In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
def generate_population():
    chromosome = [[i, j, k, l, m, n, o, p]
           for i in range(1, 9)
           for j in range(1, 9)
           for k in range(1, 9)
           for l in range(1, 9)
           for m in range(1, 9)
           for n in range(1, 9)
           for o in range(1, 9)
           for p in range(1, 9)
           if all([i != j, i != k, i != l, i != m, i != n, i != o, i != p,
                   j != k, j != l, j != m, j != n, j != o, j != p,
                   k != l, k != m, k != n, k != o, k != p,
                   l != m, l != n, l != o, l != p,
                   m != n, m != o, m != p,
                   n != o, n != p,
                   o != p])]
    chromosome = np.array(chromosome)
    chromosome = pd.DataFrame(chromosome)
    return chromosome
```

In [3]:
```
initial_population = generate_population()
initial_population
```

Out[3]:

	0	1	2	3	4	5	6	7
0	1	2	3	4	5	6	7	8
1	1	2	3	4	5	6	8	7
2	1	2	3	4	5	7	6	8
3	1	2	3	4	5	7	8	6
4	1	2	3	4	5	8	6	7
...	...	...	...	...	...	...	...	...
40315	8	7	6	5	4	1	3	2
40316	8	7	6	5	4	2	1	3
40317	8	7	6	5	4	2	3	1
40318	8	7	6	5	4	3	1	2
40319	8	7	6	5	4	3	2	1

40320 rows × 8 columns

In [4]:

```
def fitness(population):
    pop_size = population.shape[0]
    x = 0
    y = 0
    b = 0
    c = 0
    Fit = []
    for k in range(pop_size):
        for i in range(8):
            c = 0
            for j in range(8):
                if(i != j):
                    x = abs(i-j)
                    y = abs(population.iloc[k][i] - population.iloc[k][j])
                    if(x == y):
                        c += 1
        b = 28-c
        Fit.append(b)
```

```
    Fitness = np.array(Fit)
    return Fitness
```

In [5]:
```
Fitness = fitness(initial_population)
Fitness
```

Out[5]:
```
array([21, 27, 23, ..., 23, 27, 21])
```

In [6]:
```
data = pd.DataFrame(initial_population)
data['Fit'] = pd.DataFrame(Fitness)
```

In [7]:
```
data_100 = data.sample(n=100)
data_100 = data_100.reset_index(drop = True)
data_100
```

Out[7]:

	0	1	2	3	4	5	6	7	Fit
0	4	1	6	8	5	7	2	3	27
1	4	1	7	2	5	3	8	6	27
2	4	6	1	5	3	2	8	7	27
3	5	8	3	1	6	2	4	7	28
4	5	8	3	4	7	1	6	2	27
...	...	...	...	...	...	...	...	...	...
95	4	2	7	8	6	1	3	5	28
96	7	4	5	1	6	3	8	2	28
97	2	3	4	6	1	8	5	7	28
98	5	2	3	8	1	7	4	6	28
99	2	5	4	6	3	7	1	8	28

100 rows × 9 columns

In [8]:
```
def selection(data):
    selected_parent = data.sample(n=5)
    selected_parent = selected_parent.sort_values("Fit", ascending=False)
```

```
    selected_parent1 = selected_parent.iloc[0]
    selected_parent2 = selected_parent.iloc[1]
    return selected_parent1[:8], selected_parent2[:8]
```

In [9]:
```
def crossover(C1, C2):
    point = np.random.randint((1,7), size=1)
    point = int(point)

    C1_1 = C1[:point]
    C1_2 = C1[point:]

    C2_1 = C2[:point]
    C2_2 = C2[point:]

    C1_tuple = (C1_1, C2_2)
    C1 = np.hstack(C1_tuple)

    C2_tuple = (C2_1, C1_2)
    C2 = np.hstack(C2_tuple)
    return C1, C2
```

In [10]:
```
def mutation(ch):
    point1 = np.random.randint(8, size=1)
    point1 = int(point1)

    point2 = np.random.randint(8, size=1)
    point2 = int(point2)

    first_ele = ch[point1]
    second_ele = ch[point2]

    ch[point1] = second_ele
    ch[point2] = first_ele

    return ch
```

In [11]:
```
Parent1 = []
Child_Gen1 = []
for i in range(25):
    Pa1, Pa2 = selection(data_100)
    Parent1.append(Pa1)
    Parent1.append(Pa2)
```

```
    Child1, Child2 = crossover(Pa1, Pa2)

    Child1 = mutation(Child1)
    Child2 = mutation(Child2)

    Child_Gen1.append(Child1)
    Child_Gen1.append(Child2)

Parent1_df = pd.DataFrame(Parent1)
Parent1_df = Parent1_df.reset_index(drop = True)

Child_Gen1 = pd.DataFrame(Child_Gen1)
Child_Gen1 = Child_Gen1.reset_index(drop = True)
```

In [12]:
```
Child_Gen1
```

Out[12]:

	0	1	2	3	4	5	6	7
0	2	3	1	6	4	8	5	7
1	1	5	6	8	2	3	4	7
2	7	2	3	8	6	5	1	4
3	7	6	5	1	4	2	3	8
4	4	6	7	2	8	1	3	5
5	1	6	7	5	8	3	2	4
6	2	8	4	6	3	7	1	5
7	7	4	5	1	2	8	3	6
8	8	5	1	3	6	7	2	4
9	6	8	4	7	2	5	3	1
10	6	5	7	1	2	4	8	3
11	1	4	6	3	7	8	5	2
12	2	3	6	4	8	1	7	5
13	6	3	4	8	7	5	2	1
14	2	5	3	1	6	4	8	7
15	2	3	4	8	7	1	6	5
16	5	6	7	1	3	8	4	2
17	5	6	3	4	7	1	8	2
18	4	5	6	8	3	1	7	2
19	3	6	2	4	7	5	8	1
20	4	6	1	3	5	7	2	8
21	1	4	8	6	3	2	5	7
22	2	7	8	6	3	5	1	4
23	4	3	6	8	7	1	2	5

	0	1	2	3	4	5	6	7
24	6	3	4	8	7	1	2	5
25	6	4	3	8	1	5	2	7
26	4	2	1	8	6	7	3	5
27	6	3	5	1	2	8	4	7
28	3	8	1	4	7	2	5	6
29	5	2	1	8	4	7	3	6
30	3	7	4	1	6	5	2	8
31	1	4	3	6	5	2	8	7
32	1	4	8	2	5	6	3	7
33	5	4	2	3	8	6	7	1
34	2	5	4	6	3	8	1	7
35	3	4	2	5	8	7	6	1
36	6	2	1	5	4	8	7	3
37	4	3	6	8	7	1	2	5
38	5	2	8	1	4	7	6	3
39	6	4	3	1	8	5	7	2
40	5	6	7	2	4	1	3	8
41	3	7	5	1	6	8	2	4
42	1	4	6	8	5	2	3	7
43	6	2	1	5	4	8	7	3
44	5	6	2	4	8	3	1	7
45	6	2	5	1	3	7	8	4
46	4	6	2	3	8	1	7	5
47	7	6	3	8	2	4	1	5

	0	1	2	3	4	5	6	7
48	2	5	8	4	3	1	6	7
49	3	6	5	8	2	4	1	7

In [13]:
```
Gen1_Fitness = fitness(Child_Gen1)
data_Gen1 = Child_Gen1
data_Gen1['Fit'] = pd.DataFrame(Gen1_Fitness)
```

In [14]:
```
data_Gen1
```

Out[14]:

	0	1	2	3	4	5	6	7	Fit
0	2	3	1	6	4	8	5	7	27
1	1	5	6	8	2	3	4	7	28
2	7	2	3	8	6	5	1	4	27
3	7	6	5	1	4	2	3	8	28
4	4	6	7	2	8	1	3	5	27
5	1	6	7	5	8	3	2	4	28
6	2	8	4	6	3	7	1	5	27
7	7	4	5	1	2	8	3	6	27
8	8	5	1	3	6	7	2	4	28
9	6	8	4	7	2	5	3	1	28
10	6	5	7	1	2	4	8	3	28
11	1	4	6	3	7	8	5	2	28
12	2	3	6	4	8	1	7	5	27
13	6	3	4	8	7	5	2	1	27
14	2	5	3	1	6	4	8	7	27
15	2	3	4	8	7	1	6	5	27
16	5	6	7	1	3	8	4	2	27
17	5	6	3	4	7	1	8	2	28
18	4	5	6	8	3	1	7	2	28
19	3	6	2	4	7	5	8	1	28
20	4	6	1	3	5	7	2	8	27
21	1	4	8	6	3	2	5	7	28
22	2	7	8	6	3	5	1	4	28
23	4	3	6	8	7	1	2	5	28

	0	1	2	3	4	5	6	7	Fit
24	6	3	4	8	7	1	2	5	28
25	6	4	3	8	1	5	2	7	27
26	4	2	1	8	6	7	3	5	27
27	6	3	5	1	2	8	4	7	28
28	3	8	1	4	7	2	5	6	26
29	5	2	1	8	4	7	3	6	27
30	3	7	4	1	6	5	2	8	28
31	1	4	3	6	5	2	8	7	27
32	1	4	8	2	5	6	3	7	28
33	5	4	2	3	8	6	7	1	28
34	2	5	4	6	3	8	1	7	28
35	3	4	2	5	8	7	6	1	27
36	6	2	1	5	4	8	7	3	28
37	4	3	6	8	7	1	2	5	28
38	5	2	8	1	4	7	6	3	27
39	6	4	3	1	8	5	7	2	28
40	5	6	7	2	4	1	3	8	28
41	3	7	5	1	6	8	2	4	28
42	1	4	6	8	5	2	3	7	28
43	6	2	1	5	4	8	7	3	28
44	5	6	2	4	8	3	1	7	27
45	6	2	5	1	3	7	8	4	28
46	4	6	2	3	8	1	7	5	27
47	7	6	3	8	2	4	1	5	27

	0	1	2	3	4	5	6	7	Fit
48	2	5	8	4	3	1	6	7	27
49	3	6	5	8	2	4	1	7	28

In [15]:
```
Parent2 = []
Child_Gen2 = []
for i in range(12):
    Pa1, Pa2 = selection(data_Gen1)
    Parent2.append(Pa1)
    Parent2.append(Pa2)

    Child1, Child2 = crossover(Pa1, Pa2)

    Child1 = mutation(Child1)
    Child2 = mutation(Child2)

    Child_Gen2.append(Child1)
    Child_Gen2.append(Child2)

Parent2_df = pd.DataFrame(Parent2)
Parent2_df = Parent2_df.reset_index(drop = True)

Child_Gen2 = pd.DataFrame(Child_Gen2)
Child_Gen2 = Child_Gen2.reset_index(drop = True)
```

In [16]:
```
Child_Gen2
```

Out[16]:

	0	1	2	3	4	5	6	7
0	1	4	6	2	5	8	3	7
1	7	2	1	5	4	8	6	3
2	7	2	1	5	4	8	6	3
3	1	4	8	2	7	6	3	5
4	6	2	3	1	5	7	8	4
5	1	4	6	3	7	8	5	2
6	6	4	5	1	3	7	8	2
7	3	4	5	1	6	8	2	7
8	1	5	6	7	2	3	4	8
9	1	6	7	4	8	3	2	5
10	4	2	5	8	6	7	3	1
11	7	4	5	6	2	8	3	1
12	6	3	1	5	2	8	4	7
13	8	5	1	4	6	7	2	3
14	2	6	3	4	7	5	8	1
15	6	4	8	7	2	5	3	1
16	7	6	5	1	4	2	3	8
17	6	5	2	1	7	4	8	3
18	3	4	2	5	8	6	7	1
19	6	8	4	3	2	5	7	1
20	5	6	7	2	4	1	3	8
21	8	4	6	3	7	1	5	2
22	7	2	3	8	6	5	1	4
23	6	3	8	1	2	5	4	7

In [17]:
```
Gen2_Fitness = fitness(Child_Gen2)
data_Gen2 = Child_Gen2
data_Gen2['Fit'] = pd.DataFrame(Gen2_Fitness)
```

In [18]:
```
data_Gen2
```

Out[18]:

	0	1	2	3	4	5	6	7	Fit
0	1	4	6	2	5	8	3	7	28
1	7	2	1	5	4	8	6	3	28
2	7	2	1	5	4	8	6	3	28
3	1	4	8	2	7	6	3	5	28
4	6	2	3	1	5	7	8	4	28
5	1	4	6	3	7	8	5	2	28
6	6	4	5	1	3	7	8	2	28
7	3	4	5	1	6	8	2	7	28
8	1	5	6	7	2	3	4	8	27
9	1	6	7	4	8	3	2	5	26
10	4	2	5	8	6	7	3	1	28
11	7	4	5	6	2	8	3	1	28
12	6	3	1	5	2	8	4	7	28
13	8	5	1	4	6	7	2	3	26
14	2	6	3	4	7	5	8	1	28
15	6	4	8	7	2	5	3	1	28
16	7	6	5	1	4	2	3	8	28
17	6	5	2	1	7	4	8	3	28
18	3	4	2	5	8	6	7	1	27
19	6	8	4	3	2	5	7	1	28
20	5	6	7	2	4	1	3	8	28
21	8	4	6	3	7	1	5	2	28
22	7	2	3	8	6	5	1	4	27
23	6	3	8	1	2	5	4	7	27

In [19]:
```
Parent3 = []
Child_Gen3 = []
for i in range(6):
    Pa1, Pa2 = selection(data_Gen2)
    Parent3.append(Pa1)
    Parent3.append(Pa2)

    Child1, Child2 = crossover(Pa1, Pa2)

    Child1 = mutation(Child1)
    Child2 = mutation(Child2)

    Child_Gen3.append(Child1)
    Child_Gen3.append(Child2)

Parent3_df = pd.DataFrame(Parent3)
Parent3_df = Parent3_df.reset_index(drop = True)

Child_Gen3 = pd.DataFrame(Child_Gen3)
Child_Gen3 = Child_Gen3.reset_index(drop = True)
```

In [20]:
```
Child_Gen3
```

Out[20]:

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

In [21]:
```
Gen3_Fitness = fitness(Child_Gen3)
data_Gen3 = Child_Gen3
data_Gen3['Fit'] = pd.DataFrame(Gen3_Fitness)
```

In [22]:
```
data_Gen3
```

Out[22]:

	0	1	2	3	4	5	6	7	Fit
0	5	4	6	2	1	8	3	7	28
1	7	2	8	5	4	1	6	3	26
2	5	6	7	2	1	4	3	8	28
3	1	6	4	2	5	8	3	7	28
4	2	6	3	7	4	5	8	1	27
5	8	6	4	3	7	1	5	2	28
6	5	6	7	1	4	2	3	8	28
7	8	4	6	3	5	1	7	2	27
8	5	4	6	3	7	1	8	2	28
9	1	4	2	6	5	8	3	7	27
10	1	4	6	2	7	8	3	5	28
11	5	6	3	4	7	2	8	1	28

In [23]:

```
Parent4 = []
Child_Gen4 = []
for i in range(4):
    Pa1, Pa2 = selection(data_Gen3)
    Parent4.append(Pa1)
    Parent4.append(Pa2)

    Child1, Child2 = crossover(Pa1, Pa2)

    Child1 = mutation(Child1)
    Child2 = mutation(Child2)

    Child_Gen4.append(Child1)
    Child_Gen4.append(Child2)

Parent4_df = pd.DataFrame(Parent4)
Parent4_df = Parent4_df.reset_index(drop = True)

Child_Gen4 = pd.DataFrame(Child_Gen4)
Child_Gen4 = Child_Gen4.reset_index(drop = True)
```

In [24]:
```
Child_Gen4
```

Out[24]:

	0	1	2	3	4	5	6	7
0	8	6	5	3	7	1	4	2
1	5	4	6	8	7	1	3	2
2	8	6	4	2	7	1	5	3
3	5	3	6	4	7	2	8	1
4	1	4	6	2	7	8	5	3
5	5	4	6	7	3	1	8	2
6	1	4	6	2	7	3	8	5
7	2	4	6	3	7	1	8	5

In [25]:
```
Gen4_Fitness = fitness(Child_Gen4)
data_Gen4 = Child_Gen4
data_Gen4['Fit'] = pd.DataFrame(Gen4_Fitness)
```

In [26]:
```
data_Gen4
```

Out[26]:

	0	1	2	3	4	5	6	7	Fit
0	8	6	5	3	7	1	4	2	28
1	5	4	6	8	7	1	3	2	27
2	8	6	4	2	7	1	5	3	27
3	5	3	6	4	7	2	8	1	27
4	1	4	6	2	7	8	5	3	28
5	5	4	6	7	3	1	8	2	28
6	1	4	6	2	7	3	8	5	27
7	2	4	6	3	7	1	8	5	28

In [30]:
```
# Assuming data_Gen1, data_Gen2, data_Gen3, and data_Gen4 are DataFrame objects

CHILD = pd.concat([data_Gen1, data_Gen2, data_Gen3, data_Gen4], ignore_index=True)
```

In [31]:
```
CHILD = CHILD.sort_values("Fit", ascending=False)
CHILD = CHILD.reset_index(drop = True)
CHILD.head(10)
```

Out[31]:

	0	1	2	3	4	5	6	7	Fit
0	2	4	6	3	7	1	8	5	28
1	5	4	2	3	8	6	7	1	28
2	6	2	1	5	4	8	7	3	28
3	4	3	6	8	7	1	2	5	28
4	5	6	7	2	1	4	3	8	28
5	6	4	3	1	8	5	7	2	28
6	5	6	7	2	4	1	3	8	28
7	3	7	5	1	6	8	2	4	28
8	1	4	6	8	5	2	3	7	28
9	6	2	1	5	4	8	7	3	28

In [ ]: