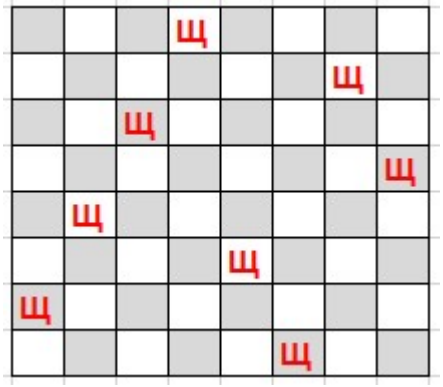
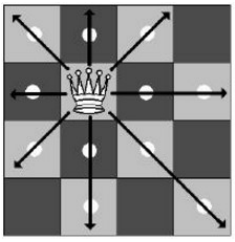
Practice: 8 queen problem with Genetic Algorithms

The problem consists of placing 8 queens on an 8x8 chessboard. The queens must not attack among them.



A queen attacks all the pieces that are in the same row, column or diagonal.



wikipedia.org

For calculating if a queen attacks another, we need to identify if they stay in the same row, column or diagonal. The diagonal can be calculated by adding or subtracting the rows and the columns.

row + column

	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	8
2	2	3	4	5	6	7	8	9
3	3	4	5	6	7	8	9	10
4	4	5	6	7	8	9	10	11
5	5	6	7	8	9	10	11	12
6	6	7	8	9	10	11	12	13
7	7	8	9	10	11	12	13	14

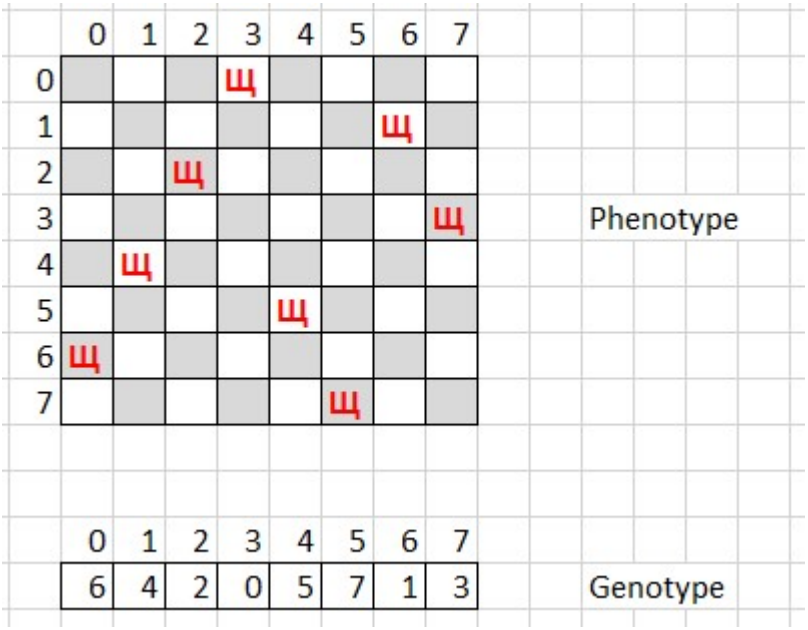
row - column

	0	1	2	3	4	5	6	7
0	0	-1	-2	-3	-4	-5	-6	-7
1	1	0	-1	-2	-3	-4	-5	-6
2	2	1	0	-1	-2	-3	-4	-5
3	3	2	1	0	-1	-2	-3	-4
4	4	3	2	1	0	-1	-2	-3
5	5	4	3	2	1	0	-1	-2
6	6	5	4	3	2	1	0	-1
7	7	6	5	4	3	2	1	0

The recommendation is the following:

- Representation: permutation
- Parents' selection: roulette
- Crossover: partially mapped crossover
- Mutation: swap
- Population model: generational

Representation



The **genotype** is a permutation of numbers where the array's position represents the column and the value the row of each queen.

Search space:
As a permutation, the search space is
= 8*7*6*5*4*3*2*1
= 5.05658 e+15

Step 1: Initial population

Create a function for generating N individuals as random permutations.

Step 2: Fitness function

Given an individual (permutation) we can calculate the number of queens that attacks at least another queen q_a. Then, the worst solution will have $q_a = 8$, this is, 8 queens that attack others, and the best solution will have $q_a = 0$, this is, 0 queens with attacks. For transforming this number to a fitness we can use $f = 8 - q_a + 1$.

Step 3: Implement the parent selection function

The function's input is the whole population and the fitness's vector, the output is the new parents' population. The implementation must be using the roulette methodology.

Step 4: Implement the crossover function

The function's input is the parents' population and the crossover probability, the output is the new offspring population. The implementation must be using partially mapped crossover.

Step 5: Implement the mutation function

The function's input is the offspring' population and the mutation probability, the output is the new mutated population. The implementation must be using swap mutation.

Step 6: Implement the function to manage the elite

The function's input is the new population, its fitness, the elite, and its fitness, the output is the population and its fitness. If in the new population, we find an individual better than the elite, then we replace the elite with the fittest individual. Else, we select an individual from the population with negative tournament and replace the selected individual with the elite.

Step 7: Implement the genetic algorithm

Implement the algorithm that use all functions for evolving individuals and solve the traveling salesman problem.

Step 8: Draw the problems' solution

At the end of the algorithm, draw the solution that the best individual represents. It must be a solution of the 8 queens problem.

Evaluation	
Element	Points
Step 1: Create the initial population	5
Step 2: Create a function for calculating the individual's fitness	10
Step 3: Implement the parent selection function	15
Step 4: Implement the crossover function	15
Step 5: Implement the mutation function	15
Step 6: Implement the function to manage the elite	15
Step 7: Implement the genetic algorithm	15
Step 8: Draw the problems' solution	10