import numpy as np

import random

def population\_set(n = 8, start\_population = 8):

return selection(n, start\_population)

def selection(n, start\_population):

individualss = []

max\_posibilities = (n \* (n - 1)) / 2 #28

while(len(individualss) <= n):

temp\_arr = np.random.randint(0, n, (start\_population,))

individual\_fitness = fitness(temp\_arr)

if individual\_fitness / max\_posibilities > 0.18:

individualss.append(temp\_arr)

return np.array(individualss)

def crossover(x, y):

#random number between 1 to 6 (inclusive) to improvise crossover

n = np.random.randint(1, 7)

checker = np.concatenate([np.array(x[:n]), np.array(y[n:])], axis = 0).tolist()

return checker

def probability\_test(child):

mutation\_threshold = 0.8

if np.random.uniform(0, 1) < mutation\_threshold:

return mutate(child)

else:

return child

def fitness(x):

horizontal, diagonal = 0, 0

for i in range(len(x)):

for j in range(i + 1, len(x)):

if(x[i] == x[j]):

horizontal += 1

elif(abs(i - j) == abs(x[i] - x[j])):

diagonal += 1

total = horizontal + diagonal

value = int((len(x) \* (len(x) - 1)) / 2) - total

return value

def fitness\_loc(\_\_child, fitness\_array, location\_array, fit\_max):

k\_fit = fitness(\_\_child)

fitness\_array.append(k\_fit)

location\_array.append(\_\_child)

if fit\_max < k\_fit:

fit\_max = max(fitness\_array)

location\_address = fitness\_array.index(fit\_max)

return fit\_max, location\_array[location\_address]

def mutate(children):

x\_i = np.random.randint(0, 8)

y\_i = np.random.randint(0, 8)

if children[x\_i] != y\_i:

children[x\_i] = y\_i

else:

children[x\_i] = np.random.randint(0, 8)

return children

def genetic\_algorithm(population):

fitness\_array = []

location\_array = []

fit\_max = 0

loc\_max = 0

for i in range(100000):

new\_population = []

for i in range(len(population)):

x = population[np.random.randint(0, len(population))]

y = population[np.random.randint(0, len(population))]

child = crossover(x, y)

#print(child)

child = probability\_test(child)

fit\_max, loc\_max = fitness\_loc(child, fitness\_array, location\_array, fit\_max)

new\_population.append(child)

population = new\_population

if(fit\_max == 28):

print("Found")

print(loc\_max)

break

if \_\_name\_\_ == "\_\_main\_\_":

genetic\_algorithm(population\_set(8, 8))