Noise Reduction

Objective
Give practice with recursion in C.

Story
The animals have been making too much noise, and guests are complaining. Some animals antagonize nearby animals and cause them to emit screams of rage or cries of annoyance. You have experimented with different animals next to each other, and you feel that you now have enough information to move forward with your plans.

The animals will be in a line of exhibits. Each animal will be in its own exhibit, and there are an equal number of animals as exhibits. You have found that animals only antagonize the animals in exhibits within **2 exhibits** of them. Animals that are antagonized will generate a specific level of noise. The total noise of the park is the sum of all the levels of noises generated.

To alleviate some stress from the animals and pain from the guests ear's you will attempt to find some animal ordering that is a few decibels lower.

Problem
Given a list of noises generated by all pairs of animals, determine the least sum of noise that can be generated.

Input
Input will begin with a line containing 1 integer, ***n*** ($1 \le n \le 11$), representing the number of animal exhibits. The following ***n*** lines will each contain n space separated non-negative integers, representing the noises generated by animal pairs. The ***i***-th value on the ***j***-th line represents the noise level generated by animal ***j*** when antagonized by animal ***i***. Each of the given individual noise levels generated will be at most 1,000,000. The ***i***-th value of the ***i***-th line will always be 0. Animals don't generate noise from themselves.

Output
Output should contain 1 line of output, containing a single integer representing the least possible noise level sum.

Sample Input	**Sample Output**

`4`
`0 1 2 3`
`4 0 5 6`
`7 8 0 9`
`8 7 6 0`
	`51`

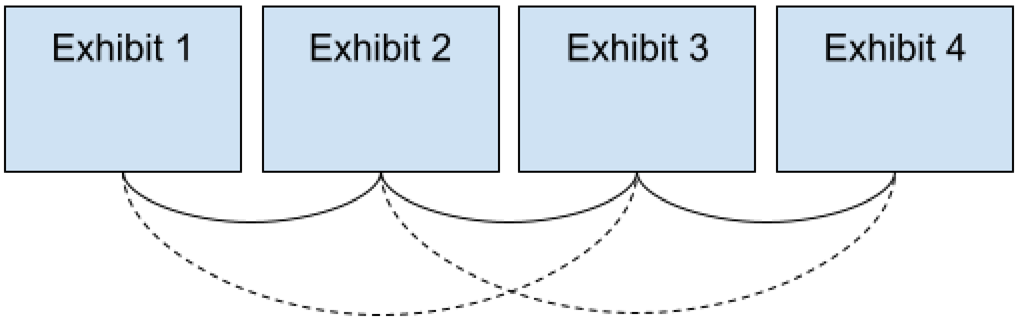
5	47
0 2 9 1 1	
2 0 9 2 2	
9 9 0 9 2	
1 2 9 0 1	
5 5 5 5 0	

Explanation

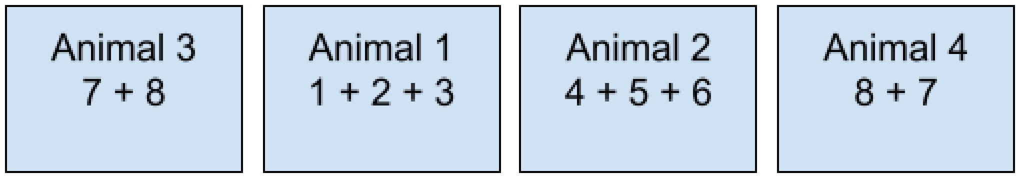
Case 1

There are 4 animals in this case. If we consider the animals number 1 through 4, then an optimal arrangement to create the lowest noise sum is 3, 1, 2, and 4. **Note there are other optimal arrangements.**

Animals in the different cages generate noise due to animals that are within 2 exhibits of themselves. In this case we have 4 exhibit. Below is a picture that shows how exhibits relate to each other in terms of noise generation. Solid lines represent exhibits within a distance of 1, and dashed lines represent animals within a distance of 2.



Note that Exhibit 2 and 3 reach all the other exhibits in this case. The picture below demonstrates how much noise is generated by each animal with the given optimal arrangement.



Animal 3 generates 7 noise (because of animal 1 on the right) and 8 noise (because of animal 2 on the right's right). **Total of 15.**
Animal 1 generates 2 noise (because of animal 3 on the left), 1 noise (because of animal 2 on the right), and 3 noise (because of animal 4 on the right's right). **Total of 6.**
Animal 2 generates 5 noise (because of animal 3 on the left's left), 4 noise (because of animal 1 on the left), and 6 noise (because of animal 4 on the right). **Total of 15.**
Animal 4 generates 8 noise (because of animal 1 on the left's left) and 7 noise (because of animal 2 on the left). **Total of 15.**

The grand total is 51.

Case 2
Given the animals are numbered 1 through 5 inclusive, an optimal arrangement is 2, 1, 4, 5, and 3.

Animal 2 generates 2 + 2 noise (4 total)
Animal 1 generates 2 + 1 + 1 noise (4 total)
Animal 4 generates 1 + 2 + 9 + 1 noise (13 total)
Animal 5 generates 5 + 5 + 5 noise (15 total)
Animal 3 generates 9 + 2 noise (11 total)

Grand total is 47.

Hints

Noise Grid Storage: You should store the noises in a 2D array.

Global Variables: You can use global variables to make argument passing more reasonable.

Permutations: You should try permuting all the animals. When you have a permutation you can check the amount of noise generated by each animal using a loop. (You could make this into a function).

Out of Bounds: Be careful not to index out of bound when computing how much noise is generated.

Answer Storage: You could either return the least noise generated through your permutation function, or you could use a global variable to hold onto this value.

Minimums: You want to store the minimum noise generated.

Large Answers: If you are going to create some large “impossible” value to initialize your answer make sure you use a large enough value. Although the noise generated by a single pair can be a million, the total noise generated could be much larger.

Grading Criteria

- Good comments, whitespace, and variable names
    - 15 points
- No extra input output (e.g. input prompts, “Please enter the number of friends:”)
    - 10 points
- Prevent reusing an animal multiple times in your arrangement
    - 10 points
- Create some way to find the total noise generated (either store the result as you go or loop when arrangement is finished).
    - 5 points
- After using an animal for some a possible arrangement make sure to undo any changes to allow for reusing the animal in the future.
    - 10 points
- Programs will be tested on <u>10 cases</u>
    - 5 points <u>each</u>

No points will be awarded to programs that do not compile using “gcc -std=gnu11 -lm”.

Sometime a requested technique will be given, and solutions without the requested technique will have their maximum points total reduced. For this problem use recursion. ***<u>Without this programs will earn at most 50 points!</u>***

Any case that causes a program to return a non-zero return code will be treated as wrong. Additionally, any case that takes longer than the maximum allowed time (the max of {5 times my solutions time, 10 seconds}) will also be treated as wrong.

<u>No partial credit will be awarded for an incorrect case.</u>

Challenge: If you want to give yourself a challenge, try writing a program that outputs an optimal order of animals.