****

**Ahsanullah University of Science and Technology**

*Department of Computer Science & Engineering*

|  |  |
| --- | --- |
| Course No. | CSE 4108 |
| Course Name | Artificial Intelligence Lab |
| Assignment No. | 04 |

**Submitted To:**

Md. Siam Ansary Tonmoy Hossain

Department of CSE, AUST Department of CSE, AUST

**Submitted By:**

|  |  |
| --- | --- |
| Name | Tahiya Ahmed Chowdhury |
| ID No. | 17.02.04.048 |
| Session | Fall – 2020 |
| Section | A (A2) |

**Date of Submission:** August 23, 2021

**Implement Genetic Algorithm for the 8 Queens Problem using Python. For Mutation step, use Swap Mutation.**

**Python code:**

**import** numpy **as** np

**import** random

inputSet **=** **[]**

**def** randomNumGenerator**():**

**for** i **in** **range(**0**,** 15**):**

a **=** **[]**

**for** j **in** **range(**0**,** 8**):**

a**.**append**(**random**.**randrange**(**0**,**7**));**

inputSet**.**append**(**a**)**

**def** prepareForfitnessCalculation**():**

**for** i **in** **range(**0**,**15**):**

b **=** **[]**

**for** j **in** **range(**0**,**8**):**

tempLst **=** **[]**

**for** k **in** **range(**0**,**1**):**

tempLst**.**append**(**inputSet**[**i**][**j**])**

tempLst**.**append**(**j**)**

b**.**append**(**tempLst**)**

fitnessCalculation**(sorted(**b**),**i**)**

**def** fitnessCalculation**(**positionList**,**k**):**

h **=** 0

**for** i **in** **range(**0**,**7**):**

**for** j **in** **range(**i**+**1**,**7**):**

t1 **=** **abs(**positionList**[**i**][**0**]** **-** positionList**[**j**][**0**])**

t2 **=** **abs(**positionList**[**i**][**1**]** **-** positionList**[**j**][**1**])**

**if** positionList**[**i**][**0**]** **==** positionList**[**j**][**0**]:**

h **=** h**+**1

**elif** t1 **==** t2**:**

h **=** h**+**1

thisdict**[**k**]** **=** h

**def** getFirstTwoKeys**(**dictionary**):**

n **=** 0

Index **=** **[]**

**for** state **in** dictionary**:**

Index**.**append**(**state**)**

n **=** n**+**1

**if** n **==** 2**:**

**break**

**return** Index

**def** crossover**():**

n **=** 0**;** IndexAsc **=** **[];** IndexDes **=** **[]**

Parent1 **=** **[];** Parent2 **=** **[]**

Child1 **=** **[];** Child2 **=** **[]**

IndexAsc **=** getFirstTwoKeys**(**sort\_dict**)**

IndexDes **=** getFirstTwoKeys**(**sort\_dict\_rev**)**

**for** i **in** **range(**0**,**15**):**

**if(**i **==** IndexAsc**[**0**]):**

**for** j **in** **range(**0**,**8**):**

Parent1**.**append**(**inputSet**[**i**][**j**])**

**elif(**i **==** IndexAsc**[**1**]):**

**for** j **in** **range(**0**,**8**):**

Parent2**.**append**(**inputSet**[**i**][**j**])**

**for** j **in** **range(**0**,**4**):**

Child1**.**append**(**Parent1**[**j**])**

Child2**.**append**(**Parent2**[**j**])**

**for** j **in** **range(**4**,**8**):**

Child1**.**append**(**Parent2**[**j**])**

Child2**.**append**(**Parent1**[**j**])**

m **=** random**.**randrange**(**0**,**7**)**

n **=** random**.**randrange**(**0**,**7**)**

Child1**[**m**],** Child1**[**n**]** **=** Child1**[**n**],** Child1**[**m**]**

Child2**[**m**],** Child2**[**n**]** **=** Child2**[**n**],** Child2**[**m**]**

**for** i **in** **range(**0**,**15**):**

**if(**i **==** IndexDes**[**0**]):**

**for** j **in** **range(**0**,**8**):**

inputSet**[**i**][**j**]** **=** Child1**[**j**]**

**elif(**i **==** IndexDes**[**1**]):**

**for** j **in** **range(**0**,**8**):**

inputSet**[**i**][**j**]** **=** Child2**[**j**]**

**print(**"\nThe children after crossover:"**)**

**print(**Child1**)**

**print(**Child2**)**

**print(**"\nThe input set after rearranging:"**)**

**print(**inputSet**)**

randomNumGenerator**()**

**for** d **in** **range(**0**,**100**):**

thisdict **=** **{}**

prepareForfitnessCalculation**()**

**print(**"\nThe input set at first:"**)**

**print(**inputSet**)**

**print(**"\nThe entry numbers vs fitness scores:"**)**

**print(**thisdict**)**

sort\_dict **=** **dict(sorted(**thisdict**.**items**(),** key**=lambda** item**:** item**[**1**]))**

sort\_dict\_rev **=** **dict(sorted(**thisdict**.**items**(),** key**=lambda** item**:** item**[**1**],**reverse**=True))**

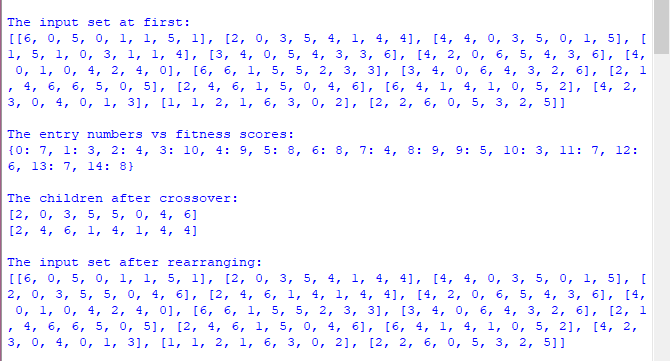
crossover**()**

**Analyzing code:**

Here the list inputSet is a nested list which contains 8 columns in 15 rows. In each row, the queen positions are denoted in their colums. Thus we have total 15 position sets. The function randomNumGenerator assigns values from the range 0 to 7 i.e the queen positions to the inputSet. Then the inputSet is modified row by row by storing each rows in a tempList to pass it to the fitnessCalculation function. This was fitness of each set is determined and stored in the dictionary values of thisdict with row numbers as the keys. The dictionary is then sorted into ascending and descending order and stored into sort\_dict and sort\_dict\_reverse respectively. The getFirstTwoKeys function is used to get Parent1 and Parent2, the entries with lowest fitness score and create Child1 and Child2 by passing sort\_dict. Finally, using the same function, the entries with maximum fitness score is selected using sort\_dict\_reverse and replaced with Child1 and Child2. Thus a modified inputSet is obtained.

This procedure runs in a loop of 100 times. And each time the fitness score is calculated.

**Python output:**

****

**Analyzing output:**

Initially the inputSet and the fitness score for each rows i.e. entries are shown. From the fitness score of above output we can see that entry 01 and 10 has the minimum fitness sores. Thus they are selected as the Parent1 and Parent2 and then Child1 and Child2 is formed from them. Then swap mutation is performed and finally entry 03 and 04, the entries with maximum fitness, are replaced by Child1 and Child2. Thus a modified inputSet is obtained which maybe used for next iteration.