Solution 1)

a. In this case, we want to find the number of ways to choose 2 cities from the set of 10 cities to create a link (gene). So the formula would be:

   C(10, 2) = 45

   So, there will be 45 genes used in the chromosome of each individual.

b. 45

Solution 2)

a)

- For x1 = 6 5 4 1 3 5 3 2:
  f(x1) = (6 + 5) - (4 + 1) + (3 + 5) - (3 + 2) = 11 - 5 + 8 - 5 = 9
- For x2 = 8 7 1 2 6 6 0 1:
  f(x2) = (8 + 7) - (1 + 2) + (6 + 6) - (0 + 1) = 15 - 3 + 12 - 1 = 23
- For x3 = 2 3 9 2 1 2 8 5:
  f(x3) = (2 + 3) - (9 + 2) + (1 + 2) - (8 + 5) = 5 - 11 + 3 - 13 = -16
- For x4 = 4 1 8 5 2 0 9 4:
  f(x4) = (4 + 1) - (8 + 5) + (2 + 0) - (9 + 4) = 5 - 13 + 2 - 13 = -19

Arranging the individuals in order from fittest to least fit:

x2 = 8 7 1 2 6 6 0 1 (fitness: 23)
x1 = 6 5 4 1 3 5 3 2 (fitness: 9)
x3 = 2 3 9 2 1 2 8 5 (fitness: -16)
x4 = 4 1 8 5 2 0 9 4 (fitness: -19)

b)

i) Cross the fittest two individuals using a one-point crossover at the middle point:

x2 = 8 7 1 2 | 6 6 0 1
x1 = 6 5 4 1 | 3 5 3 2

The offspring after one-point crossover:

Offspring 1: 8 7 1 2 | 3 5 3 2
Offspring 2: 6 5 4 1 | 6 6 0 1

ii) Cross the second and third fittest individuals using a two-point crossover (points b and f):

x1 = 6 5 | 4 1 3 | 5 3 2
x3 = 2 3 | 9 2 1 | 2 8 5

The offspring after two-point crossover:

Offspring 3: 6 5 | 9 2 1 | 5 3 2
Offspring 4: 2 3 | 4 1 3 | 2 8 5

iii) Cross the first and third fittest individuals (ranked 1st and 3rd) using a uniform crossover:

x2 = 8 7 1 2 6 6 0 1
x3 = 2 3 9 2 1 2 8 5

The offspring after uniform crossover:
Offspring 5: 8 7 9 2 1 2 0 1
Offspring 6: 2 3 1 2 6 6 8 5

c)

To evaluate the fitness of the new population, we need to calculate the fitness values for each offspring using the given fitness function.
Fitness calculations for the offspring:

f(Offspring 1) = (8 + 7) - (1 + 2) + (3 + 5) - (3 + 2) = 18 - 3 + 8 - 5 = 18
f(Offspring 2) = (6 + 5) - (4 + 1) + (6 + 6) - (0 + 1) = 11 - 5 + 12 - 1 = 17
f(Offspring 3) = (6 + 5) - (9 + 2) + (1 + 2) - (5 + 3) = 11 - 11 + 3 - 8 = -5
f(Offspring 4) = (2 + 3) - (4 + 1) + (3 + 1) - (2 + 8) = 5 - 5 + 4 - 10 = -6
f(Offspring 5) = (8 + 7) - (9 + 2) + (1 + 2) - (0 + 1) = 15 - 11 + 3 - 1 = 6
f(Offspring 6) = (2 + 3) - (1 + 2) + (6 + 6) - (8 + 5) = 5 - 3 + 12 - 13 = 1

The fitness values for the offspring are:

f(Offspring 1) = 18
f(Offspring 2) = 17
f(Offspring 3) = -5
f(Offspring 4) = -6
f(Offspring 5) = 6
f(Offspring 6) = 1

Comparing the fitness of the offspring with the initial population, we find that the overall fitness has improved.
**The fittest offspring (Offspring 1) has a higher fitness value than any individual in the initial population.

d)

To find the chromosome representing the optimal solution with the maximum fitness, we need to consider the fitness function and the range of values for each gene (digit between 0 and 9).
Given the fitness function: f(x) = (a + b) - (c + d) + (e + f) - (g + h)
We need to maximize the positive terms (a + b + e + f) and minimize the negative terms (c + d + g + h) to maximize the fitness.
Considering the range of values for each gene (digit between 0 and 9), the optimal solution would have the following:

- The highest possible values for a, b, e, and f (each equal to 9).
- The lowest possible values for c, d, g, and h (each equal to 0).

The chromosome representing the optimal solution would be: 9 9 0 0 9 9 0 0
The value of the maximum fitness would be:

f(max) = (9 + 9) - (0 + 0) + (9 + 9) - (0 + 0) = 36

e)

Looking at the initial population, we cannot determine whether it will be able to reach the optimal solution without the mutation operator. The performance and convergence of a genetic algorithm depend on various factors such as the population size, selection criteria, crossover operators, mutation operators, etc. . It's possible that the initial population may contain individuals that are relatively close to the optimal solution, but the convergence to the optimal solution will depend on the specific dynamics and parameters of the genetic algorithm.

Solution 3)

a)

A possible chromosome representation for an individual in this Genetic Algorithm could be a sequence of assignments of cabin crews to the three planes on a particular day.

For example, if we represent each cabin crew by a unique number (1 to 5), a chromosome could look like this:

Chromosome: 3 1 5 2 4

“This chromosome represents the assignment of crews to the three planes on a particular day, where crew 3 is assigned to plane 1, crew 1 to plane 2, crew 5 to plane 3, crew 2 to plane 1 (the next day), and crew 4 to plane 2 (the day after).”

b)

The alphabet of this algorithm would consist of the numbers 1 to 5, representing the cabin crews. The size of the alphabet is 5 since there are 5 different crews to choose from.

c)

Suppose we have the following chromosome representing an individual's assignment of crews to planes for a particular day:

Chromosome: 3 1 5 2 4

To calculate the fitness of this individual, we can use the suggested fitness function:

Fitness(x) = (Number of different crews used) - (Penalty for consecutive workdays)

Number of different crews used: In this chromosome, we have crews 3, 1, 5, 2, and 4 assigned to the planes. So the number of different crews used is 5.

Penalty for consecutive workdays: To determine the penalty for consecutive workdays, we need to check if any crew has worked for more than two consecutive days.

Looking at the chromosome, we see that crew 3 worked on the previous day (plane 1) and is assigned again on the current day (plane 2).

DAY:	1	2	3	4	5	6	7
CHR:	315	243	152	431	524	315	243
PLN:	123	123	123	123	123	123	123

As visible, each day crew assigned to plane 1 is assigned to plane 3 the next day.

This indicates a violation of the consecutive workday constraint. Therefore, a penalty needs to be applied.
Assuming the penalty for each violation is -10, the penalty for this chromosome would be -10.
Now we can calculate the fitness of this individual:
Fitness(x) = (Number of different crews used) - (Penalty for consecutive workdays)
= 5 - 10 = -5
So, the fitness of the individual with the chromosome 3 1 5 2 4 is -5 based on the given fitness function.

Programming Assignment

```
import heapq

def min_cost_of_joining_ropes(ropes):
    heap = []
    for rope in ropes:
        heapq.heappush(heap, rope)

    total_cost = 0

    while len(heap) > 1:
        rope1 = heapq.heappop(heap)
        rope2 = heapq.heappop(heap)

        cost = rope1 + rope2

        total_cost += cost

        heapq.heappush(heap, cost)

    return total_cost
```

```python
ropes = [4, 3, 2, 6]
minimum_cost = min_cost_of_joining_ropes(ropes)
print("Minimum cost of joining all ropes:",
minimum_cost)
```

```
```