8Queens_Problem_using_Genetic_Algorithm

February 20, 2022

The goal of this assignment is to implement a genetic algorithm to solve 8 queens problem

Import necessary libraries

[1]:
```
import random
import matplotlib.pyplot as plt
```

[2]:
```
class Individual:
    def __init__(self, indiv):
        self.indiv = indiv
        self.fitness = self.fitnessfunction()
        self.si = 2          #any nonsensical number

    def fitnessfunction(self) -> int:
        nonattacking = 0
        for i in range(len(self.indiv)):
            h = i - 1
            j = i + 1
            right = left = 1
            while (h >= 0 or j < len(self.indiv)):
                if h>= 0 :
                    if self.indiv[h] != self.indiv[i] and self.indiv[h] != self.
↪indiv[i]-left and self.indiv[h] != self.indiv[i]+left :
                        nonattacking += 1
                    left += 1
                    h -= 1
                if j < len(self.indiv):
                    if self.indiv[j] != self.indiv[i] and self.indiv[j] != self.
↪indiv[i]-right and self.indiv[j] != self.indiv[i]+right :
                        nonattacking += 1
                    right += 1
                    j += 1
        nonattacking = nonattacking/2
        return nonattacking
```

Selection of parents

[3]:
```
def roulette(population):
    x = random.random()
```

```
    current = 0
    for individual in population:
        current += individual.si
        if current > x:
            return individual
```

[4]:
```
def newPopulation(population):
    newPopulation = []
    while len(newPopulation) < PopulationSize:
        parent1 = roulette(population).indiv
        parent2 = roulette(population).indiv
        crossover = random.randint(0, 7)
        child1 = parent1[:crossover] + parent2[crossover:]
        child2 = parent2[:crossover] + parent1[crossover:]
        mutation(child1)
        mutation(child2)
        newPopulation.append(Individual(child1))
        newPopulation.append(Individual(child2))
    sumfitness = sum(c.fitness for c in newPopulation)
    for individual in newPopulation:
        individual.si = individual.fitness / sumfitness
    avgfitness = sumfitness/PopulationSize
    return (newPopulation, avgfitness)
```

Defining mutation

[5]:
```
def mutation(child):
    if(random.random() < MutationPct):
        index = random.randint(0,7)
        child[index] = random.randint(1,8)
```

[6]:
```
def myplot(numofgenerations, avgfitnessarray):
    plt.plot(numofgenerations, avgfitnessarray)
    plt.ylabel("Average fitness")
    plt.xlabel("Generation")
    #plt.yticks(range(0,30,2))
    #plt.xticks(range(0,100,10))
    plt.ylim(0,28)
    plt.tight_layout()
    plt.show()
```

[7]:
```
#MutationPct can be a random value between 0.0 and 1.0
MutationPct = 0.3
#PopulationSize can be 10,100,500,1000 etc
PopulationSize = 100
#Number of Iterations
NumIterations = 500
population = []
```

[8]:
```python
for i in range(PopulationSize):
   population.append(Individual(random.sample(range(1,9),8)))
sumfitness = sum(c.fitness for c in population)
for individual in population:
    individual.si = individual.fitness / sumfitness
avgfitness = sumfitness/PopulationSize
print("Initial population")
for individual in population:
    print(individual.indiv, '  ', individual.fitness, '  ', individual.si)
```

```
Initial population
[3, 7, 5, 2, 1, 6, 4, 8]    25.0    0.010813148788927335
[6, 8, 5, 3, 7, 2, 1, 4]    21.0    0.009083044982698962
[1, 2, 8, 4, 6, 5, 7, 3]    16.0    0.006920415224913495
[6, 1, 4, 2, 5, 8, 3, 7]    25.0    0.010813148788927335
[7, 4, 8, 6, 5, 2, 1, 3]    22.0    0.009515570934256055
[6, 5, 2, 3, 8, 4, 7, 1]    23.0    0.009948096885813149
[4, 3, 1, 2, 8, 6, 5, 7]    22.0    0.009515570934256055
[6, 7, 2, 3, 4, 1, 8, 5]    20.0    0.00865051903114187
[7, 6, 5, 4, 8, 3, 1, 2]    17.0    0.007352941176470588
[2, 8, 5, 7, 3, 6, 4, 1]    26.0    0.01124567474048443
[8, 4, 7, 1, 3, 5, 6, 2]    26.0    0.01124567474048443
[4, 5, 7, 3, 1, 2, 8, 6]    25.0    0.010813148788927335
[6, 1, 7, 2, 8, 3, 4, 5]    24.0    0.010380622837370242
[4, 5, 8, 3, 1, 6, 2, 7]    25.0    0.010813148788927335
[2, 4, 8, 5, 7, 1, 3, 6]    26.0    0.01124567474048443
[4, 2, 5, 3, 7, 6, 8, 1]    25.0    0.010813148788927335
[3, 6, 8, 7, 5, 2, 1, 4]    23.0    0.009948096885813149
[2, 1, 8, 7, 4, 3, 5, 6]    23.0    0.009948096885813149
[3, 5, 1, 8, 2, 7, 4, 6]    24.0    0.010380622837370242
[8, 2, 1, 7, 3, 6, 4, 5]    23.0    0.009948096885813149
[3, 6, 7, 2, 4, 1, 8, 5]    27.0    0.011678200692041523
[8, 3, 5, 6, 4, 7, 2, 1]    20.0    0.00865051903114187
[5, 7, 8, 4, 2, 1, 6, 3]    24.0    0.010380622837370242
[3, 4, 6, 1, 7, 8, 2, 5]    20.0    0.00865051903114187
[3, 8, 1, 5, 7, 2, 6, 4]    24.0    0.010380622837370242
[5, 7, 8, 2, 4, 1, 3, 6]    24.0    0.010380622837370242
[5, 2, 3, 7, 8, 1, 6, 4]    24.0    0.010380622837370242
[5, 7, 4, 1, 8, 6, 2, 3]    26.0    0.01124567474048443
[5, 3, 7, 1, 6, 8, 4, 2]    22.0    0.009515570934256055
[5, 6, 3, 8, 7, 1, 2, 4]    20.0    0.00865051903114187
[1, 4, 2, 8, 5, 3, 7, 6]    24.0    0.010380622837370242
[4, 6, 1, 5, 3, 2, 8, 7]    22.0    0.009515570934256055
[7, 4, 6, 3, 5, 2, 1, 8]    24.0    0.010380622837370242
[8, 4, 6, 1, 2, 3, 5, 7]    22.0    0.009515570934256055
[6, 4, 8, 7, 1, 2, 3, 5]    22.0    0.009515570934256055
[3, 2, 5, 8, 4, 7, 6, 1]    23.0    0.009948096885813149
[6, 8, 1, 5, 7, 4, 3, 2]    21.0    0.009083044982698962
```

[3, 1, 8, 4, 7, 2, 5, 6]	24.0	0.010380622837370242
[3, 5, 6, 4, 1, 2, 8, 7]	24.0	0.010380622837370242
[5, 1, 7, 3, 8, 2, 4, 6]	26.0	0.01124567474048443
[1, 6, 8, 7, 2, 4, 3, 5]	25.0	0.010813148788927335
[5, 2, 8, 1, 4, 3, 7, 6]	24.0	0.010380622837370242
[5, 3, 4, 2, 6, 1, 8, 7]	19.0	0.008217993079584774
[3, 7, 4, 1, 8, 6, 5, 2]	27.0	0.011678200692041523
[8, 3, 1, 7, 2, 6, 4, 5]	24.0	0.010380622837370242
[2, 4, 8, 6, 5, 7, 1, 3]	24.0	0.010380622837370242
[2, 7, 8, 1, 4, 6, 5, 3]	24.0	0.010380622837370242
[3, 1, 4, 7, 6, 5, 2, 8]	23.0	0.009948096885813149
[5, 4, 3, 6, 1, 8, 2, 7]	19.0	0.008217993079584774
[6, 8, 7, 4, 5, 2, 3, 1]	19.0	0.008217993079584774
[6, 8, 4, 3, 1, 7, 5, 2]	23.0	0.009948096885813149
[2, 3, 6, 8, 7, 1, 4, 5]	25.0	0.010813148788927335
[7, 5, 6, 1, 8, 4, 3, 2]	22.0	0.009515570934256055
[7, 6, 2, 8, 4, 3, 5, 1]	21.0	0.009083044982698962
[8, 7, 4, 5, 2, 6, 3, 1]	20.0	0.00865051903114187
[6, 8, 5, 4, 3, 7, 1, 2]	20.0	0.00865051903114187
[2, 3, 1, 8, 7, 4, 6, 5]	24.0	0.010380622837370242
[5, 7, 4, 8, 3, 1, 2, 6]	23.0	0.009948096885813149
[1, 4, 5, 8, 7, 6, 3, 2]	20.0	0.00865051903114187
[6, 8, 7, 4, 3, 2, 1, 5]	21.0	0.009083044982698962
[3, 5, 8, 7, 2, 4, 1, 6]	24.0	0.010380622837370242
[5, 1, 4, 6, 2, 7, 3, 8]	25.0	0.010813148788927335
[3, 4, 8, 5, 6, 2, 1, 7]	24.0	0.010380622837370242
[5, 2, 6, 3, 8, 7, 4, 1]	25.0	0.010813148788927335
[3, 1, 5, 8, 4, 6, 2, 7]	22.0	0.009515570934256055
[2, 4, 6, 7, 3, 1, 5, 8]	26.0	0.01124567474048443
[8, 4, 6, 1, 2, 3, 5, 7]	22.0	0.009515570934256055
[7, 4, 2, 1, 8, 3, 6, 5]	20.0	0.00865051903114187
[4, 1, 6, 8, 5, 7, 3, 2]	24.0	0.010380622837370242
[3, 5, 6, 2, 1, 7, 4, 8]	26.0	0.01124567474048443
[3, 6, 2, 8, 4, 5, 7, 1]	23.0	0.009948096885813149
[4, 1, 7, 3, 5, 8, 2, 6]	25.0	0.010813148788927335
[6, 5, 2, 8, 7, 4, 1, 3]	26.0	0.01124567474048443
[6, 4, 3, 5, 7, 2, 8, 1]	24.0	0.010380622837370242
[7, 3, 6, 1, 8, 2, 4, 5]	21.0	0.009083044982698962
[5, 4, 2, 7, 1, 3, 6, 8]	24.0	0.010380622837370242
[8, 7, 4, 6, 5, 2, 1, 3]	25.0	0.010813148788927335
[4, 3, 5, 7, 1, 6, 8, 2]	25.0	0.010813148788927335
[3, 5, 7, 6, 1, 8, 2, 4]	23.0	0.009948096885813149
[6, 5, 4, 3, 2, 8, 7, 1]	17.0	0.007352941176470588
[2, 6, 8, 1, 5, 3, 7, 4]	26.0	0.01124567474048443
[5, 7, 2, 4, 1, 3, 8, 6]	26.0	0.01124567474048443
[5, 2, 3, 6, 1, 4, 7, 8]	18.0	0.007785467128027681
[1, 8, 7, 6, 3, 4, 2, 5]	21.0	0.009083044982698962
[3, 7, 2, 5, 1, 6, 8, 4]	24.0	0.010380622837370242

```
[7, 2, 6, 8, 4, 1, 3, 5]    27.0    0.011678200692041523
[6, 7, 4, 3, 8, 2, 5, 1]    23.0    0.009948096885813149
[1, 2, 8, 3, 5, 7, 4, 6]    24.0    0.010380622837370242
[3, 1, 7, 5, 8, 6, 2, 4]    26.0    0.01124567474048443
[6, 4, 5, 8, 2, 7, 1, 3]    25.0    0.010813148788927335
[5, 8, 1, 3, 2, 4, 6, 7]    22.0    0.009515570934256055
[6, 4, 8, 5, 3, 1, 7, 2]    26.0    0.01124567474048443
[6, 8, 4, 1, 2, 3, 5, 7]    22.0    0.009515570934256055
[1, 4, 7, 5, 2, 3, 8, 6]    25.0    0.010813148788927335
[1, 2, 4, 3, 6, 5, 8, 7]    18.0    0.007785467128027681
[6, 3, 1, 5, 2, 4, 7, 8]    24.0    0.010380622837370242
[6, 4, 3, 7, 2, 1, 8, 5]    23.0    0.009948096885813149
[3, 5, 4, 8, 6, 7, 1, 2]    23.0    0.009948096885813149
[6, 2, 7, 3, 5, 8, 1, 4]    25.0    0.010813148788927335
[7, 1, 2, 8, 3, 4, 6, 5]    22.0    0.009515570934256055
```

[9]:
```
count = 0
numofgenerations = [0]
avgfitnessarray = [avgfitness]
```

[10]:
```
while count < NumIterations:
    count += 1
    numofgenerations.append(count)
    print("Iteration: ", count)
    (population, avgfitness) = newPopulation(population)
    avgfitnessarray.append(avgfitness)
    # print(avgfitness)
    # for individual in population:
    #         print(individual.indiv, '   ', individual.fitness, '   ', individual.
 ↪si)
    for individual in population:
        if individual.fitness == 28:
            for elem in population:
                    print(elem.indiv)
            print("Resolved: ", individual.indiv)
            myplot(numofgenerations, avgfitnessarray)
            exit(1)
```

```
Iteration:  1
Iteration:  2
Iteration:  3
Iteration:  4
Iteration:  5
Iteration:  6
Iteration:  7
Iteration:  8
Iteration:  9
Iteration:  10
```

```
Iteration:  11
Iteration:  12
Iteration:  13
Iteration:  14
Iteration:  15
Iteration:  16
Iteration:  17
Iteration:  18
Iteration:  19
Iteration:  20
Iteration:  21
Iteration:  22
Iteration:  23
Iteration:  24
Iteration:  25
Iteration:  26
Iteration:  27
Iteration:  28
Iteration:  29
Iteration:  30
Iteration:  31
Iteration:  32
Iteration:  33
Iteration:  34
Iteration:  35
Iteration:  36
Iteration:  37
Iteration:  38
Iteration:  39
Iteration:  40
Iteration:  41
Iteration:  42
Iteration:  43
Iteration:  44
Iteration:  45
Iteration:  46
Iteration:  47
Iteration:  48
Iteration:  49
Iteration:  50
Iteration:  51
Iteration:  52
Iteration:  53
Iteration:  54
Iteration:  55
Iteration:  56
Iteration:  57
Iteration:  58
```

```
Iteration:  59
Iteration:  60
Iteration:  61
Iteration:  62
Iteration:  63
Iteration:  64
Iteration:  65
Iteration:  66
Iteration:  67
Iteration:  68
Iteration:  69
Iteration:  70
Iteration:  71
Iteration:  72
Iteration:  73
Iteration:  74
Iteration:  75
Iteration:  76
Iteration:  77
Iteration:  78
Iteration:  79
Iteration:  80
Iteration:  81
Iteration:  82
Iteration:  83
Iteration:  84
Iteration:  85
Iteration:  86
Iteration:  87
Iteration:  88
Iteration:  89
Iteration:  90
Iteration:  91
Iteration:  92
Iteration:  93
Iteration:  94
Iteration:  95
Iteration:  96
Iteration:  97
Iteration:  98
Iteration:  99
Iteration:  100
Iteration:  101
Iteration:  102
Iteration:  103
Iteration:  104
Iteration:  105
Iteration:  106
```

```
Iteration:  107
Iteration:  108
Iteration:  109
Iteration:  110
Iteration:  111
Iteration:  112
Iteration:  113
Iteration:  114
Iteration:  115
Iteration:  116
Iteration:  117
Iteration:  118
Iteration:  119
Iteration:  120
Iteration:  121
Iteration:  122
Iteration:  123
Iteration:  124
Iteration:  125
Iteration:  126
Iteration:  127
Iteration:  128
Iteration:  129
Iteration:  130
Iteration:  131
Iteration:  132
Iteration:  133
Iteration:  134
Iteration:  135
Iteration:  136
Iteration:  137
Iteration:  138
Iteration:  139
Iteration:  140
Iteration:  141
Iteration:  142
Iteration:  143
Iteration:  144
Iteration:  145
Iteration:  146
Iteration:  147
Iteration:  148
Iteration:  149
Iteration:  150
Iteration:  151
Iteration:  152
Iteration:  153
Iteration:  154
```

```
Iteration:  155
Iteration:  156
Iteration:  157
Iteration:  158
Iteration:  159
Iteration:  160
Iteration:  161
Iteration:  162
Iteration:  163
Iteration:  164
Iteration:  165
Iteration:  166
Iteration:  167
Iteration:  168
Iteration:  169
Iteration:  170
Iteration:  171
Iteration:  172
Iteration:  173
Iteration:  174
Iteration:  175
Iteration:  176
Iteration:  177
Iteration:  178
Iteration:  179
Iteration:  180
Iteration:  181
Iteration:  182
Iteration:  183
Iteration:  184
Iteration:  185
Iteration:  186
Iteration:  187
Iteration:  188
Iteration:  189
Iteration:  190
Iteration:  191
Iteration:  192
Iteration:  193
Iteration:  194
Iteration:  195
Iteration:  196
Iteration:  197
Iteration:  198
Iteration:  199
Iteration:  200
Iteration:  201
Iteration:  202
```

```
Iteration:  203
Iteration:  204
Iteration:  205
Iteration:  206
Iteration:  207
Iteration:  208
Iteration:  209
Iteration:  210
Iteration:  211
Iteration:  212
Iteration:  213
Iteration:  214
Iteration:  215
Iteration:  216
Iteration:  217
Iteration:  218
Iteration:  219
Iteration:  220
Iteration:  221
Iteration:  222
Iteration:  223
Iteration:  224
Iteration:  225
Iteration:  226
Iteration:  227
Iteration:  228
Iteration:  229
Iteration:  230
Iteration:  231
Iteration:  232
Iteration:  233
Iteration:  234
Iteration:  235
Iteration:  236
Iteration:  237
Iteration:  238
Iteration:  239
Iteration:  240
Iteration:  241
Iteration:  242
Iteration:  243
Iteration:  244
Iteration:  245
Iteration:  246
Iteration:  247
Iteration:  248
Iteration:  249
Iteration:  250
```

```
Iteration:  251
Iteration:  252
Iteration:  253
Iteration:  254
Iteration:  255
Iteration:  256
Iteration:  257
Iteration:  258
Iteration:  259
Iteration:  260
Iteration:  261
Iteration:  262
Iteration:  263
Iteration:  264
Iteration:  265
Iteration:  266
Iteration:  267
Iteration:  268
Iteration:  269
Iteration:  270
Iteration:  271
Iteration:  272
Iteration:  273
Iteration:  274
Iteration:  275
Iteration:  276
Iteration:  277
Iteration:  278
Iteration:  279
Iteration:  280
Iteration:  281
Iteration:  282
Iteration:  283
Iteration:  284
Iteration:  285
Iteration:  286
Iteration:  287
Iteration:  288
Iteration:  289
Iteration:  290
Iteration:  291
Iteration:  292
Iteration:  293
Iteration:  294
Iteration:  295
Iteration:  296
Iteration:  297
Iteration:  298
```

```
Iteration:  299
Iteration:  300
Iteration:  301
Iteration:  302
Iteration:  303
Iteration:  304
Iteration:  305
Iteration:  306
Iteration:  307
Iteration:  308
Iteration:  309
Iteration:  310
Iteration:  311
Iteration:  312
Iteration:  313
Iteration:  314
Iteration:  315
Iteration:  316
Iteration:  317
Iteration:  318
Iteration:  319
Iteration:  320
Iteration:  321
Iteration:  322
Iteration:  323
Iteration:  324
Iteration:  325
Iteration:  326
Iteration:  327
Iteration:  328
Iteration:  329
Iteration:  330
Iteration:  331
Iteration:  332
Iteration:  333
Iteration:  334
Iteration:  335
Iteration:  336
Iteration:  337
Iteration:  338
Iteration:  339
Iteration:  340
Iteration:  341
Iteration:  342
Iteration:  343
Iteration:  344
Iteration:  345
Iteration:  346
```

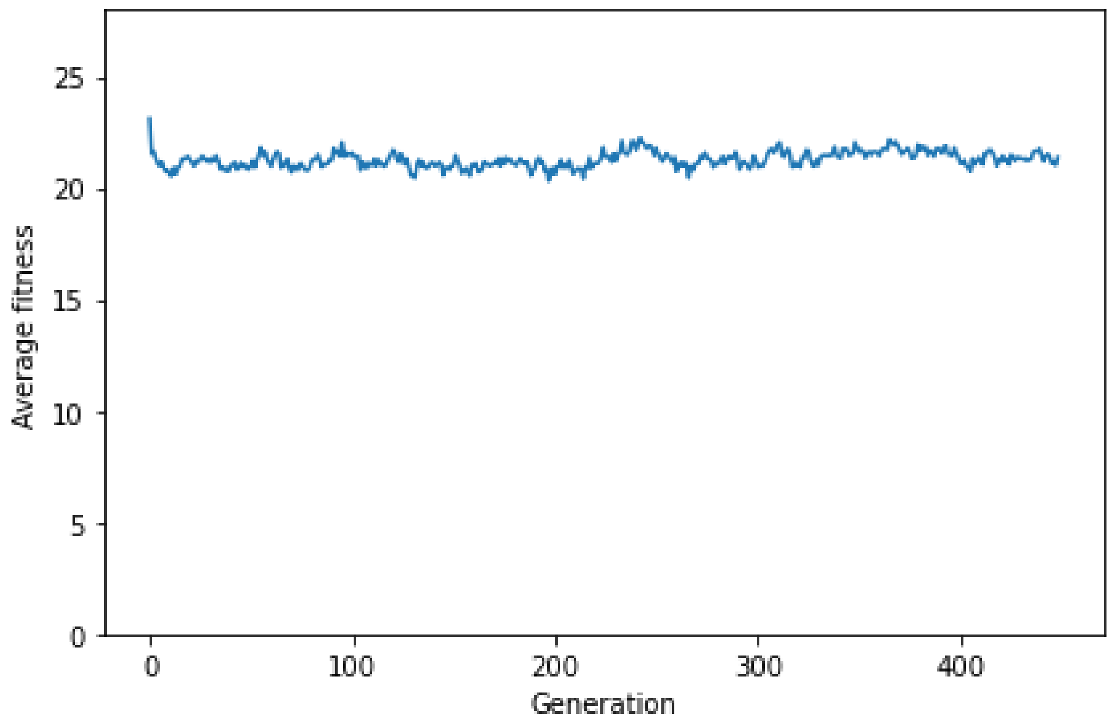
```
Iteration:  347
Iteration:  348
Iteration:  349
Iteration:  350
Iteration:  351
Iteration:  352
Iteration:  353
Iteration:  354
Iteration:  355
Iteration:  356
Iteration:  357
Iteration:  358
Iteration:  359
Iteration:  360
Iteration:  361
Iteration:  362
Iteration:  363
Iteration:  364
Iteration:  365
Iteration:  366
Iteration:  367
Iteration:  368
Iteration:  369
Iteration:  370
Iteration:  371
Iteration:  372
Iteration:  373
Iteration:  374
Iteration:  375
Iteration:  376
Iteration:  377
Iteration:  378
Iteration:  379
Iteration:  380
Iteration:  381
Iteration:  382
Iteration:  383
Iteration:  384
Iteration:  385
Iteration:  386
Iteration:  387
Iteration:  388
Iteration:  389
Iteration:  390
Iteration:  391
Iteration:  392
Iteration:  393
Iteration:  394
```

```
Iteration:  395
Iteration:  396
Iteration:  397
Iteration:  398
Iteration:  399
Iteration:  400
Iteration:  401
Iteration:  402
Iteration:  403
Iteration:  404
Iteration:  405
Iteration:  406
Iteration:  407
Iteration:  408
Iteration:  409
Iteration:  410
Iteration:  411
Iteration:  412
Iteration:  413
Iteration:  414
Iteration:  415
Iteration:  416
Iteration:  417
Iteration:  418
Iteration:  419
Iteration:  420
Iteration:  421
Iteration:  422
Iteration:  423
Iteration:  424
Iteration:  425
Iteration:  426
Iteration:  427
Iteration:  428
Iteration:  429
Iteration:  430
Iteration:  431
Iteration:  432
Iteration:  433
Iteration:  434
Iteration:  435
Iteration:  436
Iteration:  437
Iteration:  438
Iteration:  439
Iteration:  440
Iteration:  441
Iteration:  442
```

```
Iteration:  443
Iteration:  444
Iteration:  445
Iteration:  446
Iteration:  447
Iteration:  448
[1, 3, 8, 6, 1, 5, 1, 7]
[7, 1, 3, 3, 5, 8, 8, 3]
[8, 3, 8, 7, 7, 1, 2, 7]
[1, 7, 4, 2, 2, 5, 1, 1]
[2, 3, 4, 2, 7, 5, 1, 4]
[6, 3, 7, 8, 3, 4, 4, 5]
[4, 4, 4, 3, 7, 2, 8, 3]
[2, 1, 8, 8, 5, 8, 3, 7]
[1, 1, 4, 6, 8, 2, 7, 4]
[8, 3, 8, 2, 2, 5, 1, 3]
[4, 4, 4, 6, 5, 2, 1, 5]
[5, 8, 2, 3, 7, 2, 3, 7]
[2, 6, 6, 8, 1, 1, 8, 3]
[2, 8, 3, 6, 5, 2, 5, 7]
[4, 4, 4, 3, 7, 2, 8, 6]
[6, 3, 1, 8, 7, 3, 3, 6]
[8, 3, 1, 7, 7, 4, 1, 8]
[4, 1, 4, 8, 2, 5, 1, 7]
[8, 4, 4, 6, 7, 2, 2, 7]
[6, 3, 4, 2, 3, 7, 8, 7]
[3, 3, 2, 7, 3, 6, 1, 7]
[8, 3, 8, 2, 2, 6, 1, 5]
[8, 3, 8, 2, 2, 5, 6, 4]
[8, 3, 2, 7, 3, 2, 1, 4]
[8, 3, 8, 6, 5, 2, 1, 5]
[5, 6, 2, 2, 2, 5, 1, 7]
[5, 6, 8, 3, 7, 1, 2, 5]
[1, 3, 8, 6, 5, 3, 8, 3]
[2, 6, 6, 2, 7, 1, 2, 2]
[2, 6, 6, 6, 5, 2, 5, 7]
[8, 4, 4, 3, 5, 1, 5, 7]
[6, 1, 8, 6, 3, 1, 8, 7]
[5, 8, 4, 1, 6, 4, 8, 5]
[7, 6, 4, 1, 8, 4, 5, 4]
[2, 1, 8, 8, 5, 8, 8, 6]
[8, 1, 3, 6, 8, 2, 4, 3]
[1, 3, 6, 6, 7, 2, 2, 4]
[7, 3, 7, 6, 7, 4, 4, 3]
[8, 3, 8, 2, 2, 5, 1, 4]
[8, 3, 8, 2, 7, 2, 8, 5]
[2, 1, 8, 7, 3, 6, 1, 7]
[1, 3, 2, 8, 5, 8, 8, 3]
```

```
[5, 1, 4, 6, 8, 2, 7, 3]
[1, 4, 7, 1, 7, 6, 2, 3]
[8, 1, 4, 7, 7, 2, 1, 5]
[5, 3, 4, 3, 7, 4, 8, 4]
[5, 3, 4, 3, 5, 2, 1, 4]
[1, 3, 6, 6, 7, 2, 1, 4]
[6, 6, 5, 6, 5, 2, 3, 7]
[8, 7, 4, 7, 7, 5, 1, 7]
[8, 4, 4, 6, 7, 2, 5, 2]
[2, 3, 4, 6, 5, 1, 8, 7]
[5, 3, 8, 6, 7, 2, 1, 4]
[8, 1, 4, 7, 7, 2, 7, 4]
[5, 3, 2, 7, 3, 6, 8, 5]
[8, 4, 4, 1, 7, 3, 1, 3]
[5, 4, 7, 3, 5, 2, 1, 4]
[1, 3, 6, 1, 7, 2, 2, 3]
[1, 3, 2, 7, 3, 6, 1, 7]
[8, 7, 4, 7, 7, 2, 7, 3]
[5, 3, 4, 6, 7, 2, 1, 3]
[2, 7, 4, 6, 5, 7, 1, 5]
[2, 3, 1, 8, 7, 3, 8, 6]
[6, 3, 4, 6, 7, 4, 2, 4]
[8, 1, 4, 6, 7, 2, 5, 5]
[2, 6, 4, 3, 7, 6, 2, 5]
[2, 1, 4, 6, 8, 3, 8, 3]
[4, 6, 1, 3, 7, 2, 3, 7]
[2, 6, 6, 6, 7, 2, 5, 2]
[2, 3, 4, 6, 5, 2, 5, 7]
[8, 6, 4, 2, 7, 2, 1, 3]
[5, 4, 7, 1, 7, 2, 2, 4]
[8, 6, 4, 3, 7, 6, 2, 5]
[2, 6, 4, 2, 7, 2, 1, 4]
[4, 3, 6, 2, 7, 2, 2, 7]
[5, 4, 4, 6, 3, 1, 8, 7]
[8, 1, 4, 7, 7, 8, 4, 3]
[2, 1, 8, 8, 5, 2, 1, 1]
[4, 4, 6, 6, 7, 4, 2, 4]
[5, 4, 4, 3, 7, 2, 3, 7]
[8, 3, 4, 7, 7, 2, 3, 7]
[8, 7, 2, 7, 3, 2, 1, 4]
[8, 5, 4, 6, 7, 5, 8, 3]
[5, 6, 8, 3, 7, 3, 1, 7]
[2, 6, 4, 3, 2, 2, 7, 5]
[5, 4, 2, 6, 5, 2, 1, 3]
[6, 3, 4, 3, 7, 6, 2, 5]
[2, 6, 8, 3, 3, 8, 8, 2]
[5, 8, 4, 1, 8, 4, 5, 2]
[5, 1, 3, 2, 7, 8, 5, 4]
```

```
[8, 8, 6, 6, 7, 2, 5, 7]
[1, 6, 5, 6, 7, 2, 1, 8]
[1, 8, 8, 3, 5, 6, 1, 7]
[8, 3, 2, 4, 3, 8, 8, 3]
[6, 6, 4, 6, 5, 2, 5, 5]
[2, 1, 4, 6, 7, 5, 1, 7]
[3, 6, 6, 6, 7, 4, 2, 7]
[5, 3, 4, 6, 7, 1, 2, 4]
[5, 7, 4, 6, 5, 2, 1, 5]
[1, 3, 6, 6, 7, 2, 8, 5]
Resolved:  [5, 1, 4, 6, 8, 2, 7, 3]
```




```
Iteration:  449
Iteration:  450
Iteration:  451
Iteration:  452
Iteration:  453
Iteration:  454
Iteration:  455
Iteration:  456
Iteration:  457
Iteration:  458
Iteration:  459
Iteration:  460
```

```
Iteration:  461
Iteration:  462
Iteration:  463
Iteration:  464
Iteration:  465
Iteration:  466
Iteration:  467
Iteration:  468
Iteration:  469
Iteration:  470
Iteration:  471
Iteration:  472
Iteration:  473
Iteration:  474
Iteration:  475
Iteration:  476
Iteration:  477
Iteration:  478
Iteration:  479
Iteration:  480
Iteration:  481
Iteration:  482
Iteration:  483
Iteration:  484
Iteration:  485
Iteration:  486
Iteration:  487
Iteration:  488
Iteration:  489
Iteration:  490
Iteration:  491
Iteration:  492
Iteration:  493
Iteration:  494
Iteration:  495
Iteration:  496
Iteration:  497
Iteration:  498
Iteration:  499
Iteration:  500
```

[11]:
```
print(count, " Iterations completed")
```

```
500  Iterations completed
```