Code	**GENETIC_ALGORITHM.PY**
Author	Nathaniel Heatwole, PhD (heatwolen@gmail.com) (GitHub) (LinkedIn)
Summary	Use a genetic algorithm to optimize (maximize) an erratic function, one that has myriad sharp peaks and local extrema
Methods/ process	Genetic algorithm: - Iterative optimization method that mimics the workings of biological evolution. - As the iterations progress, the pool of candidate solutions generally improves. - Can optimize complex parameter spaces, including those that are non-convex, non-differentiable, and non-continuous. - Concepts: - *Fitness*: objective function (“genetic” representation). - *Selection*: randomly select persons from the current population to breed (with replacement), in proportion to their fitness value. - *Crossover*: randomly combine parameter values (“genes”) from parents. - *Mutation*: randomly replace some genes with randomly chosen values from within the parameter space (helps ensure good solutions areas are not overlooked). - *Generations*: number of iterations to continue. - *Output*: global best solution achieved (on any generation).
Objective function	Maximize:[1] $Y(X) = \lvert \sin(A * X) + \sin(B * X) + \sin(C * X) \rvert$ with: $A = (2 * \pi) / 13$ $B = (2 * \pi) / 18$ $C = (2 * \pi) / 23$
Hyper- parameters	- Population: 50 (total simulated people) - Generations: 200 (total iterations) - Mutation rate: 5% - Parameter bounds: X = 0 ... 400
Outputs	Plots: - Objective function (showing maximum value) - Close-up on area around optimal solution
Result	The algorithm efficiently locates the maximum, while avoiding “distractors” (i.e., potential solutions that are close to, but not quite as good as, the true optimal).

[1] This function is similar to that presented at https://mathblag.wordpress.com/2013/09/01/sums-of-periodic-functions.