Zombie Apocalypse v4.6.1

========== *2021 July* ==========

Instructions

- Read the problem description thoroughly then create a program to solve it.
- For the solution, we prefer you to use JavaScript/TypeScript, Java or Python.
- There is no requirement for visualising this in any way beyond the output specified below.
- Your application must run, and generate the correct result.
- Provide a README with instructions to test and run the program. Note any key design thoughts or assumptions you've made during development.
- While this is a small problem, we expect you to submit what you believe is **production quality code**. That is code that you would be happy to test, run, maintain, and evolve.
- To submit your code: either compress your files into a single Zip or GZip archive and send via email, or store your code in a private repository and give us access. Please **do not** publish the code in public repositories.
- As a general rule, we allow **3 days** from the date that you receive these instructions to submit your code, but you may request more time if needed.

This exercise should take you around 2 to 8 hours to complete, depending on your approach, experience and how much time you have to spare.

Through this exercise, we assess a number of things including the design of your solution and your programming skills. We will review the code and may offer you an interview in which we will discuss the design and code decisions you made.

We are looking for talented individuals with extraordinary ideas and problem solving skills. In your README, we would like you to describe at least one highlight of your solution that you think is great!

Problem Description

After the nuclear war, a strange and deadly virus has infected the planet producing mindless zombies. These zombies now wander the world converting any remaining living creatures they find to zombies as well.

The world is represented by an *n* x *n* grid on which **zombies** and **creatures** live.
The location of zombies and creatures can be addressed using zero-indexed *x-y* coordinates. The top left corner of the world is `(x: 0, y: 0)`. The horizontal coordinate is represented by *x*, and the vertical coordinate is represented by *y*.

At the beginning of the program, a single zombie awakes and begins to move around the grid following a sequence of movements. Valid movements are **U**p, **D**own, **L**eft, **R**ight. The movement sequence is represented by a string of single character movements, e.g. `RDRU` (**R**ight, **D**own, **R**ight, **U**p).
Zombies can move through the edge of the grid, appearing on the directly opposite side. For a 10x10 grid, a zombie moving left from (0, 4) will move to (9, 4); a zombie moving down from (3, 9) will move to (3, 0).

Each time a zombie takes a step, the new location should be logged, eg:
`zombie 0 moved to (2,3).`

If a zombie occupies the same square as a creature, the creature is transformed into another zombie.

Each time a zombie infects a creature this should be logged, eg:
`zombie 0 infected creature at (3,3)`

The creatures are aware of the zombie's presence but are so frightened that they never move.

Once a zombie has completed its movement, the first newly created zombie moves using the same sequence as the original zombie, then the second newly created zombie moves, and so on, in order of infection. Each zombie performs the same sequence of moves. Once all zombies have completed moving, the final positions of all zombies and creatures should be output, then the program ends.

Your task is to write a program that runs the above simulation using the following input parameters:
- dimensions of the grid (N),
- the initial position of the zombie,
- a list of initial positions of the creatures ,
- and a list of moves the zombies will make,

All inputs, logs and outputs are not limited to a particular format. You can use console, json, txt, a user interface, or anything else you would like.

Example input and output:

Example input:	Example output:
4 (3,1) (0,1)(1,2)(1,1) RDRU	zombies' positions: (1,1)(2,1)(3,2)(3,1) creatures' positions: none