CS 404 – Artificial Intelligence
HW 3 – Local Search

Goal: To learn more about local search which has very low memory usage and can be quite successful for many problems; and to gain further experience with programing and reporting research results.

Task: Solve the N-queen problem with different local search algorithms. N-queen problem is finding the placement of N queens on an NxN board such that noone attacks one another. The given Python code implements basic hill climbing and random restarts. You will be asked to a) run simulation experiments with the given code and b) expand the code with stochastic hill climbing.

a**) 50 pts**- Given the code in link , run **100** simulations/experiments with different initial solutions and give the number of **successes, number of iterations, and the time it takes** to find the solution on **average** in each case, for **N=10 and N=20,** for **basic** hill climbing and **random restart** with increasing number of restarts (k=10, 100, 1000)

	N=10			**N=20**		
	Percentage of success in 100 runs	**Solutions found in how many restarts on average**	**Elapsed time to complete experiments**	**Percentage of success in 100 runs**	**Solutions found in how many restarts on average**	**Elapsed time to complete experiments**
Basic Hill Climbing	0.06	-	0.010289	0.02	-	0.265
Random Restart with k=10	0.48	3.9375	0.07285	0.21	3.14	2.25
Random Restart with k=100	1	16.02	0.1444	0.87	29.39	8.97
Random Restart with k=1000	1	15.1	0.1370	1	42.56	10.75
Stochastic hill climbing *(to fill for part b)*	0.06	-	0.0136	0.02	-	0.405
Simulated Annealing *(to fill for part c) – if you will do the bonus)*	İnitial T=10000 **0.36** (for best scheduler in part c)	-	1.78	0.17	-	7.38

b) **50 pts**- Add a new function randomNeighbor(....) to implement **stochastic hill climbing**. If no better neighbour, should return current one. Leave other code the same.

Fill the results of 100 experiments to the corresponding row of the table with stochastic hill climbing.

c) Bonus-**15 pts**: Implement **simulated annealing** and fill the results of 100 experiments to the corresponding row of the table. Specify the best parameters you found (what is the schedule and initial temperature) here or in the table.

ANSWER IS ON THE NEXT PAGE

C-)
I tried using different schedulers as (T = T - 10), (T = T - 5), (T = T - 2), (T = T – 0.5) with initial temperature = 1000, 10000, 10000:
As the initial temperature increases, the success probability approaches to 1. For instance,

For T = 1000 -> **success rate:** 0.1
For T = 10000 -> **success rate:** 0.36
For T = 100000 -> **success rate:** definitely **better** than above ones, but takes too much time WITH THE BEST SCHEDULER **(T = T – 0.5).**

However, when we increase the temperature the elapsed time to complete experiments also increase. Therefore, I can not choose high temperature values for more complex problems such as N = 20.

COLAB LINK:
https://colab.research.google.com/drive/1lJs5Qzqo_5HmjJ2Nnj0bewgCWO8A41ak?usp=sharing