Q1. Write a report on the completion of your Genetic Algorithm. Describe in your own words the mechanism of mutation, recombination, and the fitness function and how you applied these in your algorithm.

For Mutation and Recombination is a function of redefining the fittestDNA1 and fittestDNA2.

- fittestDNA1, fittestDNA2 = recombination(fittestDNA1, fittestDNA2)
- fittestDNA1 = mutation(fittestDNA1, mutationChance)
- fittestDNA2 = mutation(fittestDNA2, mutationChance)

Mutation, in biology is defined as alteration of the nucleotide sequence of the genome of an organism, virus or extrachromosomal DNA, resulted from errors during DNA replication as generation passes. In the code, the process of mutation is done through the function mutation(). For the function two variable are needed to be inputed: competingDNA and mutationRatio. CompetingDNA which is the fittestDNA, which are DNA selected through function weightedDNAchoice(). mutationRatio is mutationChance defined at the beginning. The logic of this function is, each gene in the DNA has a one over mutationRatio (for the current code 1/100) of chance of changing. Therefore, made a for loop for choosing the random string.printable if 1/mutationRatio is true, choosing the I (original string) if not. By this process the value given is the mutated DNA.

Recombination, defined as the exchange of genetic material between different organisms. In biology, recombination can be categorized into one-point crossover, two-point crossover, uniform crossover, and Arithmetic crossover (in computer science). However, for my Genetic Algorithm, recombination is defined as a one-point crossover. For the function recombination() two variable is needed, which are competingDNA1 and competingDNA2. These two variables are fittestDNA1 and fittestDNA2. For the out put of this function is DNAout1 and DNAout2 which are the newly defined fittestDNA1 and fittestDNA2. The main logic of this mechanism of this function is choosing a random point between the DNA, and define it as point a. For choosing point a the range is given as (1, dnaLengh-1) is because if not there is a chance of being changed the whole DNA (if 0), and not being changed at all (if danLength). Based on the point selected, cut off each of the DNA and recombined it with each other DNA. By this process value given is the two recombined DNA.

Fitness function, is a function which applies a fitness score to each of the DNA. The higher the fitness score means higher the probability of being chosen. Lower the fitness score means lower the probability of being chosen. For determining the fitness function, it depends on each of the problem. For the Genetic Algorithm I made, the fitness function is actually a function which calculate the penalty value. This is converted as DNAfitnessPair later on. Therefore, the smaller the penalty, the similar the DNA is to the target DNA, which is my name (CHO_KEUN_HEE). The fitness value is used at the function weightedDNAchoice(). Based on the value it is selected as the fittestDNA1 and fittestDNA2.

- fittestDNA1 = weightedDNAchoice(populationWeighted)
- fittestDNA2 = weightedDNAchoice(populationWeighted)

Therefore, using three types of inversed fitness function given it is possible for the code to distinguish which DNA is close to the target.

Q2. How many fitness function were you able to implement? How was their performance? Describe how your fitness functions work and which one showed good or best results. Can you think of reasons, why fitness functions perform this differently?

To begin with, I have only implemented the three given fitness functions.

The first fitness function is sum of absolute value of difference between each component of target DNA and competing DNA. However, the data type of the components of DNAs are string, therefore, it is needed to use the function ord() to convert to decimal.

The second fitness function is sum of power of difference between each component of target DNA and competing DNA. Once again, the data type of the components of DNAs are string, therefore, ord() function is needed.

The third fitness function is a sum of function which gives a value 1 if the components of target DNA and competing DNA is different. For this function ord() is not needed because it is just comparing if the components are same of not.

Among the three fitness functions, the second one tends to work the best. Between the first and the second fitness functions, they did not show a great difference. However, for the third fitness function, it seems to not able to navigate to the target DNA as well as the other two, according to various time of running each code.

Based on the document provided by the professor, it is mentioned that a fitness function should be least complex to compute and not too quickly converge to a local optimum, and need to search the right rate of discrepancy between the DNA of a current population and the target.

In the case of the third fitness function, the penalty range is from 0 to dnaLength, which is 12 for my code. Then based on the code, DNAfitnessPair for each DNA in the currentPopulation would not have much of a difference. Therefore, fittestDNA1 and fittestDNA2 chosen by the weightedDNAchoice function might not be as close to the target as the one chosen in case of first and second fitness function. Which proves why the third fitness functions was not able to navigate well to the target DNA.

On the same logic, if there is a fitness function which the penalty range is too big, the chance of a single DNA being selected as fittestDNA1 and fittestDNA2 might increase. Making recombination useless, and only rely on mutation.