Code Explanation

The codes start by taking necessary parameters. These parameters are entry *number*, *iteration* *number*, *Generation* *Size* and *p-size.* *Entry* *Number* represent how many times the code will start all over again and create a randomly determined first Generation. *Iteration* *Number* stands for how many time the code will create new generation. *Generation* *Size* will determine how many creature will be created in each generation. If this number is too few than it will create a problem since the population will not be able sustain itself. *P* value is given parameter which determines how many True value will there be in an array of total number of cities.

When starting a generation, a randomly created Generation will appear. This generation will be composed of individuals or creatures which all the creatures are a feasible solution. An individual will be constructed by assigning a random number to each of its node. If that value is smaller than the value of p/*size* it will be have a True value. *p*/*size* is because in the long run we expect p number of nodes to be true for *size* many nodes. Generating a random number for each node might seem absurd but this is to conserve the uniformity of the individual. Distortion of uniformity is caused by the random number that is drawn using the programming language. With the way, we do it this uniformity is relatively conserved. We will check every creature’s feasibility and if the creature is not feasible we will make an abortion. This process will be repeated for every individual in a generation. At the end, first initial generation will be created. This first generation will be presented in a sorted way.

After this part, iterative part will start with breeding new individuals and by killing not fit ones. An individual will be unfit by these conditions.

**fitness(bestfit) + generationSpan \* 0.2 < fitness(i)**

**60% of the individuals will die randomly**

**fitness(bestfit) + generationSpan \* 0.7 < fitness(i)**

**all of them will die.**

We will place a new breed to every dead once place. By using this breading algorithm.

New Breed

Father

Mother

When new breed is created, it is first tested if it is a feasible individual. If not we make an abortion. If feasible then we control if there is a mutation. There are four types of mutation category 0,1,2 and 3. Type of mutation is also the number open nodes will be closed and number of closed nodes that will be opened. If the child survives all this it will be place if no same configuration is already existing in the next generation. This control sort of acts like a taboo list. It will also prevent the convergence of the generation to a single individual.

Best answer and its configuration will be recorded to Memory Sheet it will also be recorded to Model in addition to the best solutions at every generation. Paragraph 2,3,4 & 5 will be repeated by the Entry Number times.