

# Python 数据分析与应用

## 实验三：NumPy 数值计算

丁烨

[dingye@dgut.edu.cn](mailto:dingye@dgut.edu.cn)

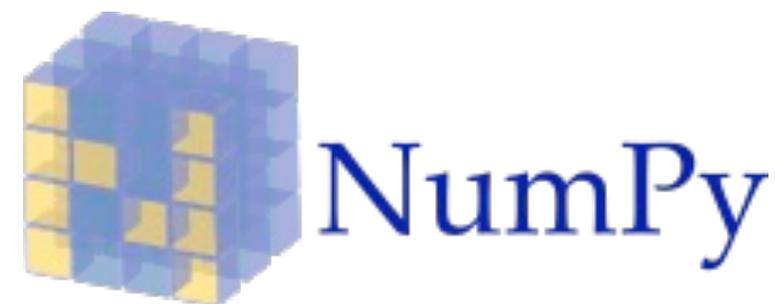
网络空间安全学院

2023-11-01



東莞理工學院  
DONGGUAN UNIVERSITY OF TECHNOLOGY

- ❖ NumPy 是 Python 的一个扩展程序库
- ❖ 官方网站: <https://www.numpy.org/>
- ❖ 源代码: <https://github.com/numpy/numpy>
  
- ❖ 支持大规模的多维数组与矩阵运算
- ❖ 针对数组运算提供大量的数学函数库
- ❖ NumPy 是 SciPy、Matplotlib 等扩展程序库的基础组件
  
- ❖ 原作者: Travis Oliphant
- ❖ 初始版本: 2006 年
- ❖ 最新版本: 1.14.5 (2018 年 6 月 12 日)



- ❖ 使用 pip 安装 NumPy:
- ❖ `pip3 install --user -U numpy`
- ❖ 如果安装不成功, 可尝试使用 apt 安装:
- ❖ `sudo apt install python3-numpy`

❖ 测试 NumPy 是否安装成功:

```
>>> import numpy as np
>>> x = np.array([1, 2, 3])
>>> x
array([1, 2, 3])
```

- ❖ NumPy 的主要数据结构是一个同构多维数组，大于矩阵的概念
- ❖ 维度 (dimension) 在 NumPy 中称为 axis (原意坐标轴，方便起见仍可称为维度)
- ❖ 例如，在三维空间中的一个坐标：
  - ❖ [1, 2, 1]
  - ❖ 是一个一维数组，包含三个元素 (element)，即长度为 3
- ❖ [[ 1., 0., 0.], [ 0., 1., 2.]]
- ❖ 是一个二维数组，第一维长度为 2，第二维长度为 3

- ❖ 康威生命棋 (Conway's Game of Life)
- ❖ 英国数学家 John Horton Conway 在 1970 年发明的一种细胞自动机
- ❖ 它最初于 1970 年 10 月在《科学美国人》杂志上
- ❖ Martin Gardner 的“数学游戏”专栏出现
  
- ❖ 细胞自动机 (Cellular Automaton)
- ❖ 一种离散模型
- ❖ 由若干个方格组成，每个方格按照一定的规律随着时间演变成有限的状态
- ❖ 演变规律通常由方格的相邻方格（“邻域”）决定
- ❖ 每次演变时，每个方格均遵从同一规矩一齐演变

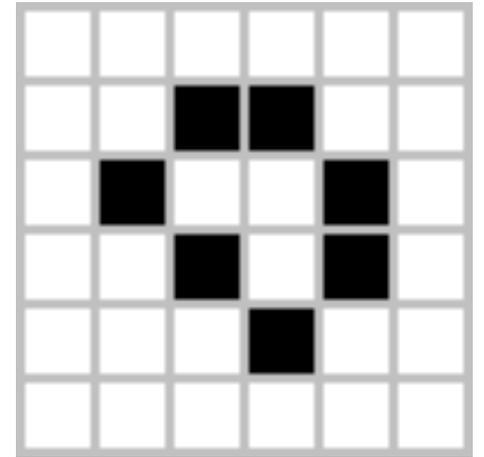
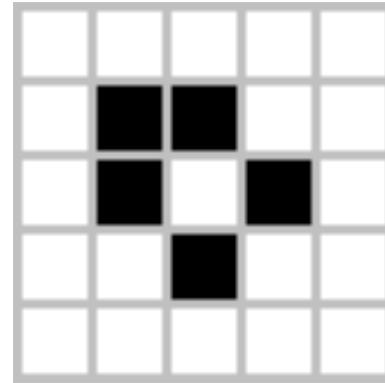
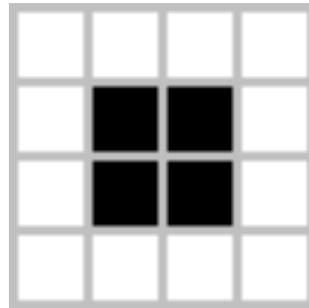
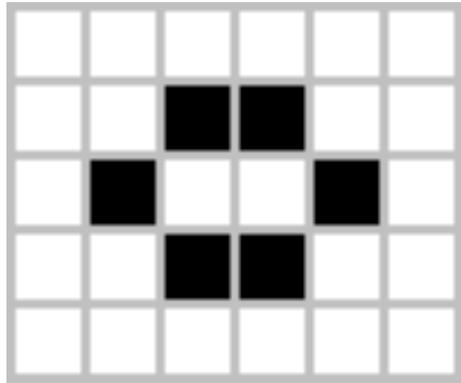
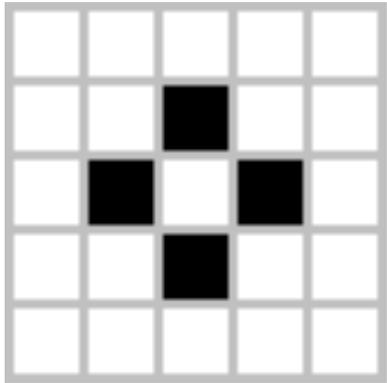
- ❖ 生命游戏中，对于任意细胞，规则如下：
- ❖ 每个细胞有两种状态：**存活**或**死亡**
- ❖ 每个细胞与以自身为中心的周围八格细胞产生互动
  
- ❖ **当前细胞为存活状态时：**
- ❖ 当周围的存活细胞低于（不包含）2个时，该细胞变成死亡状态（模拟生命数量稀少）
- ❖ 当周围有2个或3个存活细胞时，该细胞保持原样
- ❖ 当周围有超过3个存活细胞时，该细胞变成死亡状态（模拟生命数量过多）
  
- ❖ **当前细胞为死亡状态时：**
- ❖ 当周围有3个存活细胞时，该细胞变成存活状态（模拟繁殖）

- ❖ 可以把最初的细胞结构定义为种子
- ❖ 当所有在种子中的细胞同时被以上规则处理后，可以得到第一代细胞图
- ❖ 按规则继续处理当前的细胞图，可以得到下一代的细胞图，周而复始

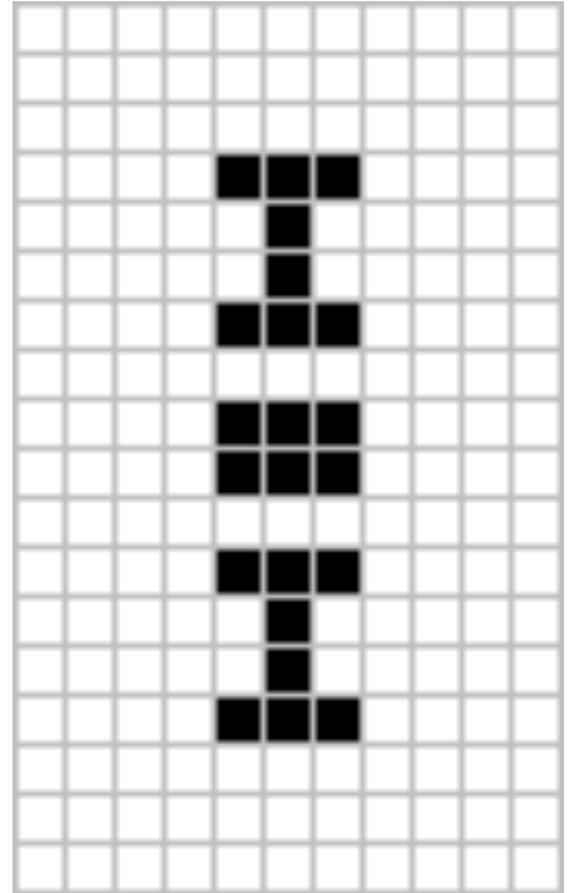
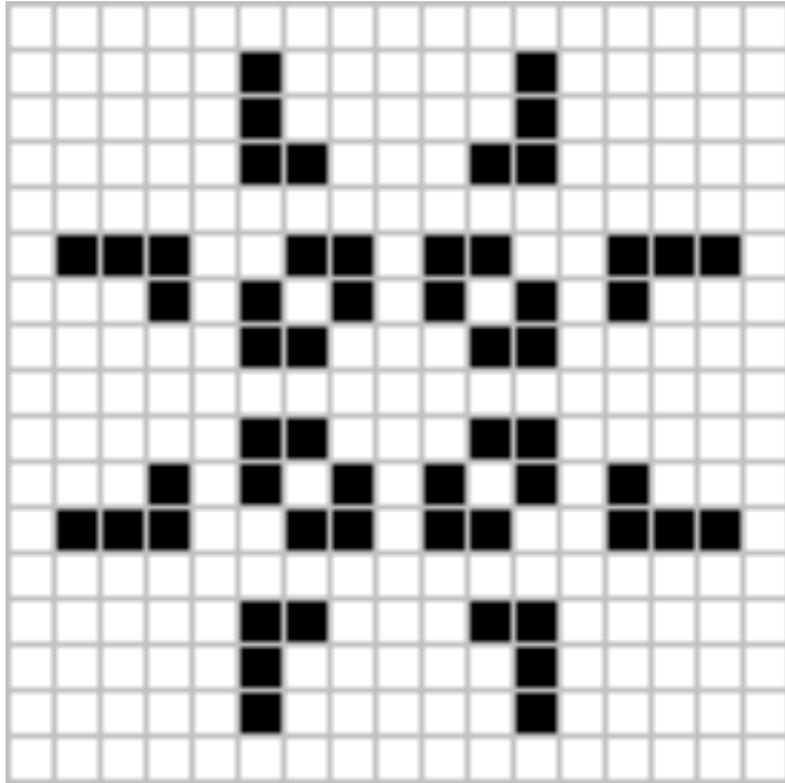
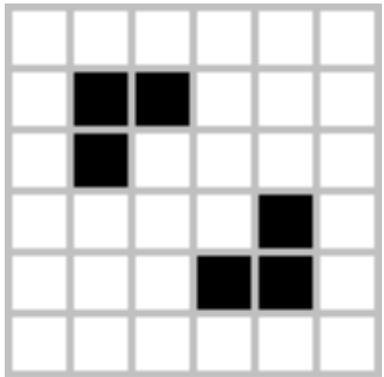
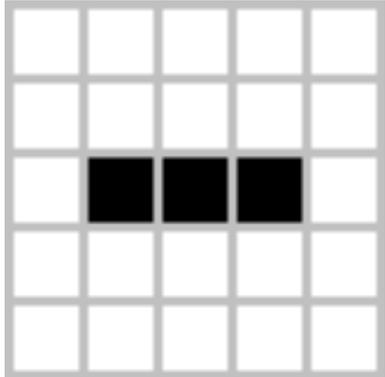
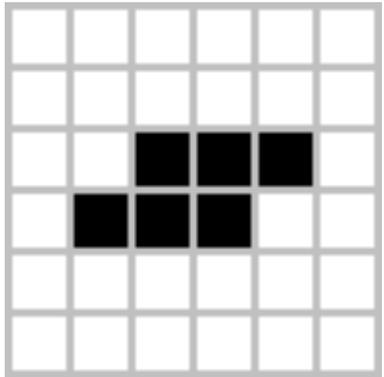
- ❖ 康威生命棋是一个零玩家游戏
- ❖ 它包括一个二维矩形世界，这个世界中的每个方格居住着一个活着的或死了的细胞
- ❖ 一个细胞在下一个时刻生死取决于相邻八个方格中活着的或死了的细胞的数量
- ❖ 如果相邻方格活着的细胞数量过多，这个细胞会因为资源匮乏而在下一个时刻死去
- ❖ 相反，如果周围活细胞过少，这个细胞会因太孤单而死去

- ❖ 实际中，玩家可以设定周围活细胞的数目怎样时才适宜该细胞的生存
- ❖ 如果这个数目设定过高：
  - ❖ 世界中的大部分细胞会因为找不到太多的活的邻居而死去，直到整个世界都没有生命
  - ❖ 如果这个数目设定过低：
    - ❖ 世界中会被生命充满而没有什么变化
- ❖ 康威猜想所有的物种到最后都只会会有三种结局：
  - ❖ 灭绝（所有的细胞都消失了）
  - ❖ 稳定（跟前一轮的模样相同）
  - ❖ 振荡（跟前  $k$  轮的模样相同）

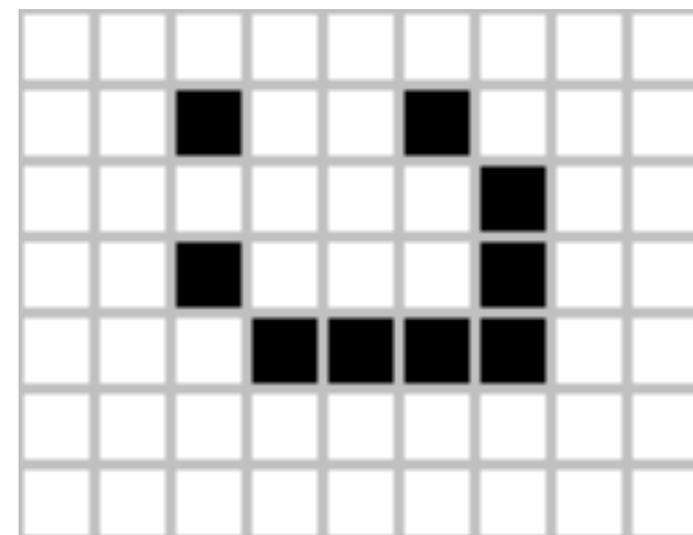
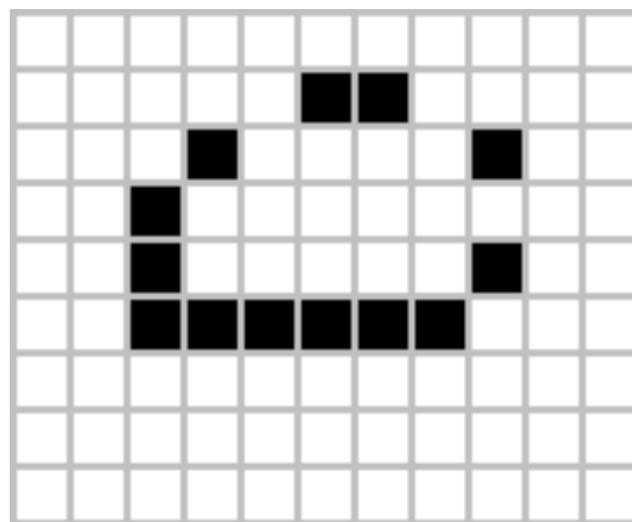
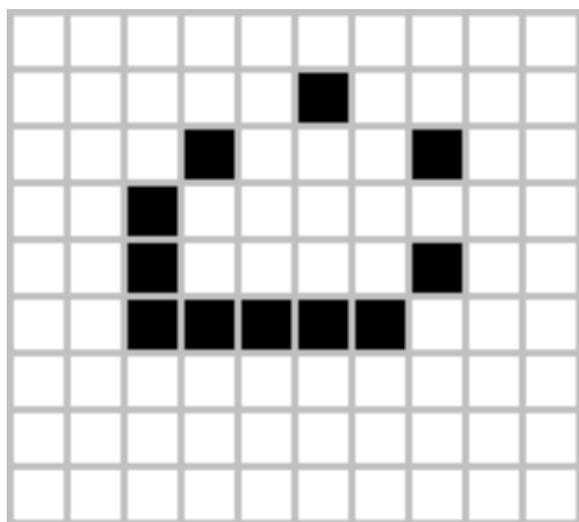
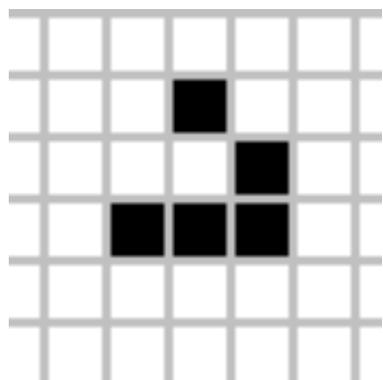
### ❖ 稳定状态



### ❖ 振荡状态



### ❖ 会移动的振荡状态



```
import sys

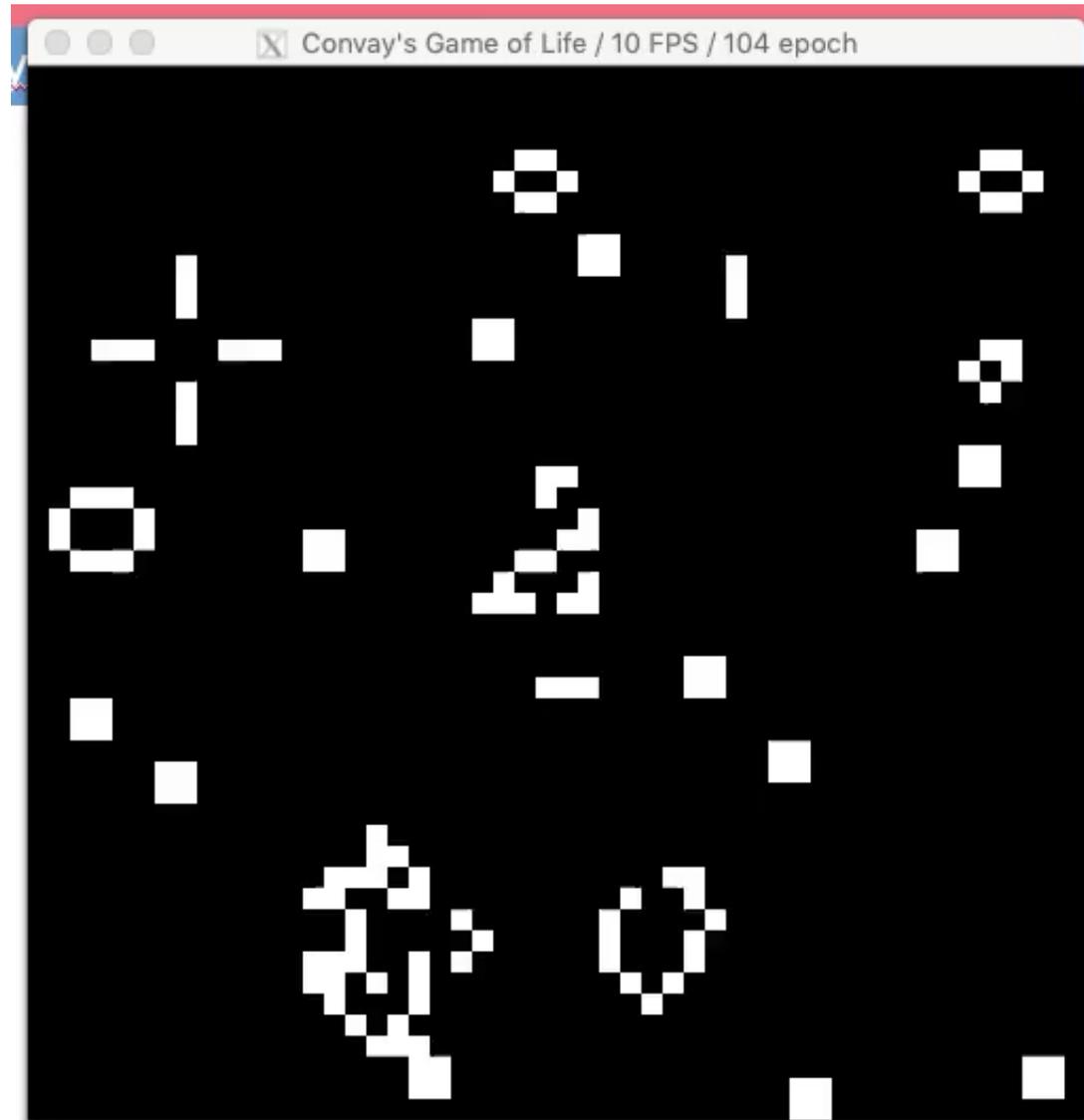
import numpy as np
import pygame

if __name__ == '__main__':
    main()
```

```
def life_step(x, y, world):  
    neighbours = np.sum(world[x - 1: x + 2, y - 1: y + 2]) - world[x, y]  
    if world[x, y] == 1 and not 2 <= neighbours <= 3:  
        return 0  
    elif neighbours == 3:  
        return 1  
    return world[x, y]
```

```
def main():  
    screen_size = 500  
    tile_size = 10  
    background_color = (0, 0, 0)  
    cell_color = (255, 255, 255)  
    pygame.init()  
    screen = pygame.display.set_mode((screen_size, screen_size))  
    pygame.display.set_caption('Conway\'s Game of Life')  
    clock = pygame.time.Clock()  
    world = np.random.choice(a=[0, 1], size=(screen_size // tile_size + 1, screen_size // tile_size + 1))  
    fps = 10  
    epoch = 0
```

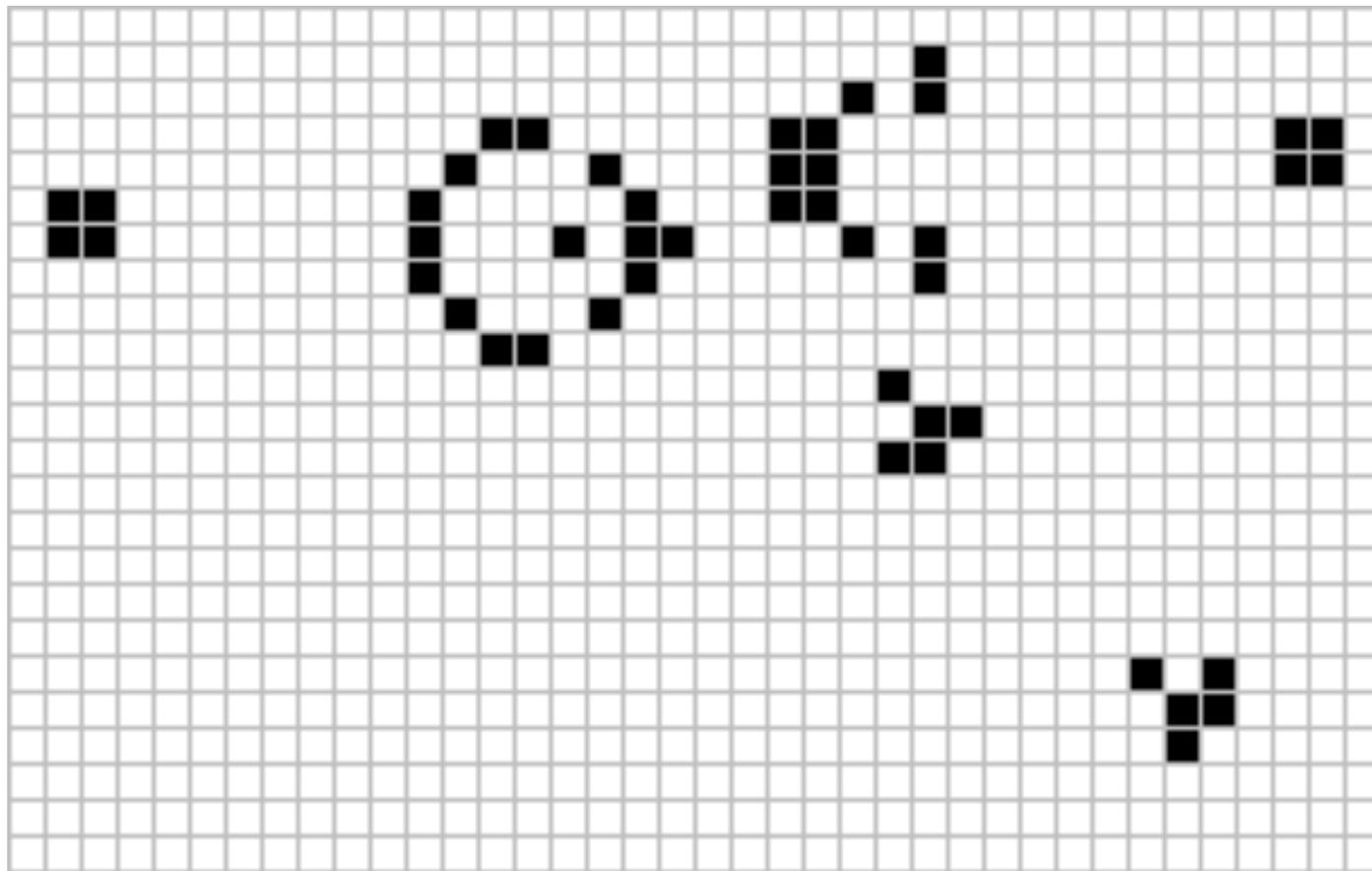
```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            sys.exit()
    pygame.display.set_caption("Conway's Game of Life / " + str(fps) + " FPS / " + str(epoch) + " epoch")
    screen.fill(background_color)
    new_world = np.copy(world)
    for (x, y), value in np.ndenumerate(world):
        new_world[x, y] = life_step(x, y, world)
        if new_world[x, y] == 1:
            pygame.draw.rect(screen, cell_color, (tile_size * (x - 1), tile_size * (y - 1), tile_size, tile_size), 0)
    world = new_world
    epoch += 1
    pygame.display.update()
    clock.tick(fps)
```



- ❖ 通过设定种子细胞，可以得到复杂的“智能”细胞
- ❖ [https://upload.wikimedia.org/wikipedia/commons/e/e6/Conways\\_game\\_of\\_life\\_breeder\\_animation.gif](https://upload.wikimedia.org/wikipedia/commons/e/e6/Conways_game_of_life_breeder_animation.gif)



- ❖ 为康威生命棋提供种子细胞使其产生 “Gosper's Glider Gun”
- ❖ [http://www.conwaylife.com/wiki/Gosper\\_glider\\_gun](http://www.conwaylife.com/wiki/Gosper_glider_gun)



❖ 提示:

❖ 种子细胞状态:

❖ <http://www.conwaylife.com/patterns/gosperglidergun.cells>

```
[(5, 1), (5, 2), (6, 1), (6, 2), (5, 11), (6, 11), (7, 11), (4, 12),  
(3, 13), (3, 14), (8, 12), (9, 13), (9, 14), (6, 15), (4, 16), (5, 17),  
(6, 17), (7, 17), (6, 18), (8, 16), (3, 21), (4, 21), (5, 21), (3, 22),  
(4, 22), (5, 22), (2, 23), (6, 23), (1, 25), (2, 25), (6, 25), (7, 25),  
(3, 35), (4, 35), (3, 36), (4, 36)]
```

❖ 提示:

❖ 将随机生成种子细胞的代码:

```
world = np.random.choice(a=[0, 1], size=(screen_size // tile_size + 1, screen_size // tile_size + 1))
```

❖ 修改为:

```
world = np.zeros((screen_size // tile_size + 1, screen_size // tile_size + 1))
fun = [(5, 1), (5, 2), (6, 1), (6, 2), (5, 11), (6, 11), (7, 11), (4, 12),
        (3, 13), (3, 14), (8, 12), (9, 13), (9, 14), (6, 15), (4, 16), (5, 17),
        (6, 17), (7, 17), (6, 18), (8, 16), (3, 21), (4, 21), (5, 21), (3, 22),
        (4, 22), (5, 22), (2, 23), (6, 23), (1, 25), (2, 25), (6, 25), (7, 25),
        (3, 35), (4, 35), (3, 36), (4, 36)]
for i in fun:
    world[i[0] + 1][i[1] + 1] = 1
```

- ❖ 在作业系统中下载并完成本实验课对应实验报告
- ❖ <https://hw.dgut.edu.cn/>
- ❖ **注意：**所有标识为 \* 的地方都需要填写
- ❖ **截止日期：**2023-11-08 23:59:59

课程名称：Python 数据分析与应用

学期：20

实验名称	Python 语言回顾			实验序号	
姓名	***	学号	***	班级	
实验地点	***	实验日期	***	指导老师	
教师评语	-			实验成绩	
				百分制	
同组同学	无				

## 四、实验作业及分析

### 4.1 实验过程

1) \*\*\* 请将详细实验过程填写在此处 \*\*\*

### 4.2 实验结果

\*\*\* 请将实验结果截图填写在此处 \*\*\*

## 五、实验总结

\*\*\* 请撰写一段 200 字左右的实验总结 \*\*\*

*GOOD LUCK!*