Game of Life

The universe of the Game of Life is an infinite two-dimensional orthogonal grid of square cells, each of which is in one of two possible states, live or dead. Every cell interacts with its eight neighbours, which are the cells that are directly horizontally, vertically, or diagonally adjacent. At each step in time, the following transitions occur:

Any live cell with fewer than two live neighbours dies, as if by loneliness.
Any live cell with more than three live neighbours dies, as if by overcrowding.
Any live cell with two or three live neighbours lives, unchanged, to the next generation.
Any dead cell with exactly three live neighbours comes to life.

The initial pattern constitutes the 'seed' of the system. The first generation is created by applying the above rules Simultaneously to every cell in the seed — births and deaths happen simultaneously, and
the discrete moment at which this happens is sometimes called a tick. (In other words, each generation is a pure function of the one before.) The rules continue to be applied repeatedly to create further
generations.

Problem.
The inputs below represent the cells in the universe as X or - . X is a alive cell. - is a dead cell or no cell. The below inputs provide the provide pattern or initial cells in the universe. The output is
the state of the system in the next tick (one run of the application of all the rules), represented in the same format.

```
Input A:
(Block pattern)
     X X
     X X

Output A:
     X X
```

```
      X X

--------------------------------------------------------------------
-----------------------
Input B
(Boat pattern)
X X -
X - X
- X -

Output B
X X -
X - X
- X -

--------------------------------------------------------------------
-----------------------
Input C
(Blinker pattern)
      - X -
      - X -
      - X -

Output C
      - - -
      X X X
      - - -

--------------------------------------------------------------------
-----------------------
Input D
(Toad pattern)
      - X X X
      X X X -

Output D
      - - X -
      X - - X
      X - - X
      - X - -
```

Parking Lot

We have a parking lot that has a capacity of N cars at a time. The parking lots are numbered sequentially (always starting from 1) from the point of entry.

Once a car enters the lot, a ticket issued with the following details-

- Slot number - always the first available slot
- Car registration number - assume a format
- Model of the car
- Color of the car

When a car has to leave, the ticket is returned and the information is used to mark the slot free.

The program should provide search functionality to find parked cars by one of these filters-

- Registration number - include Regex support
- Color
- Model of the car

Feel free to assume the structure of the input and output. Your program will be reviewed by a human being.