Part1)

Genetic Algorithm is a nature-inspired metaheuristic optimization method which is used if we can represent our solutions in binary where there's a logical connection between binary representation and real world representation. In this case, we can represent our x1 and x2 integers with 3 bit binary since the maximum number can x1 and x2 get is 5. For example (1,2) can be represented by (001010) and adding 1 in the world representation would be the same as adding 1 in binary representation which is a great relationship between the two representations. The goal is to optimizing finding the maximized value of f (x1, x2)=20*x1*x2 + 16*x^2 – 2*x1^2 − x2^2 − (x1+x2)^2 function with $x_1+x_2 \leqslant 5$;$0 \leqslant x_1 \leqslant 5$;$0 \leqslant x_2 \leqslant 5$ these constraints.

There are multiple selection and crossover algorithms and I've implemented one point and two point crossover and roulette wheel, tournament and rank selection algorithms. Strategy design pattern is used for Selection and crossover they extend Abstract Selection and Crossover class to implement new selection and crossover methods.

Template design pattern is utilized for applying selection and crossover onto the population. Since the only thing different with different combinations of crossover and selection methods are crossover and selections nothing changes in the template method. In order to introduce different combinations of selection and crossovers, all that is required is extending abstract SelectionAndCrossover class and in the constructor, desired selection and crossover can be used.

The stopping condition of the Genetic Algorithm here is having no changes on the maximum found value of the function for 10 iterations. This number was a result of trial and error. The population pool was tested for 6 and 4 but can be changed by changing POPULATION_COUNT in the classes.

In terms of run time, the roulette wheel selection with one point crossover seem to be the one that runs the slowest with avarage of 55ms whereas the Tournament selection with one point crossover is the fastest with avg of 1.5ms. Rank selection with two point crossover avarages 30ms. All 3 implementations find the maximum value (f(2,3) = 126) most of the time with occasional second max value(f(3,2) = 105).

Example Run:

```
Max found: 126
Roulette with One Point xover: [2, 3]
Time elapsed: 53ms.
Max found: 126
Tournament with One Point xover: [2, 3]
Time elapsed: 1ms.
Max found: 126
Rank with Two Point xover: [2, 3]
Time elapsed: 27ms.
```

Class Diagram: