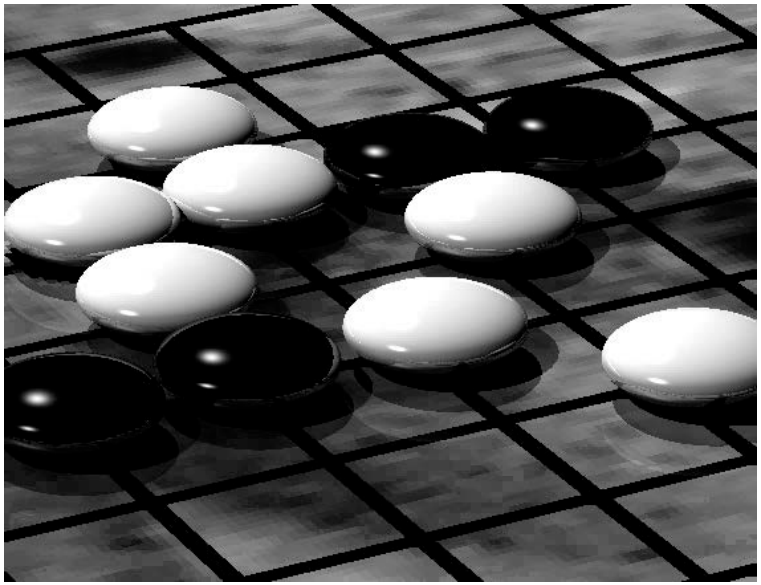
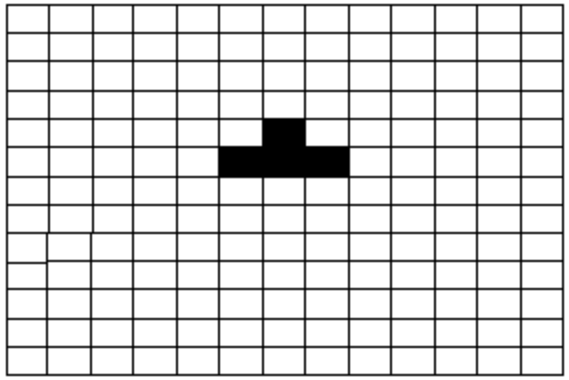
Computer modeling of physical phenomena



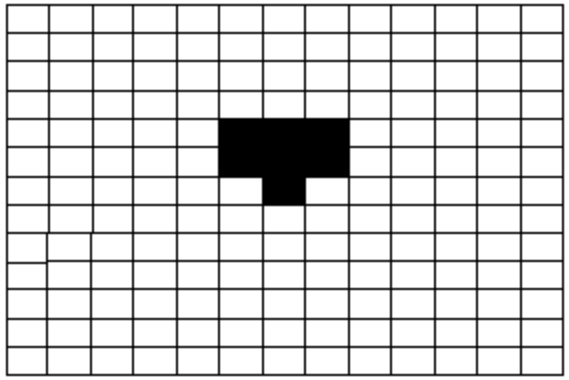
Lab II – Game of Life

Transition rules

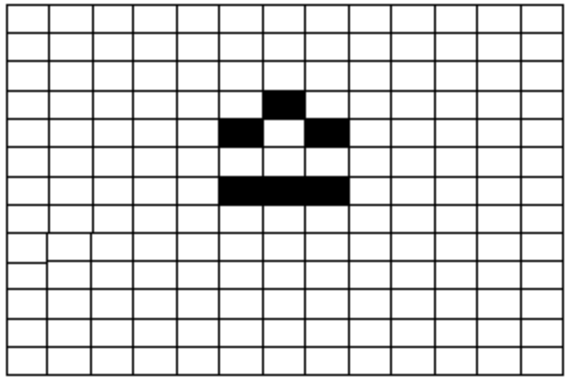
- the cell remains in state 1 (black), if it has 2 or 3 black neighbours (Moore neighbourhood)
- the cell gets transformed to black, if it has exactly 3 black neighbours (birth)
- in other cases the cell remains (or gets transformed to) white



initial state



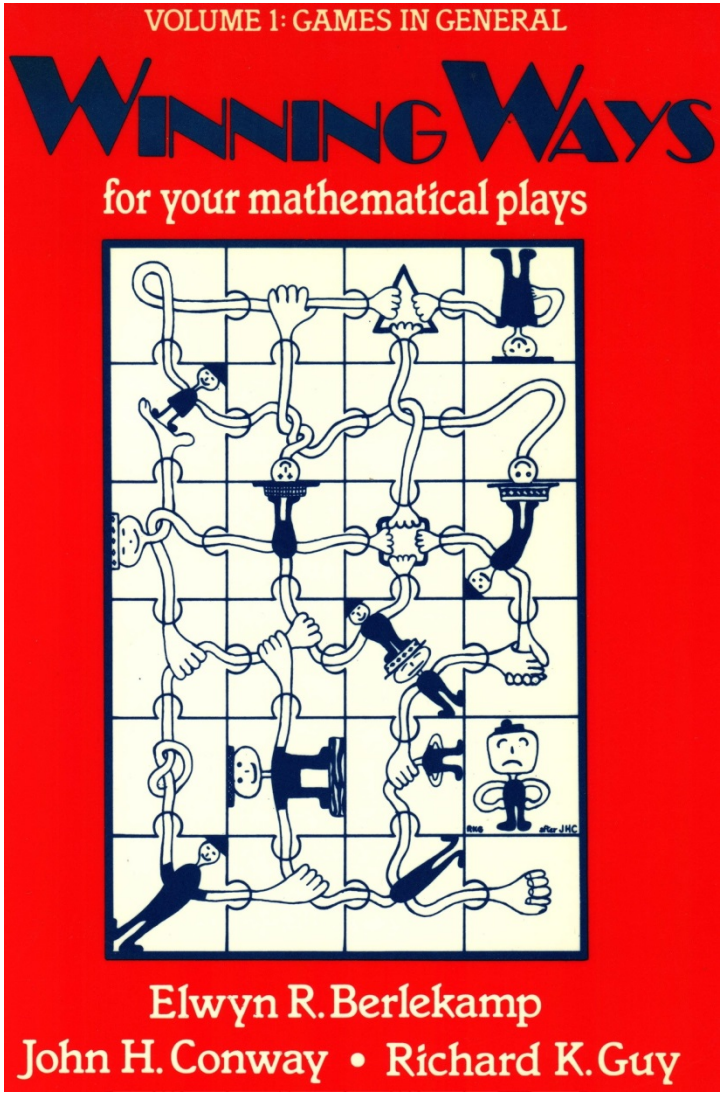
step 1



step 2

John Horton Conway

the rules first appeared in the October 1970 issue of Scientific American, in Martin Gardner's "Mathematical Games" column



Conway's criteria

The rules should lead to interesting and unpredictable behaviour, i.e.:

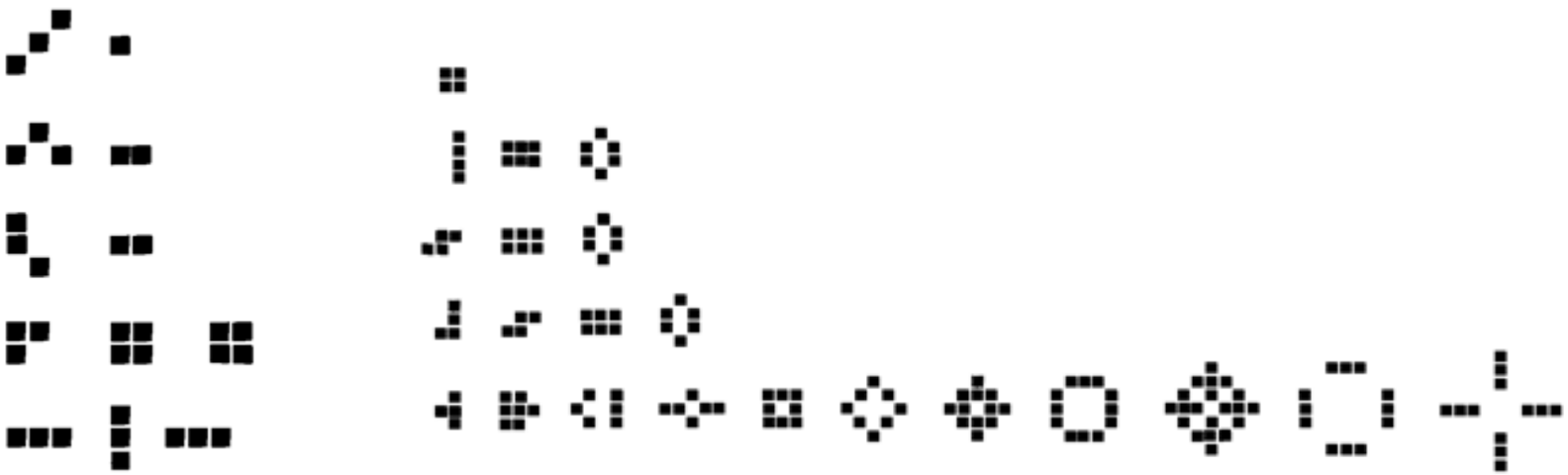
- There should be no initial pattern for which there is a simple proof that the population can grow without limit.
- There should be initial patterns that upparently do grow without limit.
- There should be simple initial patterns that grow and change for a considerable period of time before coming to an end in three possible ways: Fading away completely (from overcrowding or from becoming too sparse), settling into a stable configuration that remains unchanged thereafter, or entering an oscillating phase in which they repeat an endless cycle of two or more periods.

Life forms

N<3 cells always die out

triplets:

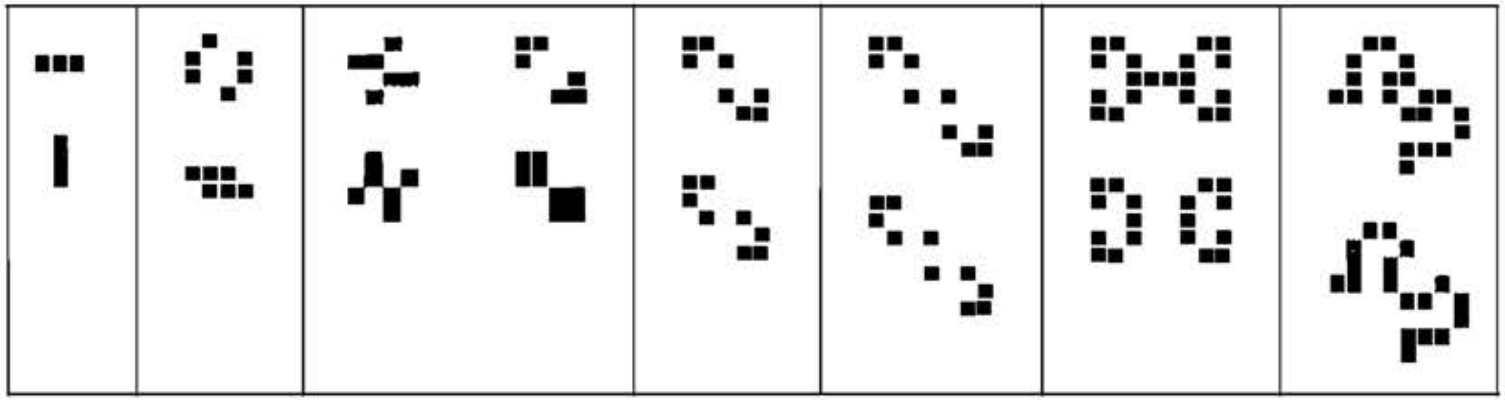
quadruplets:



Conway set up a prize (50$) for a person who will prove, before 1970 that, no initial ‘cell culture’ will continue to extend without bounds

Invariant forms and oscillators

block tub beehive ship snake pond fishhook loaf



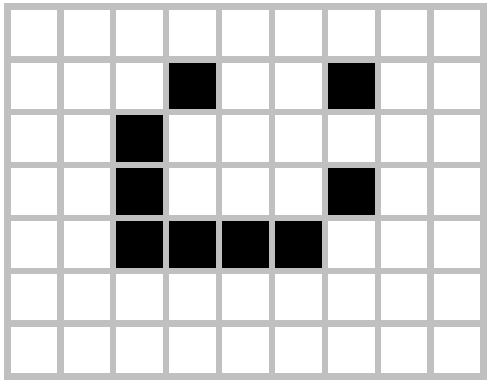
period-2 oscillators

Spaceships

- the maximal speed of information transmission in GoL universe c=1
(in grid lengths per unit time)

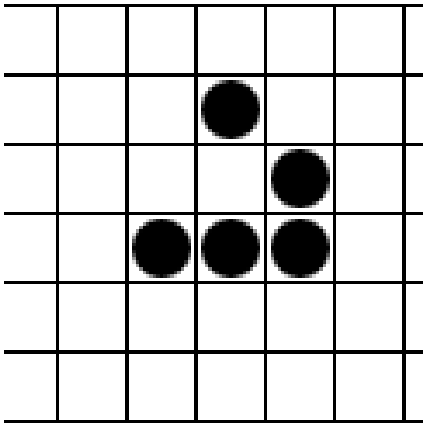
- Conway have shown that the maximum speed with which a form can move is equal to c/2; he called the forms moving with such a speed „spaceships”

Example:

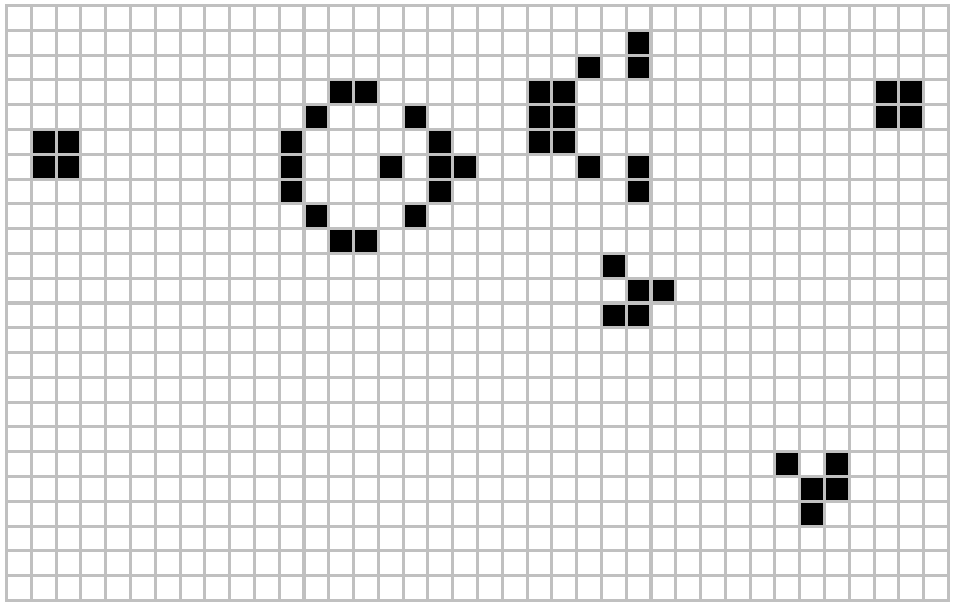


Gliders

Important life forms, moving with velocity c/4. Gliders are important because they are easily produced and can be collided with each other to form more complicated objects and can be used to transmit information over long distances – they are used e.g. in the construction of universal Turing machine in GoL universe

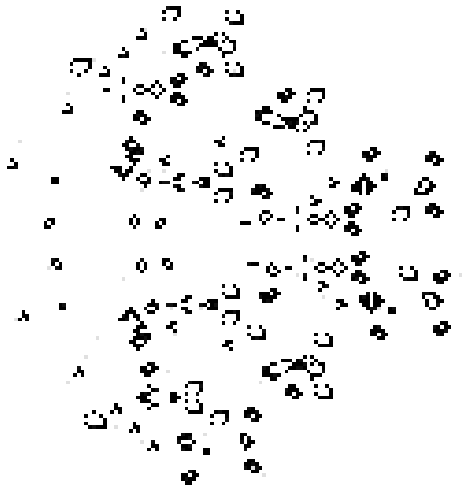


Glider gun



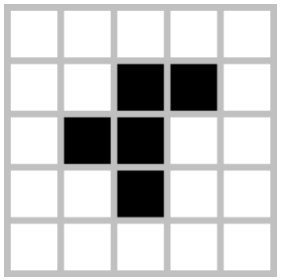
discovered in 1970 by Aprila, Beeler, Gosper, Howell, Schroeppel and Spenciner (they splitted Conway’s 50$ between themselves)

Breeder

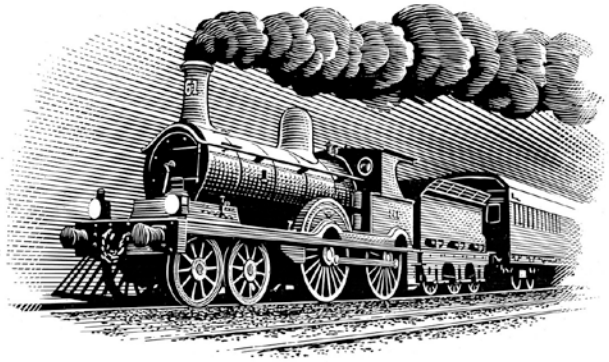


breeds the glider guns...

R-pentonimo

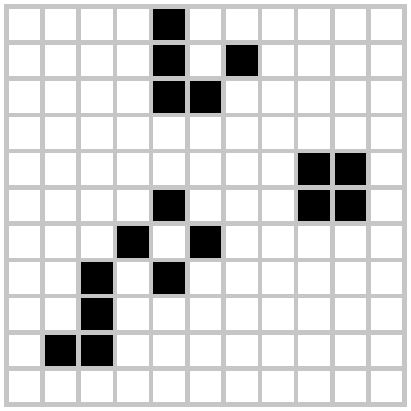


Puffer train...

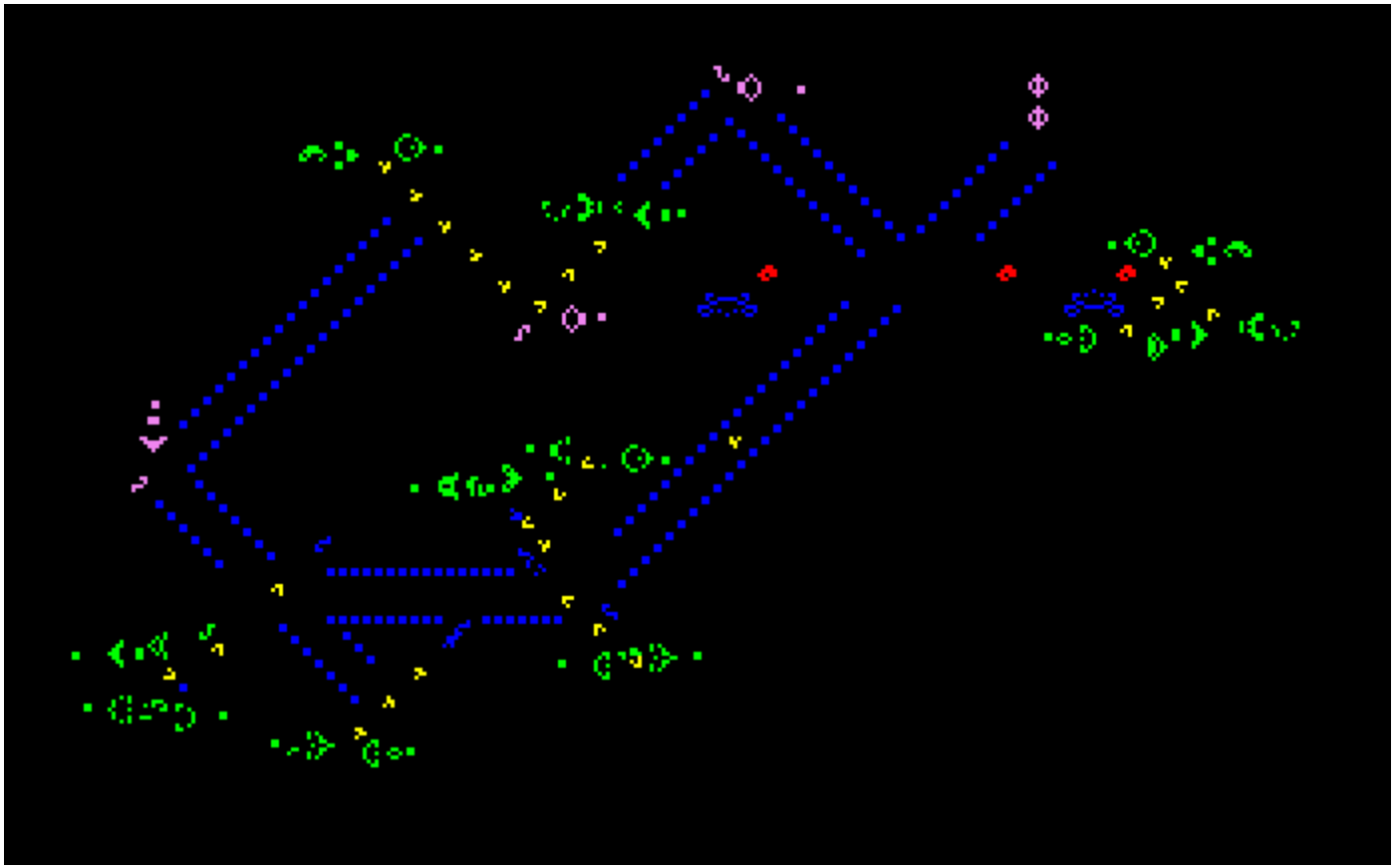


(a spaceship which
moves debris behind)

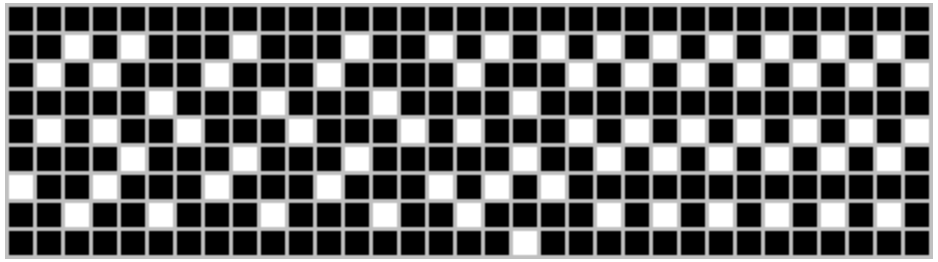
Glider eater

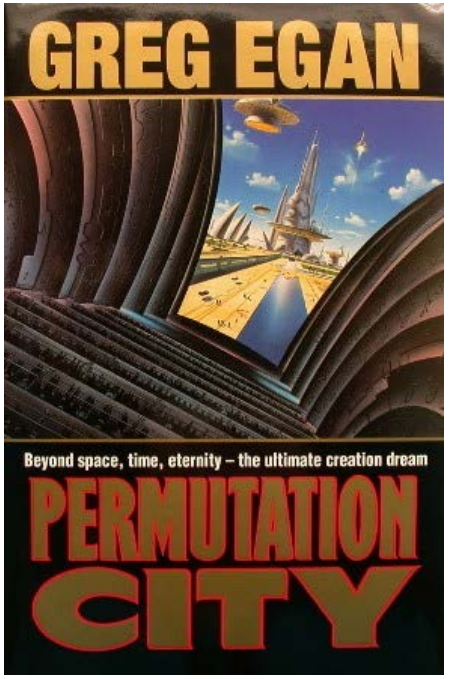


Everything together...



Garden of Eden states

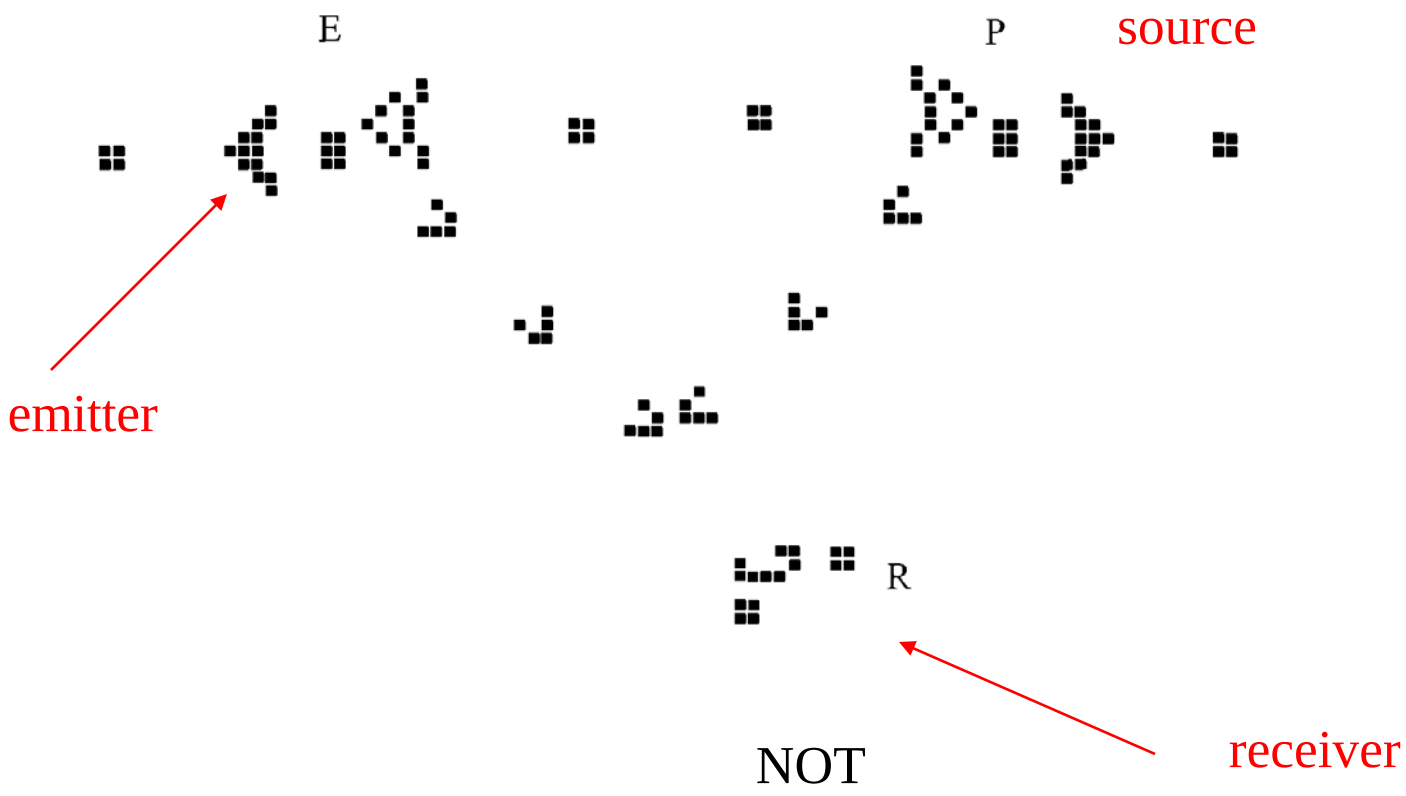




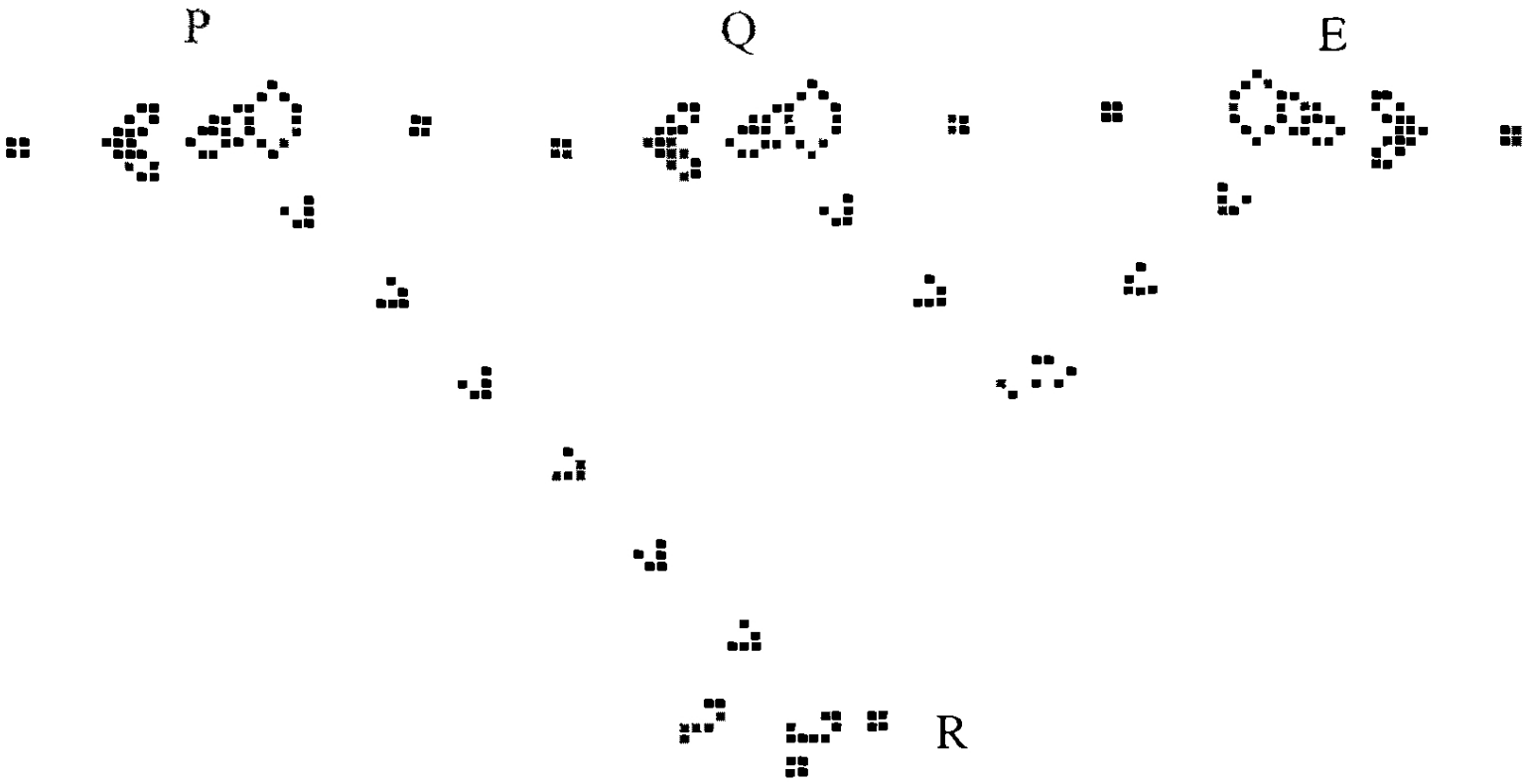
How to tell if we live in a simulation?

Universal computations

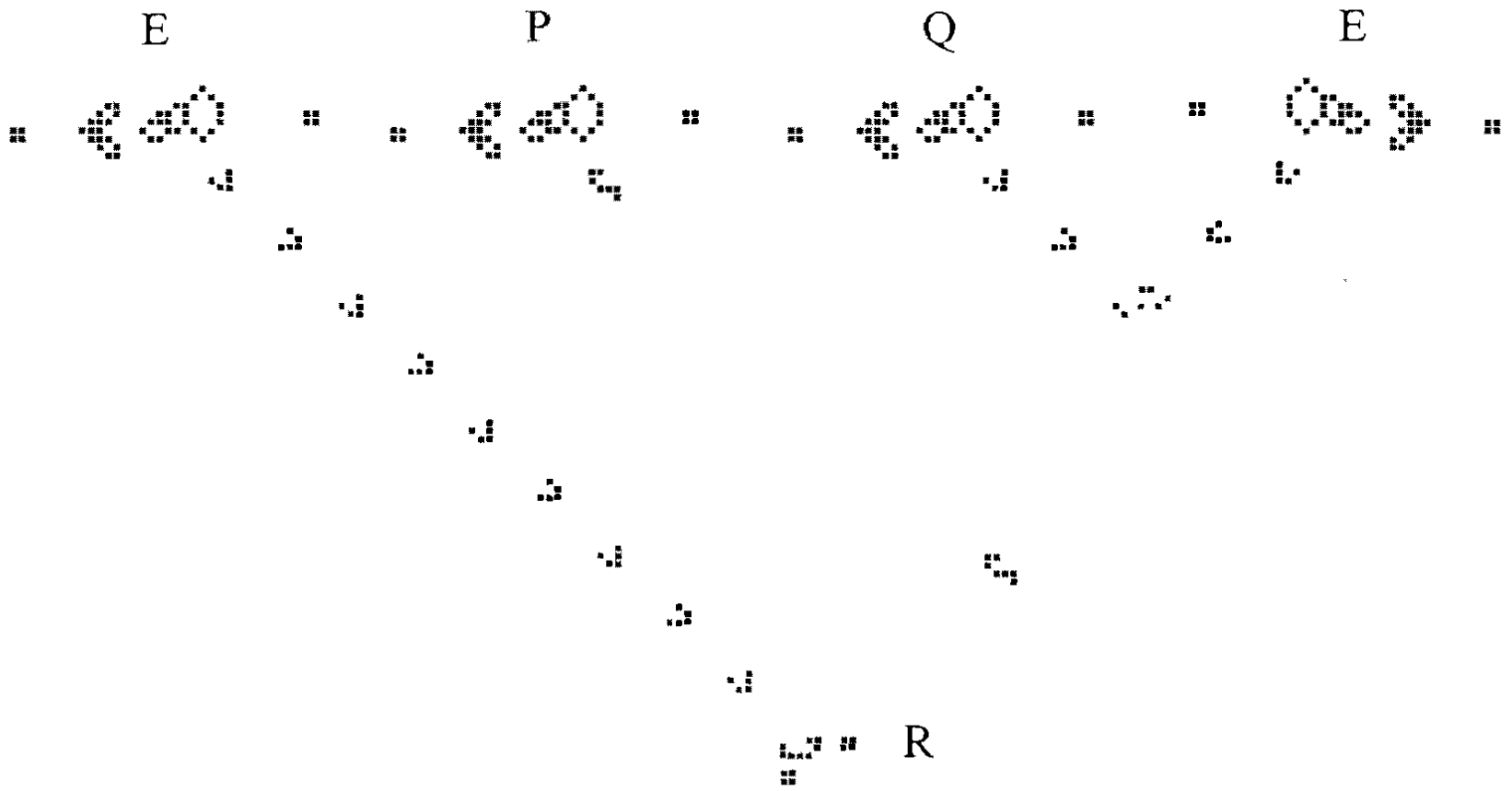
In 1982 John Conway and independently William Gosper have shown that GoL is a universal Turing machine. The proof was based on the construction of logical gates NOT, AND i OR with the use of gliders



AND



OR



Your task: random life forms

- write a game of life simulator in python, which would take any initial configuration and iterate it according to GoL rules
- try it out on a random initial pattern, e.g.:

```
Z = np.random.randint(0,2,(256,512))
```

and iterate it to the steady state, make a corresponding movie

Your task: glider & puffer

- look at the workings of a glider gun, putting in the center of the grid the following structure

[[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1],
[0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1],
[1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[1,1,0,0,0,0,0,0,0,0,1,0,0,0,1,0,1,1,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]]

- same with the puffer:

[[0,1,1,1,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,1,1,1,0],
[1,0,0,1,0,0,0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,1,0,0,1],
[0,0,0,1,0,0,0,0,1,1,0,1,0,0,0,1,0,1,1,0,0,0,0,1,0,0,0],
[0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0],
[0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0],
[0,0,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,0,0,0],
[0,0,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,1,0,0]]

- take two other forms from http://www.radicaleye.com/lifepage/glossary.html

and see how they work

Your task: beyond life

- Experiment with other rules, e.g.:
  - Mazectric (1234/3)
  - Flakes (012345678/3)
  - Serviettes (/234) - start with ■
  - WalledCities (2345/45678)

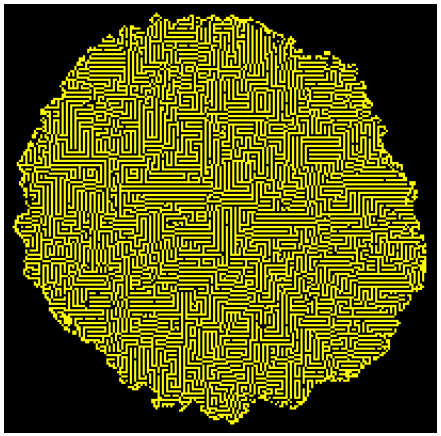
(S/B)

no. of live neighbors needed for a cell to survive

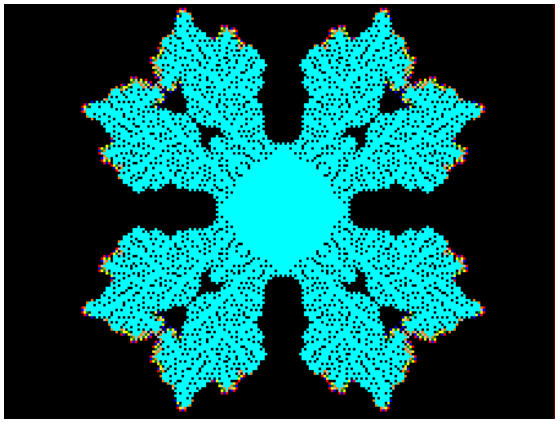
no. of live neighbors needed for a cell to be born

more rules:
http://www.mirekw.com/ca/rullex_life.html

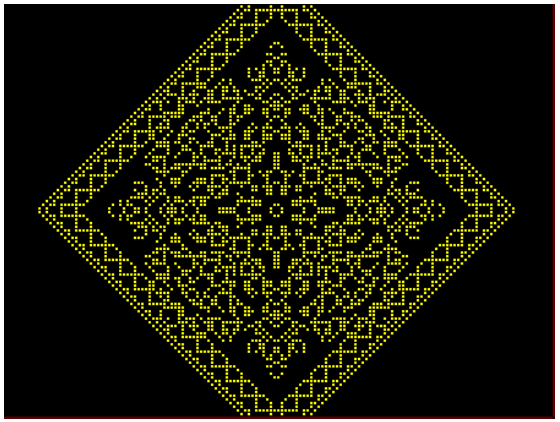
Life: 23/3



Mazectric



Flakes



Serviettes

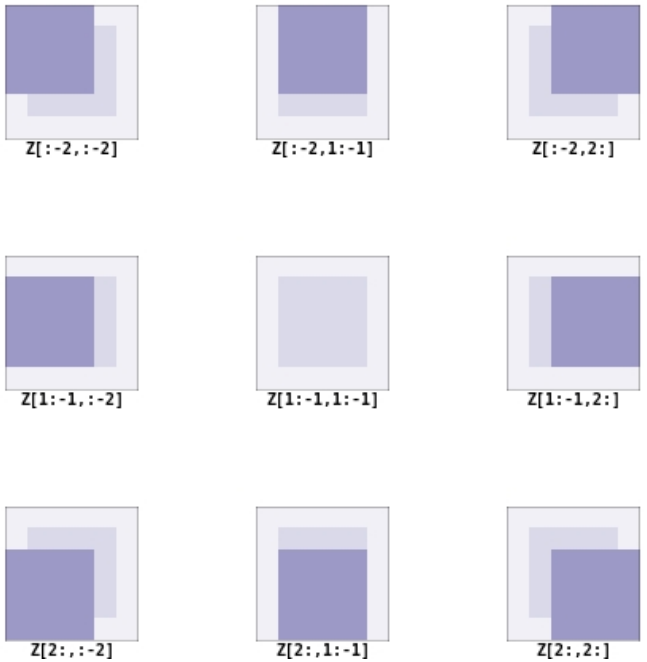
Some hints

- Boundary conditions: put the layer of white (0) cells around your system and keep them at 0
- Vectorize - do not use loops at all! Otherwise you will wait forever...

e.g. # Count neighbours

```
N = (Z[0:-2,0:-2] + Z[0:-2,1:-1] + Z[0:-2,2:] +
     Z[1:-1,0:-2]                + Z[1:-1,2:] +
     Z[2:  ,0:-2] + Z[2:  ,1:-1] + Z[2:  ,2:])
# Apply rules
birth = (N==3) & (Z[1:-1,1:-1]==0)
survive = ....
Z[...] = 0
Z[1:-1,1:-1][birth | survive ] = 1
```

etc.

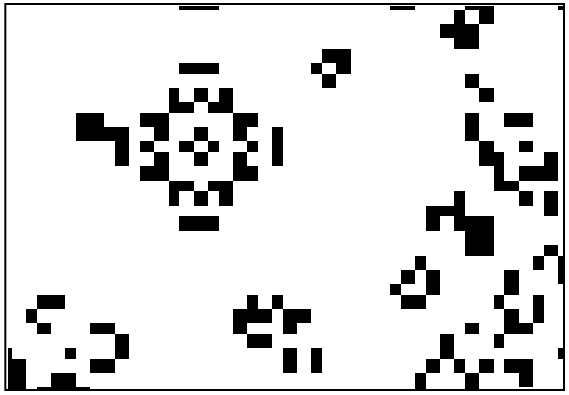


- visualize with imshow

Visualization

```
import matplotlib.pyplot as plt


plt.imshow(Z,interpolation='nearest', cmap=plt.cm.gray_r)
    filename = './png/'+str('%04d' % t) + '.png'
    plt.axis('off')
    plt.savefig(filename, dpi=300,bbox_inches='tight')
    plt.clf()
```



Extra task

Please solve previous tasks using periodic boundary conditions (PBC).