and without thread





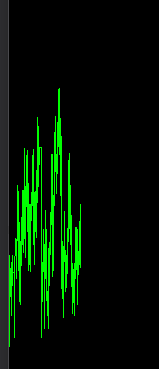


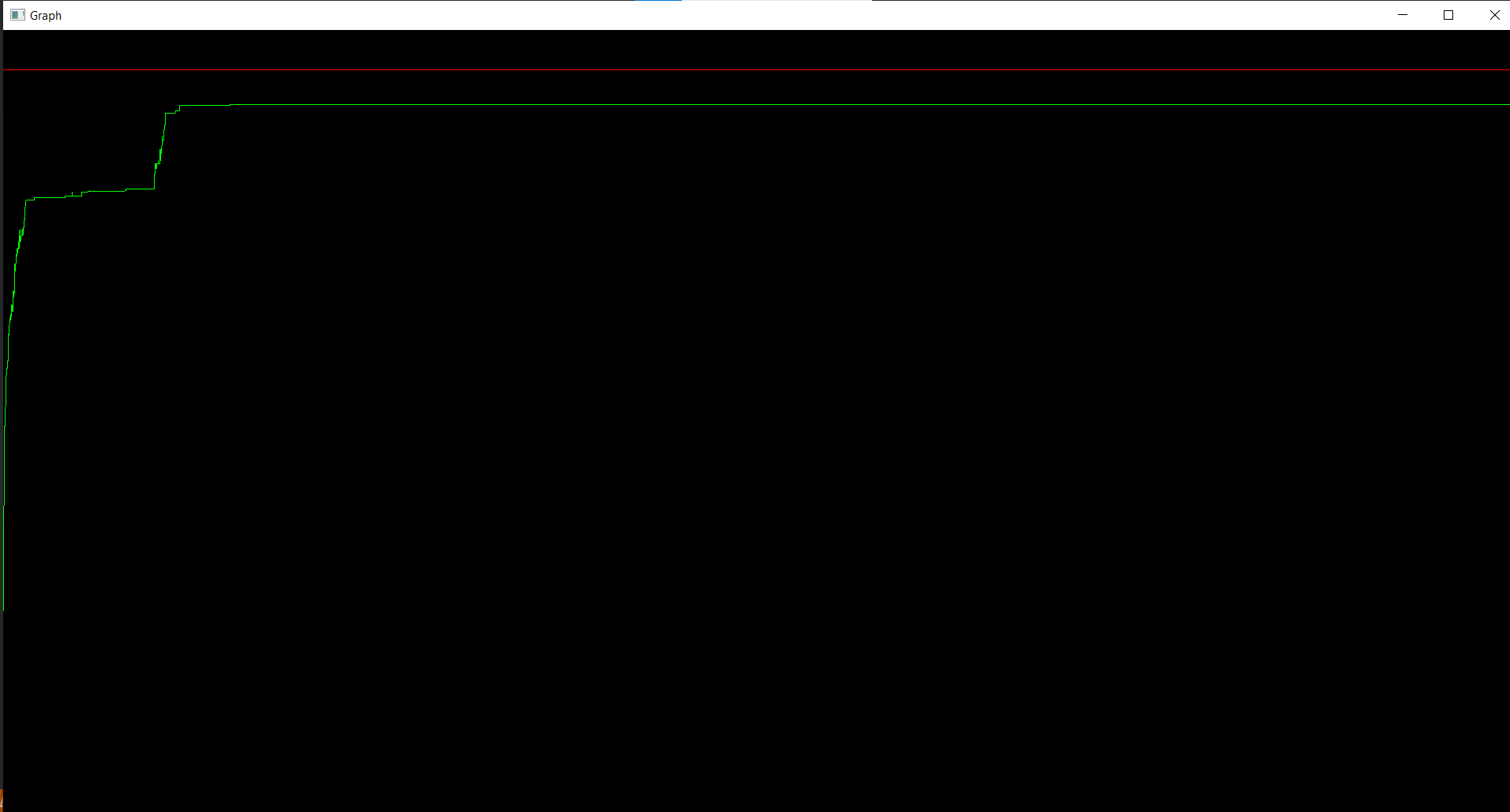
Forgot to scree visualization.

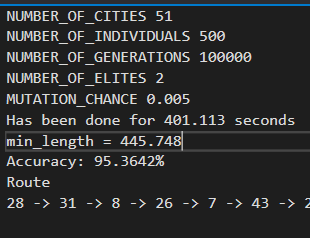
**Genetic algo Roulette Wheel selection**

Without threads, with visualization.

MUTATION\_CHANCE = 0.013

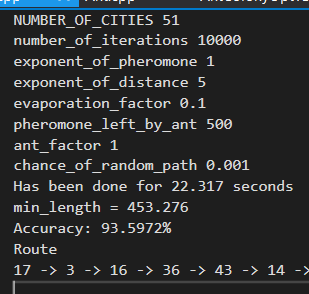




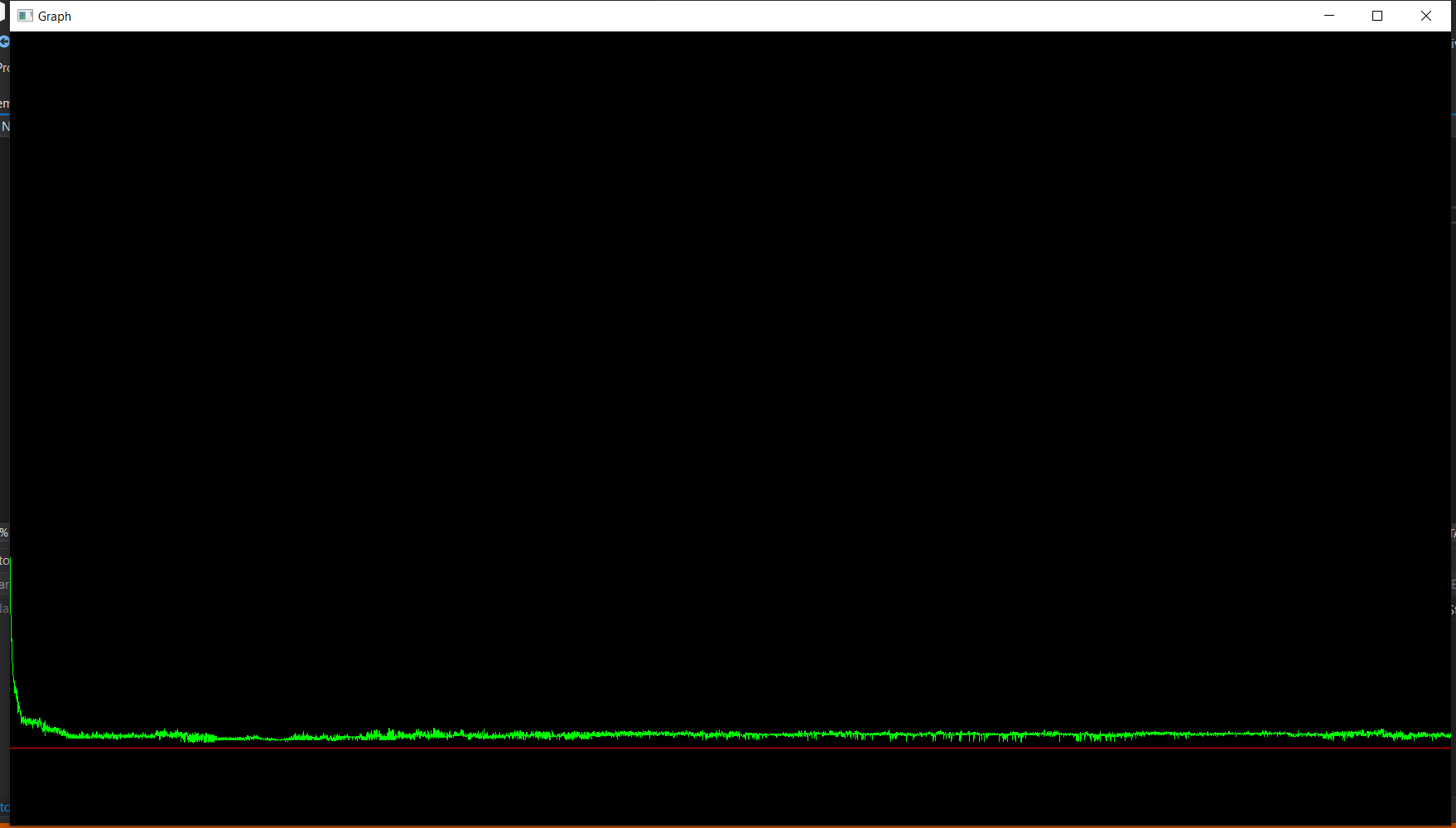


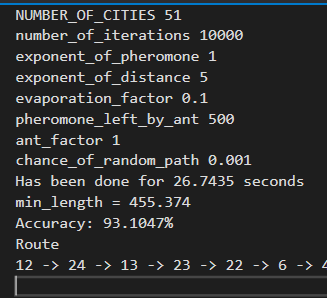
**Ant colony optimization algo**

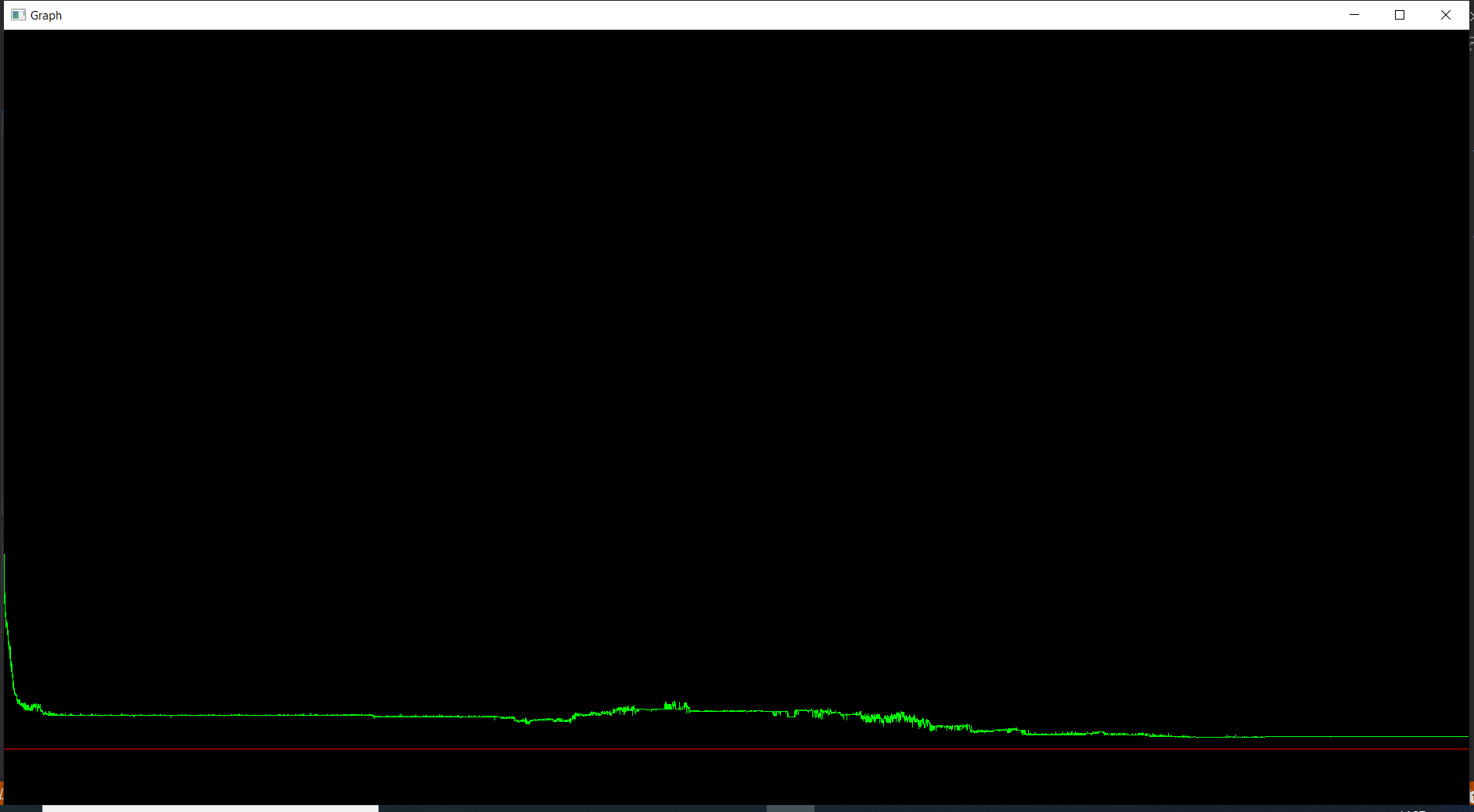
Without threads, without visualization.

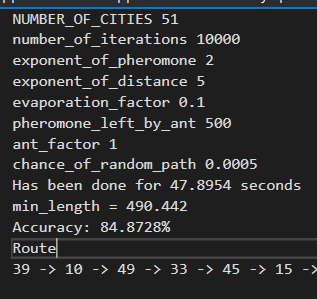


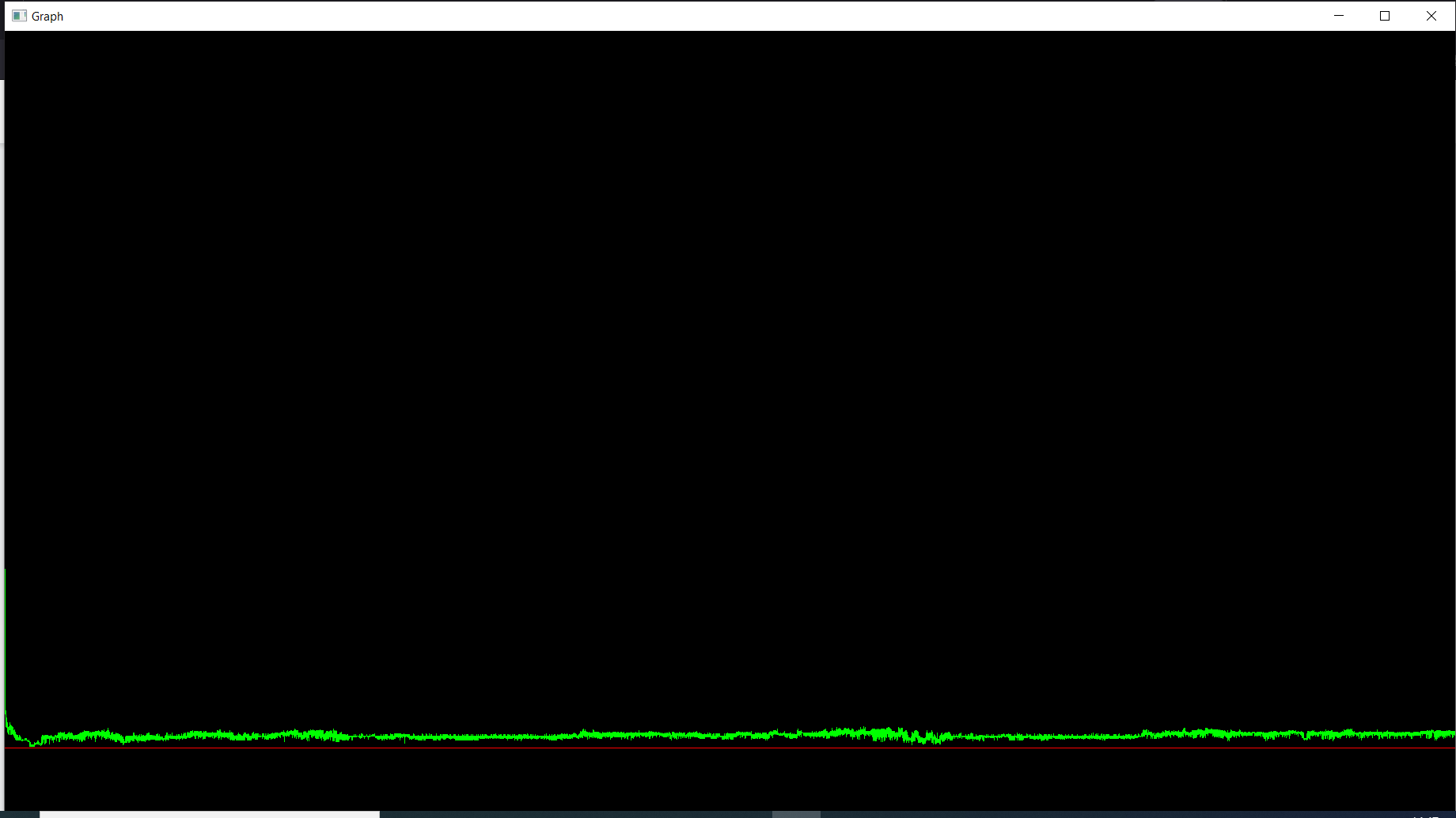
Without threads, with visualization.

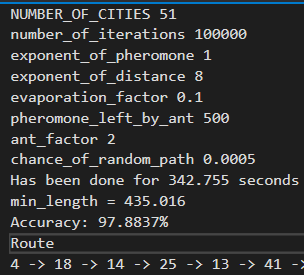


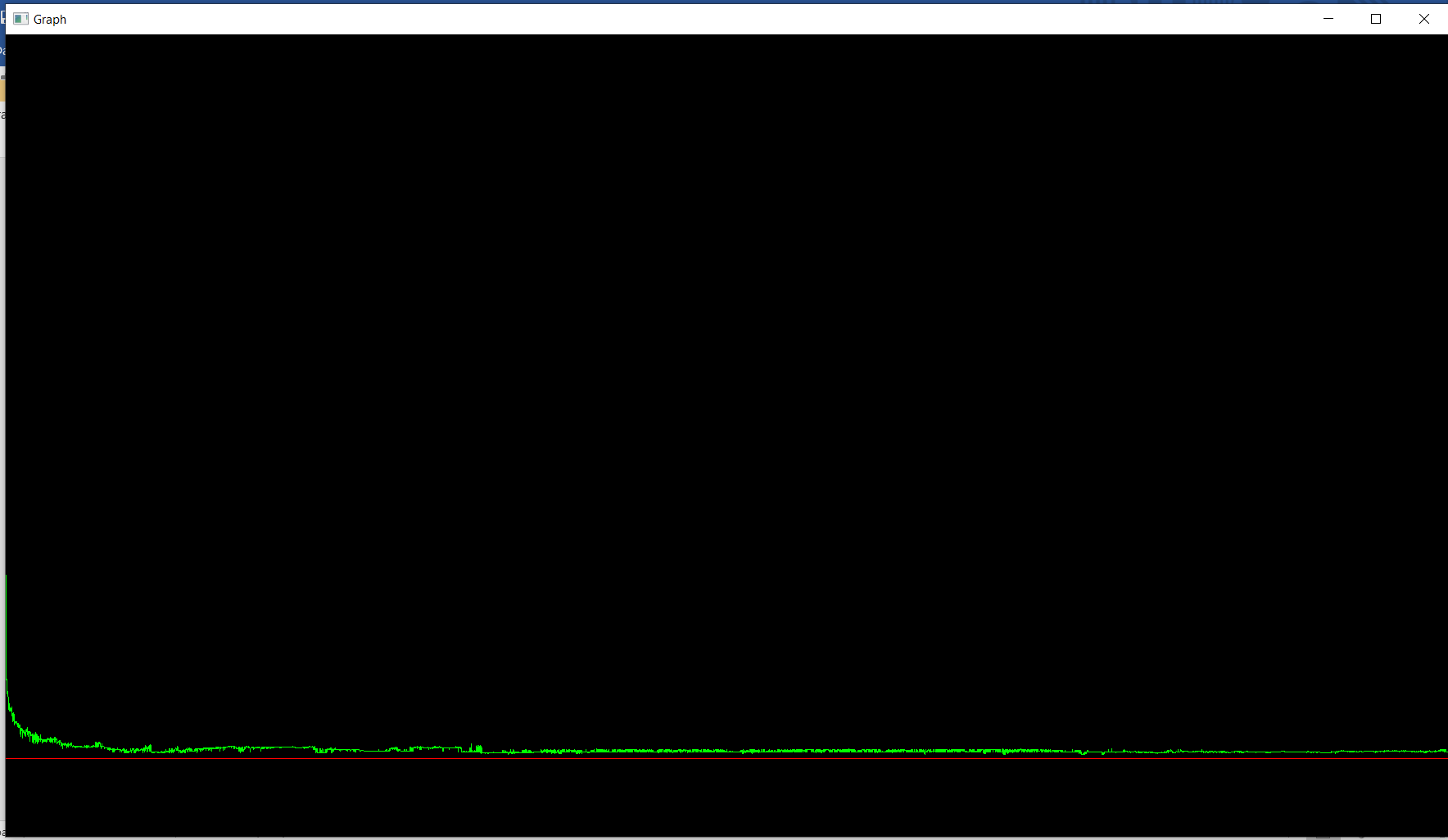


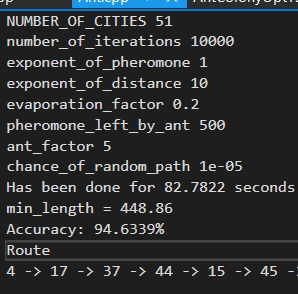












Conclusion: ACO algorithm faster finds solution with more than 90% accuracy, but I don’t know why it’s so often jumps out of minimum (something with constants I think). Genetic algorithm more stable, but to find solution with more that 90% accuracy it needs way more time(Roulette Wheel selection slower and works for lesser mutation chanches that Tournament selection). Of course, behavior of these algorithms highly depends on constants, so somebody more proficient can achieve better results.