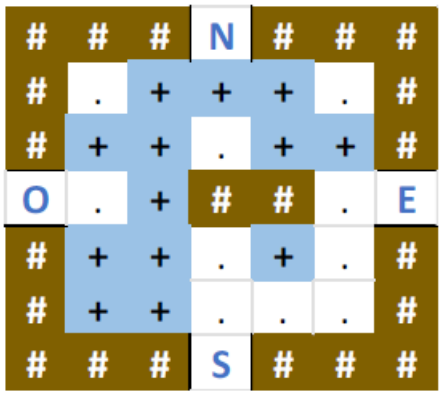
Consider the urban planning problem of a medieval walled city.

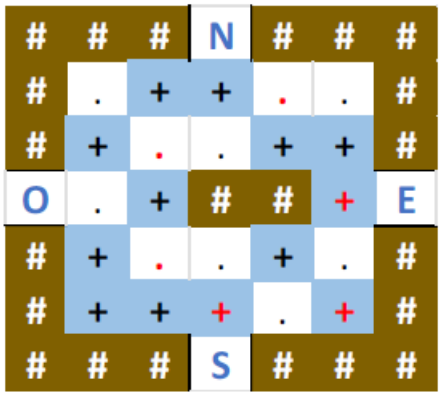


The houses in brown with a cardinal are walls or areas where there can be no construction or passage, such as cliffs or lakes. There are four gates in the wall, one for each cardinal point. The houses in white with a period are common circulation areas. The houses in blue with a plus sign are residential areas. It is possible to move from one area to any adjacent area, including diagonals. However, not all gates are accessible to all residential areas, using only common areas for circulation. For example, the north gate can only be accessed by three residential areas. All other residential areas, if they want to use the north gate, have to pass through another residential area.

Tired of dealing with problems arising from people having to pass through each other's houses to circulate, the king intends to impose a shocking decree: each residential area must have access to all four city gates!

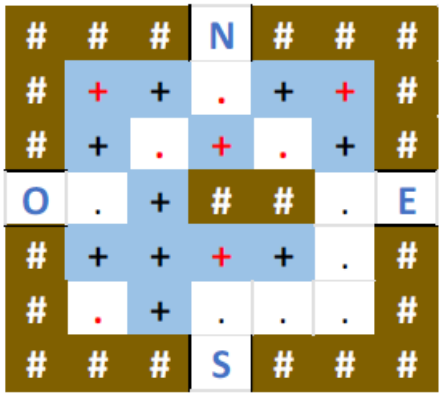
In order to satisfy the king's whim, residential areas must be moved to common circulation areas. The solution should solve the problem and ideally have the fewest number of movements.

In the problem above, a possible solution with three movements would be:



Note that only 3 residential areas have changed location. The new and old locations are indicated in red.

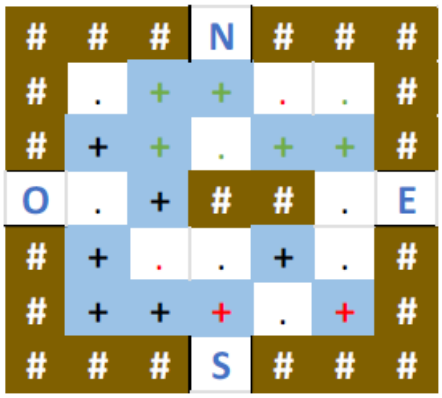
The goal is to minimize the cost, specified by the number of movements. Each movement has a unit cost. It doesn't matter in this problem if some residential areas have a long route to reach certain gates. Another solution, but with more movements, would be:



This solution has a cost of 4, while the previous one has a cost of 3, so the first solution is preferable considering the goal of reducing the number of movements.

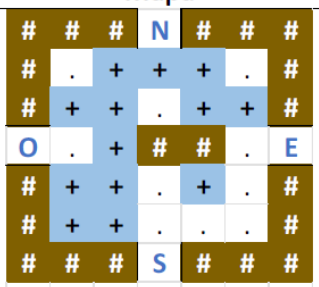
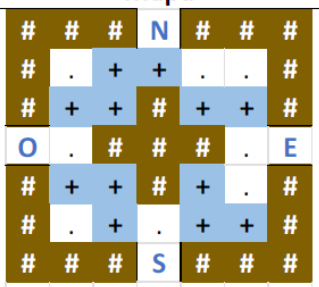
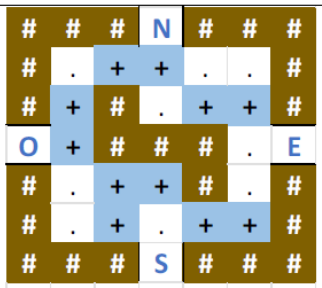
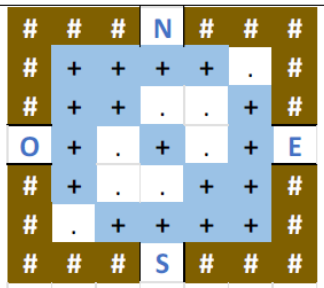
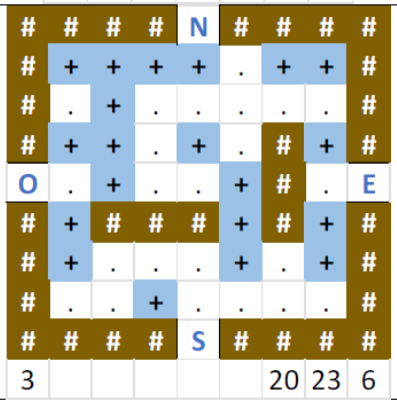
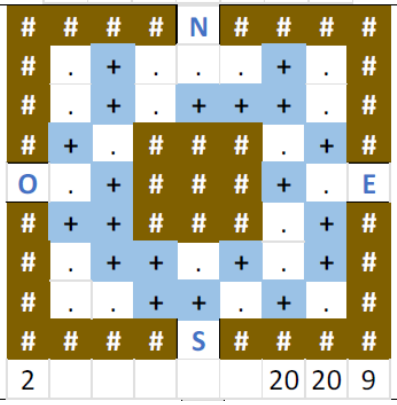
Considering that sometimes access to all 4 gates may be impossible, the king agreed to identify another criterion in such cases. In that case, the objective is to minimize the number of residential areas that do not have access to the most inaccessible gate. For each of these residential areas, the cost will be 100 units.

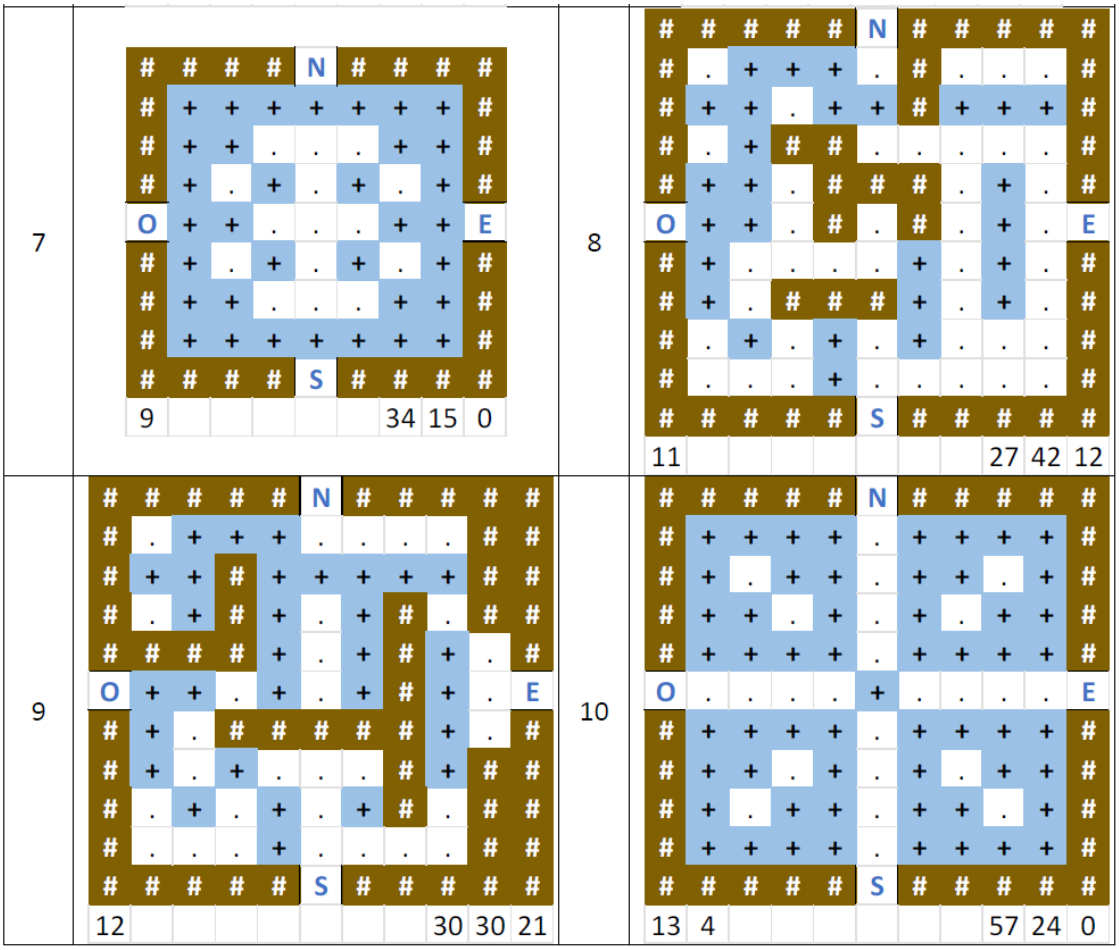
In the initial example, doing nothing would mean keeping the north gate with only 3 residential areas having access. Therefore, 10 areas would not have access, resulting in a cost of not doing anything equal to 100 * 10 = 1000 units. To clarify this point, let's consider the following solution:



In this case, two movements were made, so we have a cost of 2. However, we only have 5 residential areas, marked in green, with access to the north gate. Thus, we have 8 residential areas without access, resulting in a cost of the solution of 8 * 100 + 2 = 802.

Consider the following instances:

ID	Mapa	ID	Mapa
1	 3 13 10 2	2	 4 12 8 5
3	 4 11 9 5	4	 4 17 8 0
5	 3 20 23 6	6	 2 20 20 9



Below each map, there is additional information. The three numbers on the right represent the number of residential areas, the number of circulation areas, and the number of unusable areas. The number on the left is the best cost found by the reference resolution. For the last instance, there is a second number, which is the lowest cost found manually.

The maps of the instances can be defined as matrices with a maximum size of 10x10 and initialized statically in the code.

```
// 5x5
{{0,1,1,1,0,},{1,1,0,1,1,},{0,1,-1,-1,0,},{1,1,0,1,0,},{1,1,0,0,0,},},
{{0,1,1,0,0,},{1,1,-1,1,1,},{0,-1,-1,-1,0,},{1,1,-1,1,0,},{0,1,0,1,1,},},
{{0,1,1,0,0,},{1,-1,0,1,1,},{1,-1,-1,-1,0,},{0,1,1,-1,0,},{0,1,0,1,1,},},
{{1,1,1,1,0,},{1,1,0,0,1,},{1,0,1,0,1,},{1,0,0,1,1,},{0,1,1,1,1,},},
// 7x7
{{1,1,1,1,0,1,1,},{0,1,0,0,0,0,0,},{1,1,0,1,0,-1,1,},{0,1,0,0,1,-1,0,},{1,-1,-
1,-1,1,-1,1,},{1,0,0,0,1,0,1,},{0,0,1,0,0,0,0,},},
{{0,1,0,0,0,1,0,},{0,1,0,1,1,1,0,},{1,0,-1,-1,-1,0,1,},{0,1,-1,-1,-
1,1,0,},{1,1,-1,-1,-1,0,1,},{0,1,1,0,1,0,1,},{0,0,1,1,0,1,0,},},
{{1,1,1,1,1,1,1,},{1,1,0,0,0,1,1,},{1,0,1,0,1,0,1,},{1,1,0,0,0,1,1,},{1,0,1,0,
1,0,1,},{1,1,0,0,0,1,1,},{1,1,1,1,1,1,1,},},
// 9x9
{{0,1,1,1,0,-1,0,0,0,},{1,1,0,1,1,-1,1,1,1,},{0,1,-1,-1,0,0,0,0,0,},{1,1,0,-
1,-1,-1,0,1,0,},{1,1,0,-1,0,-1,0,1,0,},{1,0,0,0,0,1,0,1,0,},{1,0,-1,-1,-
1,1,0,1,0,},{0,1,0,1,0,1,0,0,0,},{0,0,0,1,0,0,0,0,0,},},
{{0,1,1,1,0,0,0,0,-1,},{1,1,-1,1,1,1,1,1,-1,},{0,1,-1,1,0,1,-1,0,-1,},{-1,-1,-
1,1,0,1,-1,1,0,},{1,1,0,1,0,1,-1,1,0,},{1,0,-1,-1,-1,-1,-
1,1,0,},{1,0,1,0,0,0,-1,1,-1,},{0,1,0,1,0,1,-1,0,-1,},{0,0,0,1,0,0,0,0,-1,},},
{{1,1,1,1,0,1,1,1,1,},{1,0,1,1,0,1,1,0,1,},{1,1,0,1,0,1,0,1,1,},{1,1,1,1,0,1,1
,1,1,},{0,0,0,0,1,0,0,0,0,},{1,1,1,1,0,1,1,1,1,},{1,1,0,1,0,1,0,1,1,},{1,0,1,1
,0,1,1,0,1,},{1,1,1,1,0,1,1,1,1,},},
}

// dimensions { 5, 5, 5, 5, 7, 7, 7, 9, 9, 9 }
```

To solve the problem, you should use informed search algorithms.

To solve the problem, you need to deliver the following:

- Report

- Source code of the implemented algorithms

The report should include a table with the results of the execution of the tested algorithms/configurations versus the provided instances. For each algorithm/instance, the following information should be displayed:

- Number of evaluations (maximum of 100,000 evaluations for improvement searches and 100,000 evaluations for constructive searches)

- Number of generations

- Cost

- Time spent (maximum of 1 minute)

For each instance, the best information obtained considering all executions should be provided, specifically the value of the best solution. This solution should be presented in an appendix, using a format similar to the one used in this prompt.

You should establish stopping criteria for both the number of evaluations performed and the time spent to ensure that the runs have a maximum of 100,000 evaluations (for improvement and constructive searches) and a maximum of 1 minute (whichever occurs first). It is understood that you do not need to check the stopping criteria at each moment, so if one of these limits is slightly exceeded, it is not a problem.