Task 1: The odds are even in this war of mine

The struggle between the odd and even soldiers has never been so fierce! Each battle will see even and odd soldiers thrown in together, all of them integers. The odd soldiers in the battlefield will fight using the `1` bits from their binary representation, while the even soldiers will fight using their `0` bits. If present in the battle, soldiers with the number `0` will be neutral, hence they will not be fighting for either side. But beware - there are also spies and saboteurs - each soldier that wears negative sign can have a negative effect on the battle's result.

You should return:

- `odds win` - if the number of `1`s from odd soldiers is larger than the number of `0`s from even soldiers
- `evens win` - if the number of `1`s from odd soldiers is smaller than the number of `0`s from even soldiers
- `tie` - if there is an equal number of `1`s and `0`s or when the battlefield is completely empty

Please note that any prefix that might appear in the binary representation, e.g. `0b`, should not be counted towards the battle.

Example:

Calculation for a battlefield configuration: `[5, 3, 14]`:

- odds: `5` and `3` => `101` and `11` => four `1`s
- evens: `14` => `1110` => one `0`
- result: `odds win` the battle by a score of 4-1

Examples with negative integers:

```
battle([21,-3,20]) => "evens win"     // 10101-11 vs 10100,        3-2 vs 3
battle([7,-3,-14,6]) => "evens win"   //   111-11 vs -1110+110,    3-2 vs -1+1
battle([17,-3, 32, -24]) => "tie"     // 10111-11 vs 100000-11000, 4-2 vs 5-3
```

Task 2: Minimum effort

You are moving through difficult terrain. Your goal is to move from the top left corner of the map to the bottom right, without losing too much energy. There's one more constraint: you can only move to the right and down. Each field you will move through will have a number assigned - this is the cost of travelling through that field. Save your energy and win the competition!

How to solve this problem:
Your program should accept as its first argument a path to a filename, which contains the description of one or more terrain. The first line will have the value of `n` (the size of the square matrix). This will be followed by `n` rows of the matrix (integer costs in these rows will be comma delimited). After the `n` rows, the pattern repeats. For each terrain definition, please create a separate line where you will output the minimum effort required to travel through the terrain.

Example input:

```
2
4,6
2,8
3
1,2,3
4,5,6
7,8,9
```

Example output:

```
14
21
```

Good luck!