Assignment #HW1 – Genetic Algorithm

CSSE490: Bio-Inspired Artificial Intelligence

Prepared and submitted by:

_____Lyra Lee_____

Collaboration and resources:

I worked (alone)

Resources I used to complete this assignment (websites, textbook, friends, etc.):

Java official documentation

For the check points, simply include a screenshot of the output of your program demonstrating that the code is doing what it is supposed to.

Check Point 1. Print random chromosomes

```
jdwp=transport=dt_socket,server=n,suspend=y,a
lication Support/Code/User/workspaceStorage/d
m-rhit-leey2_10900efb/bin" GeneticAlgorithm
Created Oth Generation of Chromosomes
[1, 0, 1, 0, 0, 1, 0, 1, 1, 0]
[1, 0, 0, 0, 0, 1, 0, 1, 1, 1]
[1, 0, 0, 0, 1, 1, 0, 1, 0, 0]
[1, 0, 0, 0, 1, 0, 0, 0, 1, 0]
[1, 1, 1, 0, 1, 0, 1, 1, 0, 0]
[0, 0, 1, 0, 0, 1, 1, 0, 1, 1]
[0, 1, 0, 0, 1, 1, 1, 0, 1, 0]
[0, 0, 0, 0, 0, 1, 0, 1, 0, 1]
[0, 1, 1, 0, 0, 0, 1, 1, 0, 1]
[0, 1, 0, 0, 0, 1, 1, 0, 0, 1]
lyralee@Lyras-MacBook-Pro fa21_hw1-genetic-al
gorithm-rhit-leey2 ; /usr/bin/env /Library/Ja
jdwp=transport=dt_socket,server=n,suspend=y,a
lication Support/Code/User/workspaceStorage/d
m-rhit-leey2_10900efb/bin" GeneticAlgorithm
Created Oth Generation of Chromosomes
[1, 0, 1, 0, 0, 1, 0, 1, 1, 0]
[1, 0, 0, 0, 0, 1, 0, 1, 1, 1]
[1, 0, 0, 0, 1, 1, 0, 1, 0, 0]
[1, 0, 0, 0, 1, 0, 0, 0, 1, 0]
[1, 1, 1, 0, 1, 0, 1, 1, 0, 0]
[0, 0, 1, 0, 0, 1, 1, 0, 1, 1]
[0, 1, 0, 0, 1, 1, 1, 0, 1, 0]
[0, 0, 0, 0, 0, 1, 0, 1, 0, 1]
[0, 1, 1, 0, 0, 0, 1, 1, 0, 1]
[0, 1, 0, 0, 0, 1, 1, 0, 0, 1]
lyralee@Lyras-MacBook-Pro fa21_hw1-genetic-al
```

Check Point 2. Print individual and verify fitness

```
71      for (int i = 0; i < generationZero.population; i++) {
72        generationZero.allChromosomes.get(i).evaluateFitness();
73      }
74
75      sortChromosomes(0);
76      getBestFitness(0);
77      printChromosomes(0);
78      System.out.println("Generation 0 Fitness Scores: ");
79
80      for (int i = 0; i < generationZero.population; i++) {
81        System.out.print(generationZero.allChromosomes.get(i).fitnessScore + ", ");
82      }
83      System.out.println();
84      System.out.println("Chromosome: ");
85      for (int j = 0; j < chromosomeLength; j++) {
86        System.out.print(generationZero.allChromosomes.get(j).genes.get(j) + ", ");
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
[Running] cd "/Users/lyralee/Documents/Github/fa21_hw1-genetic-algorithm-rhit-leey2/sourc
java GeneticAlgorithm
Created Oth Generation of Chromosomes
[1, 0, 0, 0, 1, 0, 0, 0, 1, 0]
[0, 0, 0, 0, 0, 1, 0, 1, 0, 1]
[1, 0, 0, 0, 1, 1, 0, 1, 0, 0]
[0, 1, 0, 0, 0, 1, 1, 0, 0, 1]
[1, 0, 1, 0, 0, 1, 0, 1, 1, 0]
[1, 0, 0, 0, 0, 1, 0, 1, 1, 1]
[0, 0, 1, 0, 0, 1, 1, 0, 1, 1]
[0, 1, 0, 0, 1, 1, 1, 0, 1, 0]
[0, 1, 1, 0, 0, 0, 1, 1, 0, 1]
[1, 1, 1, 0, 1, 0, 1, 1, 0, 0]
Generation 0 Fitness Scores:
3, 3, 4, 4, 5, 5, 5, 5, 5, 6,
Chromosome:
1, 0, 0, 0, 0, 1, 1, 0, 0, 0,  and its fitness: 3
```

Check Point 3. List of sorted chromosomes

```
lyralee@Lyras-MacBook-Pro fa21_hw1-genetic-al
gorithm-rhit-leey2 ; /usr/bin/env /Library/Ja
jdwp=transport=dt_socket,server=n,suspend=y,a
lication Support/Code/User/workspaceStorage/d
m-rhit-leey2_10900efb/bin" GeneticAlgorithm
Created 0th Generation of Chromosomes
[1, 0, 0, 0, 1, 0, 0, 0, 1, 0]
[0, 0, 0, 0, 0, 1, 0, 1, 0, 1]
[1, 0, 0, 0, 1, 1, 0, 1, 0, 0]
[0, 1, 0, 0, 0, 1, 1, 0, 0, 1]
[1, 0, 1, 0, 0, 1, 0, 1, 1, 0]
[1, 0, 0, 0, 0, 1, 0, 1, 1, 1]
[0, 0, 1, 0, 0, 1, 1, 0, 1, 1]
[0, 1, 0, 0, 1, 1, 1, 0, 1, 0]
[0, 1, 1, 0, 0, 0, 1, 1, 0, 1]
[1, 1, 1, 0, 1, 0, 1, 1, 0, 0]
Generation 0 Fitness Scores:
3, 3, 4, 4, 5, 5, 5, 5, 5, 6,
lyralee@Lyras-MacBook-Pro fa21_hw1-genetic-al
```

Check Point 4. Mutation

```
131   public void mutateChromosomes(Integer generationNum, Double mutationRate) {
132     System.out.println("Original: " + allGenerations.get(generationNum).allChromosomes.get(0).genes);
133     Chromosome ch = new Chromosome(allGenerations.get(generationNum).allChromosomes.get(0).genes);
134     ArrayList<Integer> mutatedList = ch.mutate(mutationRate);
135     System.out.println("Mutated:  " + mutatedList);
136   }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
Original: [1, 0, 0, 0, 1, 0, 0, 0, 1, 0]
Mutated:  [0, 0, 0, 0, 1, 0, 0, 0, 1, 0]
```

Check Point 5. Crossover

```
155   while (crossOverPoints > 0) {
156     crossPoint = r.nextInt(allGenerations.get(generationNum).allChromosomes.get(0).genes.size() - 2) + 1;
157
158     if (crossOverPoints == 1) {
159       System.out.println("[Trial " + (originalCrossNum - crossOverPoints + 1) + "] Cross point is " + crossPoint
160       for (int i = 0; i < crossPoint; i++) {
161         offspring.add(parent1.get(i));
162       }
163
164       for (int i = crossPoint; i < parent1.size(); i++) {
165         offspring.add(parent2.get(i));
166       }
167       crossOverPoints--;
168
169     } else {
170       System.out.println("[Trial " + (originalCrossNum - crossOverPoints + 1) + "] Cross point is " + crossPoint
171       for (int i = 0; i < crossPoint; i++) {
172         newChromosome1.add(parent1.get(i));
173         newChromosome2.add(parent2.get(i));
174       }
175
176       for (int i = crossPoint; i < parent1.size(); i++) {
177         newChromosome1.add(parent2.get(i));
178         newChromosome2.add(parent1.get(i));
```

PROBLEMS 2 OUTPUT TERMINAL DEBUG CONSOLE Code

```
p1: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
p2: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
[Trial 1] Cross point is 5
crossover p1: [0, 0, 0, 0, 0, 1, 1, 1, 1, 1]
crossover p2: [1, 1, 1, 1, 1, 0, 0, 0, 0, 0]
[Trial 2] Cross point is 1
offspring is: [0, 1, 1, 1, 1, 0, 0, 0, 0, 0]
```

Check Point 6. Check best fitness in each generation

```
[0, 0, 0, 1, 1, 1, 1, 1, 1, 0]
[0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1]
[1, 0, 1, 1, 1, 1, 1, 0, 1, 1]
[1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1]
[1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1]
[1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1]
[1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1]
[1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Used Truncation method
Best Fitness in each generation: [6, 6, 7, 7, 8, 9, 9, 9, 9, 9, 9, 10]
Evolution terminated by reaching best score
```

```
[1, 1, 1, 0, 0, 0, 1, 1, 1, 1]
[1, 1, 1, 1, 0, 0, 1, 0, 1, 1]
[1, 0, 1, 1, 1, 0, 1, 1, 1, 1]
[1, 1, 1, 1, 1, 0, 1, 0, 1, 1]
[0, 1, 1, 1, 1, 1, 0, 1, 1, 1]
[0, 1, 1, 1, 1, 1, 0, 1, 1, 1]
[1, 0, 1, 1, 1, 1, 1, 1, 0, 1]
[0, 1, 1, 1, 1, 1, 1, 1, 1, 1]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Used Roulette Wheel method
Best Fitness in each generation: [6, 6, 7, 7, 7, 8, 8, 9, 9, 9, 9, 9, 9, 9, 10]
Evolution terminated by reaching best score
```

Check Point 7. Elitism value = 1, Check best fitness

```
Number of Elite: 1
     0 crossovers complete
     Roulette Wheel successful with 1 elite children
[1, 1, 1, 1, 1, 0, 0, 1, 0, 1]
[1, 1, 0, 1, 1, 1, 0, 1, 0, 1]
[1, 1, 0, 1, 1, 1, 1, 0, 0, 1]
[1, 0, 1, 0, 1, 0, 1, 1, 1, 1]
[1, 1, 0, 0, 1, 0, 1, 1, 1, 1]
[1, 1, 0, 0, 1, 1, 1, 1, 1, 1]
[1, 1, 0, 1, 1, 1, 1, 1, 1, 1]
[1, 1, 1, 1, 1, 0, 1, 1, 1, 1]
[1, 1, 0, 1, 1, 1, 1, 1, 1, 1]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Used Roulette Wheel method
Best Fitness in each generation: [6, 7, 8, 8, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9,
 9, 9, 9, 9, 10]
Evolution terminated by reaching best score
```

Check Point 8. Including Crossover

```
 9    public Integer chromosomeLength;
10    public Integer population;
11    public Double mutationRate;
12    public String selectionType = "Roulette Wheel"; // change this to Roulette W
13    public Integer elitismPercent = 1;
14    public boolean crossover = true;
15    public Integer terminationScore;
16    public Integer maxGeneration;
17
18    ArrayList<Generation> allGenerations = new ArrayList<Generation>();
19    ArrayList<Integer> bestFitness = new ArrayList<Integer>();
20
21    public GeneticAlgorithm() {
22      this.chromosomeLength = 100; // should be 100
23      this.population = 100;
24      this.mutationRate = 1.0 / chromosomeLength;
25      this.terminationScore = 100;
26      this.maxGeneration = 200;
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1,
 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0]
[1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1,
 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0]

Used Roulette Wheel method
Best Fitness in each generation: [49, 60, 54, 61, 56, 65, 64, 63, 64, 67, 65, 63, 69, 6
9, 70, 69, 69, 70, 73, 67, 70, 71, 71, 72, 72, 68, 71, 73, 74, 73, 74, 75, 76, 75, 73,
75, 77, 77, 80, 74, 75, 80, 80, 78, 80, 80, 78, 81, 77, 77, 80, 78, 80, 83, 81, 81, 84,
 77, 80, 85, 83, 82, 85, 81, 83, 82, 86, 83, 85, 83, 89, 86, 88, 86, 86, 85, 89, 89, 85
, 87, 89, 84, 88, 87, 85, 89, 86, 85, 89, 86, 89, 83, 86, 86, 86, 93, 88, 88, 84, 87, 8
6, 87, 86, 87, 88, 87, 89, 85, 86, 87, 85, 87, 86, 82, 86, 87, 88, 89, 89, 88, 89, 88,
88, 86, 85, 85, 84, 89, 90, 90, 92, 86, 88, 91, 88, 86, 88, 91, 88, 89, 89, 91, 90, 88,
 91, 90, 91, 92, 88, 88, 86, 90, 91, 90, 91, 89, 92, 91, 88, 88, 92, 88, 92, 91, 88, 91
, 89, 91, 89, 89, 91, 86, 90, 88, 89, 89, 90, 85, 90, 87, 90, 89, 90, 90, 90, 89, 91, 9
0, 90, 91, 88, 92, 93, 91, 90, 90, 86, 93, 91, 93]
```

Experiment #1 – Only Mutation

a. An explanation of the experiment (including parameters used)

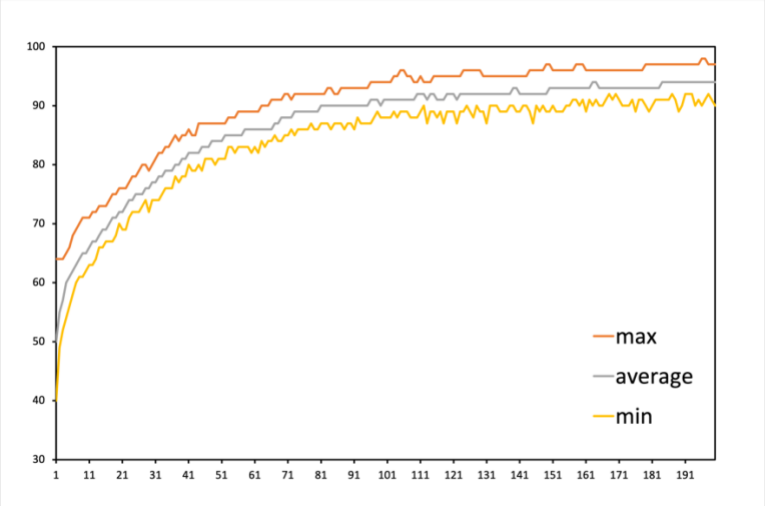
population size:	100
mutation rate:	1/100
crossover method:	None
Number of elite:	0
chromosome length:	100
selection method:	Truncation
max generations:	200

Ran the simulator with different seeds of the randomizer and had a termination score of 99.

b. A hypothesis (might just be a guess)

I expect about 180 generations will take to evolve the best solution because there is no crossover or elitism. The rates would be a shape of three curves that never cross each other.

c. Description of the results including plots (or a table of results)



Data Collected:

run #	1	2	3	4	5	6	7	8	9	10
gens	196	187	132	189	192	143	172	195	178	195

The average number of generations is 177.9 which a close number to the hypothesis. The three curves do not cross over each other and has a similar curve of a shape.

d. What if anything you can conclude and anything you learned or affirmed by doing so?

The fitness scores increase and decrease as generation changes but increase in the big picture.

Experiment #2 – Elitism of rate of 1/100

a. An explanation of the experiment (including parameters used)

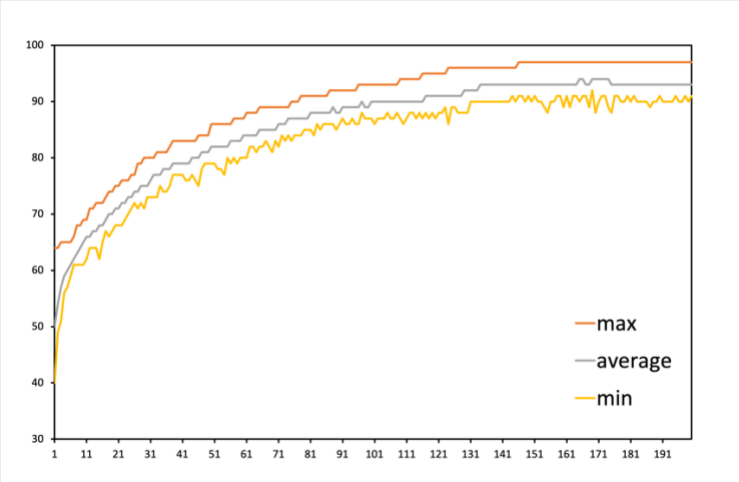
population size:	100
mutation rate:	1/100
crossover method:	None
Number of elite:	1
chromosome length:	100
selection method:	Truncation
max generations:	200

Same parameters as experiment 1 but added elitism number of 1.

b. A hypothesis (might just be a guess)

The best fitness scores will never go down since the experiment includes elitism and it will take about 150 generations to reach the best fitness score.

c. Description of the results including plots (or a table of results)



Data Collected: (Generation starts from 0 to 199, total 200 generations)

run #	1	2	3	4	5	6	7	8	9	10
gens	145	142	189	188	139	157	134	134	179	189

The three fitness lines approach to the value of 100 while maximum fitness graph only increases throughout the process of evolution. It takes approximately 140 generations to evolve the best solution.

d. What if anything you can conclude and anything you learned or affirmed by doing so?

When evolution involves elitism, the best solution always increases and although it does not reach 100, the best solution is found quickly.

Experiment #3 – Crossover and Mutation

a. An explanation of the experiment (including parameters used)

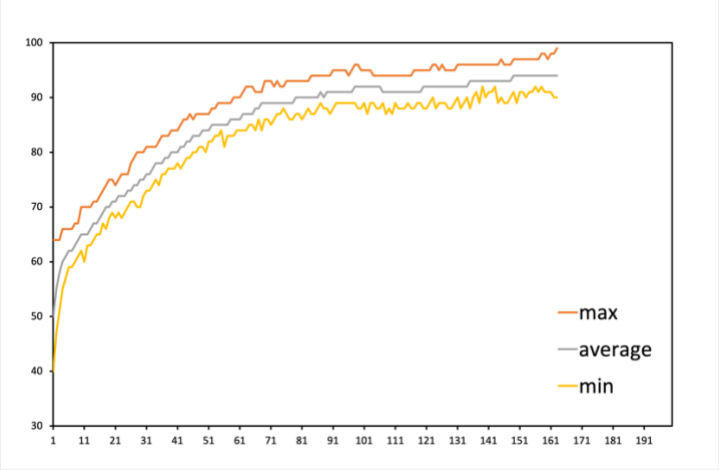
population size:	100
mutation rate:	1/100
crossover method:	One point Crossover
Number of elite:	0
chromosome length:	100
selection method:	Truncation
max generations:	200

Ran the simulator with mutation and one point crossover.

b. A hypothesis (might just be a guess)

Since the experiment includes crossover and mutation, there would be 170 generations needed to achieve the best solution.

c. Description of the results including plots (or a table of results)



Data Collected:

run #	1	2	3	4	5	6	7	8	9	10
gens	186	168	164	162	199	183	184	197	144	171

Minimum fitness scores have a lot of ups and downs whereas average and maximum fitness scores usually increase. In all of the trials the simulator found the best solution before reaching evolving 200 generations.

d. What if anything you can conclude and anything you learned or affirmed by doing so?

If there are mutations and crossovers, the best solution is found before maximum number of generations which is 200.

Experiment #4 – Only Crossover

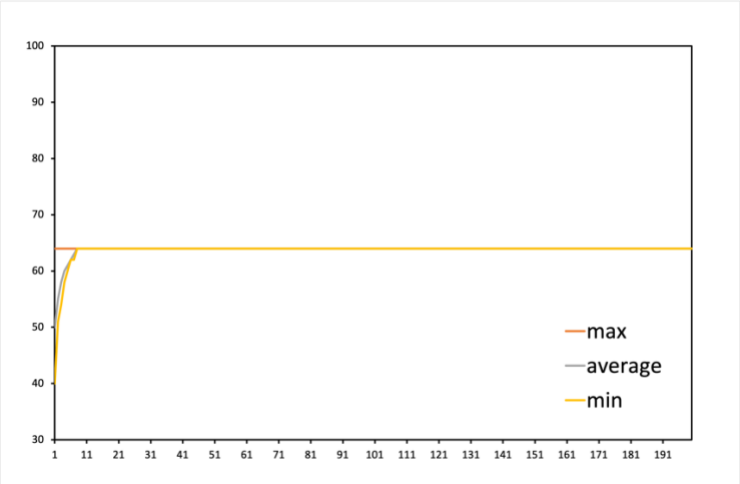
a. An explanation of the experiment (including parameters used)

population size:	100
mutation rate:	0
crossover method:	One point Crossover
Number of elite:	0
chromosome length:	100
selection method:	Truncation
max generations:	200

Ran the simulator with one point crossover, truncation method with 0 mutation rate.

b. **A hypothesis (might just be a guess)**
I expect the three plots to converge into a one value towards the end of the generation and 10 generations to evolve to the best solution.

c. Description of the results including plots (or a table of results)



Data Collected:

run #	1	2	3	4	5	6	7	8	9	10
gens	0	0	0	0	0	0	0	0	0	0

The best fitness line converges due to the lack of diversity of crossover only. The best individual in generation 0 stays as the best individual throughout all the generations.

d. What if anything you can conclude and anything you learned or affirmed by doing so?
If there is no mutation, then the best individual in generation 0 becomes the best even after evolution.

Experiment #5 – Different Mutation Rates

a. An explanation of the experiment (including parameters used)

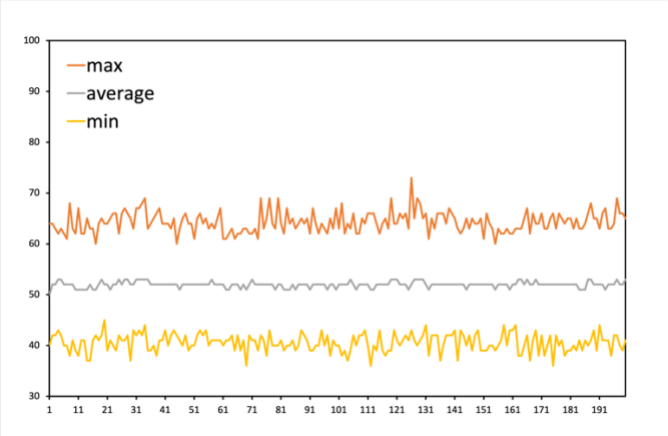
population size:	100
mutation rate:	vary
crossover method:	None
Number of elite:	0
chromosome length:	100
selection method:	Truncation
max generations:	200

Ran the simulator with the parameters above with mutation varying from 0.0001 percent to 75 percent.

c. A hypothesis (might just be a guess)

Mutation rates increase as the best solution would decrease and it would take around 20 generations to reach the best solution.

d. Description of the results including plots (or a table of results)



: mutation rate of 50%

Data Collected:

run #	1	2	3	4	5	6	7	8	9	10
gens	0	0	199	33	45	57	23	47	29	1

It appears as the mutation rate increases, the value of the best solution decreases, and the best solution is found quickly. The three lines look almost constant although they do have some ups and downs, but still have similar difference between max and average, and average and min.

d. What if anything you can conclude and anything you learned or affirmed by doing so?

As the evolution mutation rate increases, the best solution is found quickly as the solution to be a lower number compared to other experiments. If the mutation rate is very high, then the best fitness is found at generation 0 whereas if the mutation rate is very low, the best fitness is the last generation.