Try to answer the questions below and motivate your choice for the most suitable value for each parameter:

1) Change the length of the target phrase

2) Change the number of population members

3) Change the mutation rate

4) Change the range of possible characters being considered (include numbers, etc)

5) Try changing between the two breeding methods ‑ which one works better?

6) Change the mating factor ‑ what benefit might we get from increasing this?

What’s a reasonable value for it?

7) Change the maximum generations ‑ what happens to fitness over time?

8) Which function takes the longest to run? Can you improve its runtime at all? To make our code much more efficient at selecting the best parents, we can force our population to favor slightly fitter members much more than everyone else. After calculating our fitness, which are numbers from 0 to 1, we can raise it to some power, let’s say 3. Now values which are close in fitness will be much easier to differentiate between.

Although the fitness values themselves got smaller, the relative difference between them became larger. Remembering that the mating pool normalizes fitnesses to the maximum fitness, this will now award 0.33 even more tickets in the raffle compared to 0.30 or 0.27. In your script, try raising your entire fitness vector to different values before the mating pool is built (we’ll call this the ‘exponential factor’). How high can you make this value before it stops becoming beneficial? How might you want to adjust your mating factor (from Task 1.3) after introducing this exponential factor? Write your observations in the report.