CSCE-496
Assignment 2
Brice Johnson

Part I

1) To determine the best fitness function I ran 10 recorded tests(Table 1 and Table 2 at the end of the document) and analysis the data. With a mutation rate of 1.8%(5 mutations per generation, this was the point when both fitness function would converge) we can see in the data that both fitness functions were able to find the solution within the give generations (1000000 generations is the stopping point). This rules out evaluating the fitness function via completion percentages, so I will base my opinion of best fitness function over number of generations needed to find a solution. Fitness function 2 is easily the better fitness function. Fitness function 2 is the binary fitness, i.e. do the letters match or not.
2) Removing the mutation operator change the solutions drastically. When I removed the mutation when using either of the fitness functions, I was never able to have the algorithm converge on a solution. To note, just basic observations, neither fitness gave a better answer than the other without a mutation. That is, no patterns in the best chromosome emerged using either algorithm.
3) I set up a random experiment in a new function as follows. I randomly generate 17 character and then check the fitness of that word. I repeat this a maximum of 10000 times, then I output a correct random generation or the best fitness randomly generated. You can see the code in the function randomGenerations();

Part II

1) For the first experiment, I changed my crossover points. In my original algorithm I had two random crossover points each generation. For the experiment I wanted to see if having a fixed crossover points would allow the algorithm to convert with the new mutation rate I used in the original experiment. As you can see in Table 3 having a fixed crossover point, when using fitness function 2 (the best fitness function from my experiment) there was no times that the algorithm converged on a solution. The fitness at the end of the algorithm was between 7 and 13 for each of the trials listed below. Having random crossover points is critical to finding a solution.
2) For the second experiment, I change the mutation rate as the algorithm want on. Using Fitness Function 2, I decreased the mutation rate to 1.1% (3 mutations) when the fitness was below 10, then deceased the mutation rate to 0.36% (1 mutation) when the fitness was below 5. As you can see in Table 4, this change did not have a large impact on the convergence of the algorithm. If you could gather one conclusion from this data, it would be that using a decreasing mutation rate throughout the algorithm made the number of generations required to find the solution a little bit more consistent. This is seen because the standard deviation decreased by 341 points from the original algorithm to the decreasing mutation rate algorithm. If you are looking for a more consistent algorithm, using a decreasing mutation rate may get you to the results you are looking for.

Visualization

1) In Figure 1 and Figure 2 you can see visualization for both the Original Algorithm and Experiment 2 (respectively) using Fitness Function 2. Both of these show that this is a good fitness function because there are no major jumps, it looks like an exponentially decaying function which is exactly what a genetic algorithm should look like when visualized. Something to note, when seeing these visualization you can see that Experiment 2 decreases its fitness faster(gets close to 0, best fitness) than the original experiment. Unfortunately Experiment 2 still takes about the same time to settle on a correct solution as the original algorithm.

Table 1 - Original Experiment, Fitness 1

Fitness 1										
With Mutation	22753	12407	36224	16964	16964	9181	16531	33811	59119	12574
Max	36224									
Min	9181									
Mean	23607									
Std Dev.	15366									
Without Mutation	1000000	1000000	1000000	1000000	1000000	1000000	1000000	1000000	1000000	1000000
Max	1000000									
Min	1000000									
Mean	1000000									
Std Dev.	0									

Table 2 - Original Experiment, Fitness 2

Fitness 2										
With Mutation	899	1128	1790	1976	1049	4876	1269	1438	1421	1191
Max	4876									
Min	899									
Mean	1703									
Std Dev.	1162									
Without Mutation	1000000	1000000	1000000	1000000	1000000	1000000	1000000	1000000	1000000	1000000
Max	1000000									
Min	1000000									
Mean	1000000									
Std Dev.	0									

Table 3 - Fixed Crossover Point

Fitness 2 Exp.										
Exp.	1000000	1000000	1000000	1000000	1000000	1000000	1000000	1000000	1000000	1000000
Fitness	13	12	12	13	11	7	10	11	12	13
Max	1000000									
Min	1000000									
Mean	1000000									
Std Dev.	0									

Table 4 - Decrease Mutation Rate

Fitness 2 Exp.										
Exp.	2683	2328	3101	2658	1382	1818	1126	1554	2322	3768
Max	1126									
Min	3768									
Mean	2274									
Std Dev.	821									

Figure 1 - Fitness 2, Original Algorithm

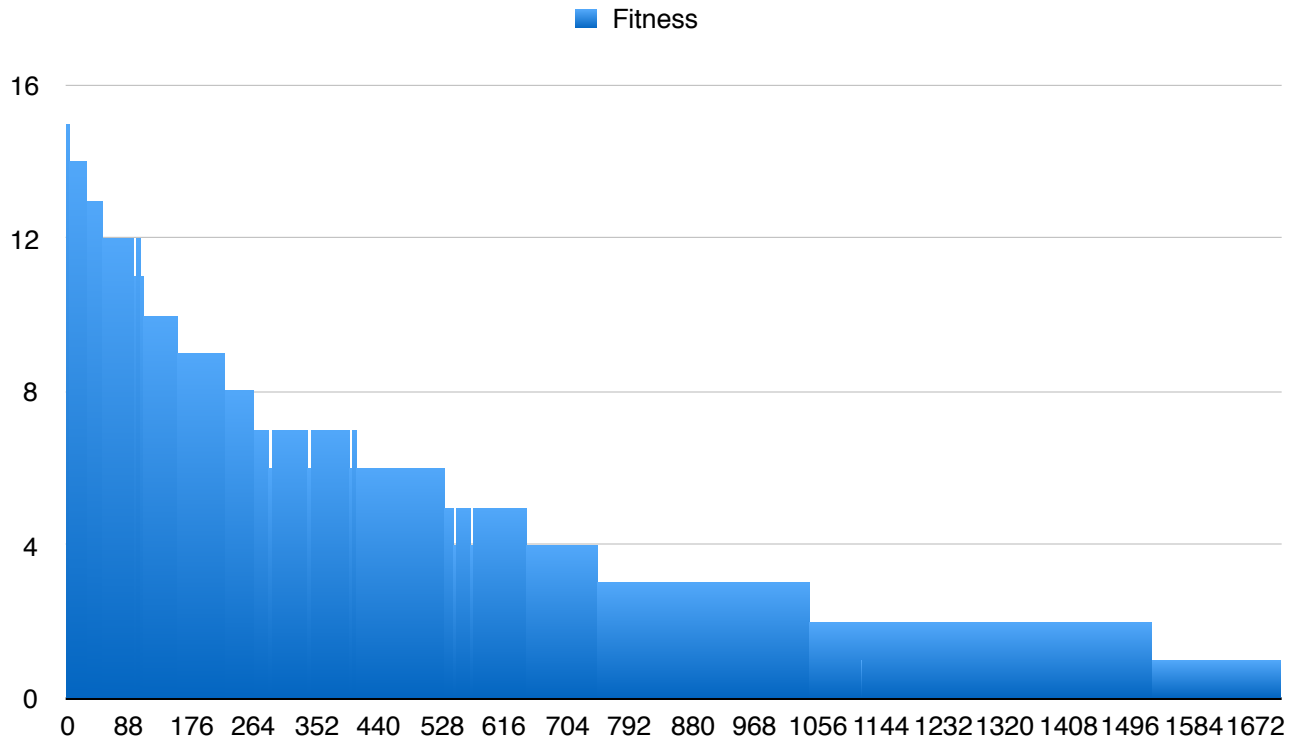

Figure 2 - Fitness 2, Decreasing Mutation Rate (Experiment 2)