

Реализация игры Конвея на языке программирования Python

Кизянов Антон Олегович

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В данной статье описан процесс создания игры «Конвея» для демонстрации работы принципов, заложенных в игру. Для создания используется язык программирования Python. Созданная игра позволяет визуально увидеть, как ведут себя клетки в зависимости от правил, заложенных Джоном Конвеем.

Ключевые слова: Python, Конвей, игра

Implementing Conway's Game in Python Programming Language

Kizyanov Anton Olegovich

Sholom-Aleichem Priamursky State University

Student

Abstract

This article describes the process of creating a Conway game to demonstrate how the principles behind the game work. For creation, the programming language Python is used. The created game allows you to visually see how cells behave depending on the rules laid down by John Conway.

Keywords: Python, Conway, game

«Игра жизни» Конвея — это симулятор клеточного автомата, который следует простым правилам для создания интересных паттернов. Он был изобретен математиком Джоном Конвеем в 1970 году и популяризирован в колонке Мартина Гарднера «Математические игры» в Scientific American. Сегодня это фаворит среди программистов и компьютерных ученых, хотя это скорее интересная визуализация, чем настоящая «игра». Двумерная доска имеет сетку «ячеек», каждая из которых следует трем простым правилам:

- Живые клетки с двумя или тремя соседями остаются в живых на следующем этапе моделирования.
- Мертвые клетки с ровно тремя соседями оживают на следующем этапе моделирования.
- Любая другая клетка умирает или остается мертвой на следующем этапе моделирования.

Живое или мертвое состояние клеток на следующем этапе моделирования полностью зависит от их текущего состояния. Клетки не «помнят» никаких старых состояний. Существует большое количество

исследований, касающихся закономерностей, порождаемых этими простыми правилами.

Цель исследования – написать игру «Конвея» для демонстрации работы его принципов на языке программирования Python.

Ранее этим вопросом интересовались А.Ю. Субботина, Н.И. Хохлов развивали тему «Реализация клеточных автоматов "игра жизнь" и клеточного автомата кохомото-ооно с применением технологии mpi» [1] в которой рассмотрен анализ результатов, полученных участниками летней школы по высокопроизводительным вычислениям МФТИ-2010 во время практикума по технологии MPI. В качестве проекта была предложена трехмерная версия игры Конвея «Жизнь». Разобраны основные способы решения, используемые участниками при разработке, приведена их теоретическая и практическая оценка по масштабируемости. Л.С. Митыпов с темой «О моделировании клеточных автоматов» [2], а подробнее про новую область прикладной математики - компьютерная геометрия, которая позволяет достаточно эффективно моделировать объекты и создавать их изображение непосредственно на экране монитора. В статье рассматривается способ математической визуализации одномерных и двумерных клеточных автоматов с использованием возможностей среды программирования Small Basic. В работе приводятся способы построения некоторых фрактальных кривых: Коха, Госпера, Гильберта, Леви. Двумерный клеточный автомат реализован на примере игры Джона Конвея «Жизнь». Представленный в статье, материал может быть полезен читателям, интересующимся компьютерной геометрией, и использован в освоении современных технологий моделирования. Г.П. Щеглов опубликовал статью «Некоторые направления исследований клеточного автомата конвея» [3] в статье излагаются направления исследования клеточного автомата Конвея: поиск предшественника монохромной фигуры, расширения по играм в цвете и объеме, спиралевидные системы и циклотроны, бильярдные и лабиринтные системы, звуковые образы. Дается классификация игр, предлагается парный, активный, в ограниченном пространстве, шахматный вариант.

Состояние ячеек хранится в словарях в переменных `cell` и `nextCells`. В обоих словарях есть кортежи `(x, y)` для ключей (где `x` и `y` - целые числа), «О» для живых клеток и «.» для мертвых клеток. Словарь переменной `cells` представляет текущее состояние ячеек, а `nextCells` хранит словарь для ячеек на следующем этапе моделирования.

```
import copy
import random
import sys
import time
```

```
width_window = 79
height_window = 20
awake = 'O'
late = '.'
```

```
nextCells = {}

for x in range(width_window):
    for y in range(heigth_window):
        if random.randint(0, 1) == 0:
            nextCells[(x, y)] = awake
        else:
            nextCells[(x, y)] = late

while True:
    print('\n' * 50)
    cells = copy.deepcopy(nextCells)

    for y in range(heigth_window):
        for x in range(width_window):
            print(cells[(x, y)], end='')
        print()
    print('Нажмите Ctrl-C, чтобы выйти.')

    for x in range(width_window):
        for y in range(heigth_window):
            left = (x - 1) % width_window
            right = (x + 1) % width_window
            above = (y - 1) % heigth_window
            below = (y + 1) % heigth_window
            numNeighbors = 0
            if cells[(left, above)] == awake:
                numNeighbors += 1
            if cells[(x, above)] == awake:
                numNeighbors += 1
            if cells[(right, above)] == awake:
                numNeighbors += 1
            if cells[(left, y)] == awake:
                numNeighbors += 1
            if cells[(right, y)] == awake:
                numNeighbors += 1
            if cells[(left, below)] == awake:
                numNeighbors += 1
            if cells[(x, below)] == awake:
                numNeighbors += 1
            if cells[(right, below)] == awake:
                numNeighbors += 1
            if cells[(x, y)] == awake and (numNeighbors == 2 or numNeighbors == 3):
                nextCells[(x, y)] = awake
            elif cells[(x, y)] == late and numNeighbors == 3:
                nextCells[(x, y)] = awake
            else:
                nextCells[(x, y)] = late

    try:
        time.sleep(1)
    except KeyboardInterrupt:
        sys.exit()
```

Результат работы кода можно увидеть на рисунке 1.

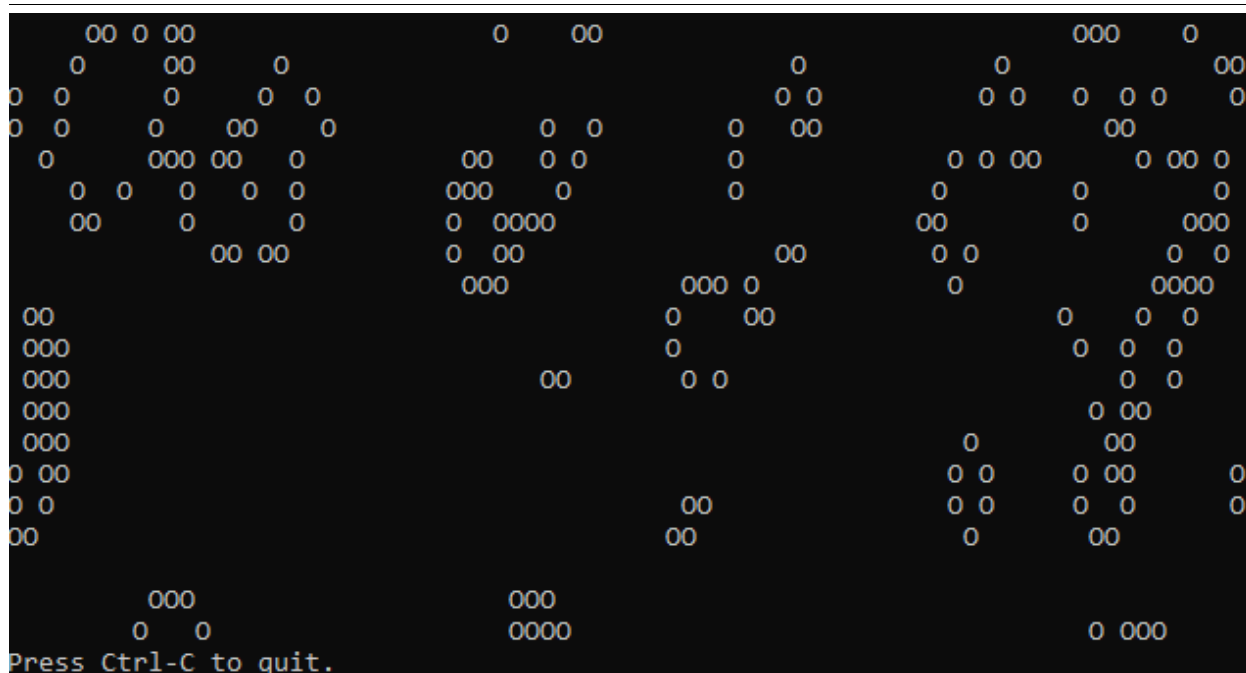


Рис. 1 Процесс игры «Конвея»

Вывод

В этой статье была реализована игра «Конвея» на языке программирования Python. Игра позволяет визуально заметить красоту заложенных в нее принципов, а также добавить свои, для получения более сложной модели распространения клеток.

Библиографический список

1. Субботина А.Ю., Хохлов Н.И. Реализация клеточных автоматов "игра жизнь" и клеточного автомата кохомото-ооно с применением технологии mpi // Компьютерные исследования и моделирование. 2010. Т. 2. № 3. С. 319-322. URL: <https://www.elibrary.ru/item.asp?id=15621766> (Дата обращения: 05.01.2022)
2. Митьпов Л.С. О моделировании клеточных автоматов // В сборнике: Геометрия многообразий и ее приложения. материалы четвертой научной конференции с международным участием. 2016. С. 164-171. URL: <https://www.elibrary.ru/item.asp?id=30356663> (Дата обращения: 05.01.2022)
3. Щеглов Г.П. Некоторые направления исследований клеточного автомата конвея // В мире научных открытий. 2010. № 6-1 (12). С. 82-84. URL: <https://www.elibrary.ru/item.asp?id=16374041> (Дата обращения: 05.01.2022)