Problem 1. Renewal

In Bulgaria there are road problems – too many holes! This is why in the parliament they decided it was about time to renew most of the roads (and get some money below the table, of course).

In Bulgaria there are **N** cities. Between some of the cities there are bidirectional roads. We can say that a route between two cities exists when we can start from the first city and end in the second city, no matter the cities in between we must go through.

Current situation in the country is not very good. Some cities do not have roads between them, others have more than one, which makes the traffic control difficult. Our parliament wants to build new roads and destroy others in such way that after the whole reconstruction there will be **exactly one route between each pair of distinct cities.** Since building and destroying new roads is expensive, the parliament wants to minimize the expenses for the new road network as much as possible.

You will be given a matrix of numbers **M** with size **NxN** representing the current state of the roads. There is a bidirectional road between city **i** and city **j** only if **M[i][j]** equals **1** and **M[j][i]** equals **1**. You will also be given matrices **B** and **D** also with size **NxN**. If between cities **i** and **j** there is no road, then the cost for its building will be **B[i][j]** and if there is a road, then the cost for its destroying will be **D[i][j]**. Prices in **B** and **D** are written as symbols where capital English letters **A, B, …, Z** represent the values **0, 1, …, 25** accordingly and the lower case letters **a, b, …, z** represent the values **26, 27, …, 51**. Your task is to write down a program, which by given road state calculates the minimum available sum needed for the reconstruction.

Input

The input data should be read from the console.

The input will contains exactly **3 * N + 1** lines.

On first input line there will be the number **N**.

On the next N lines, there will be the matrix **M**.

On the next N lines, there will be the matrix **B**.

On the next N lines, there will be the matrix **D**.

The input data will always be valid and in the format described. There is no need to check it explicitly.

Output

The output data should be printed on the console.

On the first and only output line, print the minimum sum needed for the reconstruction.

Constraints

- **N** will be between **1** and **50**, inclusive.
- **M**, **B** and **D** will be **NxN** matrices.
- **M** will contain only **0** and **1**.
- **B** and **D** will contain only lower and upper case English letters.
- For each **i** and **j**, **M[i][j]**, **B[i][j]**, **D[i][j]** will equal **M[j][i]**, **B[j][i]**, **D[j][i]** accordingly.
- There will be no road, which starts and ends in the same city.

- Values in **B**, which correspond to existing road, can be ignored.
- Values in **D**, which correspond to non-existing roads, can be ignored.
- Allowed working time for your program: **0.1** seconds. Allowed memory: **16 MB**.

Examples

Input	Output	Explanations
3 000 000 000 ABD BAC DCA ABD BAC DCA	3	We have three cities, without any roads. We build the roads 0-1 and 1-2 – final result 3.
3 011 101 110 ABD BAC DCA ABD BAC DCA	1	All cities are connected to each other. We need to destroy one of the roads. The cheapest solution is 0-1.
6 011000 101000 110000 000011 000101 000110 ABDFFF BACFFF DCAFFF FFFABD FFFBAC FFFDCA ABDFFF BACFFF DCAFFF FFFABD FFFBAC FFFDCA	7	-