Statistics library

Michael V. Koldayev

December 3, 2015

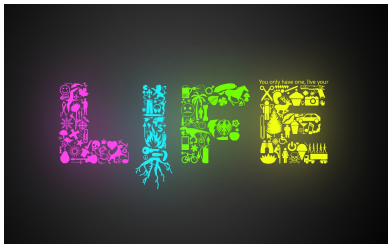


Figure 1: Game of Life

In this report I would like to show the game of life that my team created in class. The program consists of functions and all of them are related to each other. In output we can see a well-run game.

PYTHON CODE

```
#Game Of Live

import random
from graphics import *
from Tkinter import *
import tkMessageBox

#------------------------------------------------------------------------

wMain = Tk()

font = "times", 25, "bold"
lTitle = Label(wMain, text = "Conway's Game of Life", font = font)
lTitle.place(x = 120, y = 30)

font2 = "times", 20
lOption = Label(wMain, text = "Number of Starting Cells", font = font2)
lOption.place(x = 145, y = 80)

font3 = "times", 15
lOption2 = Label(wMain, text = "1/", font = font3)
lOption2.place(x = 155, y = 110)

eEntry = Entry(wMain, bd = 2)
eEntry.place(x = 175, y = 110)

#------------------------------------------------------------------------

def gameCallBack(entry):

    #wMain.destroy()

    p = []
    alives = []
    size = 15

    game = GraphWin("The Game Of Life", 400, 400)
    game.setCoords(-.2, 0, size, size + .2)

    def startUpCallBack():

        live = 1.0/int(entry) * (size**2)

        for i in range(int(live)):
            x = int(random.uniform(0, size))
            y = int(random.uniform(0, size))

            alives.append([x, y])
```

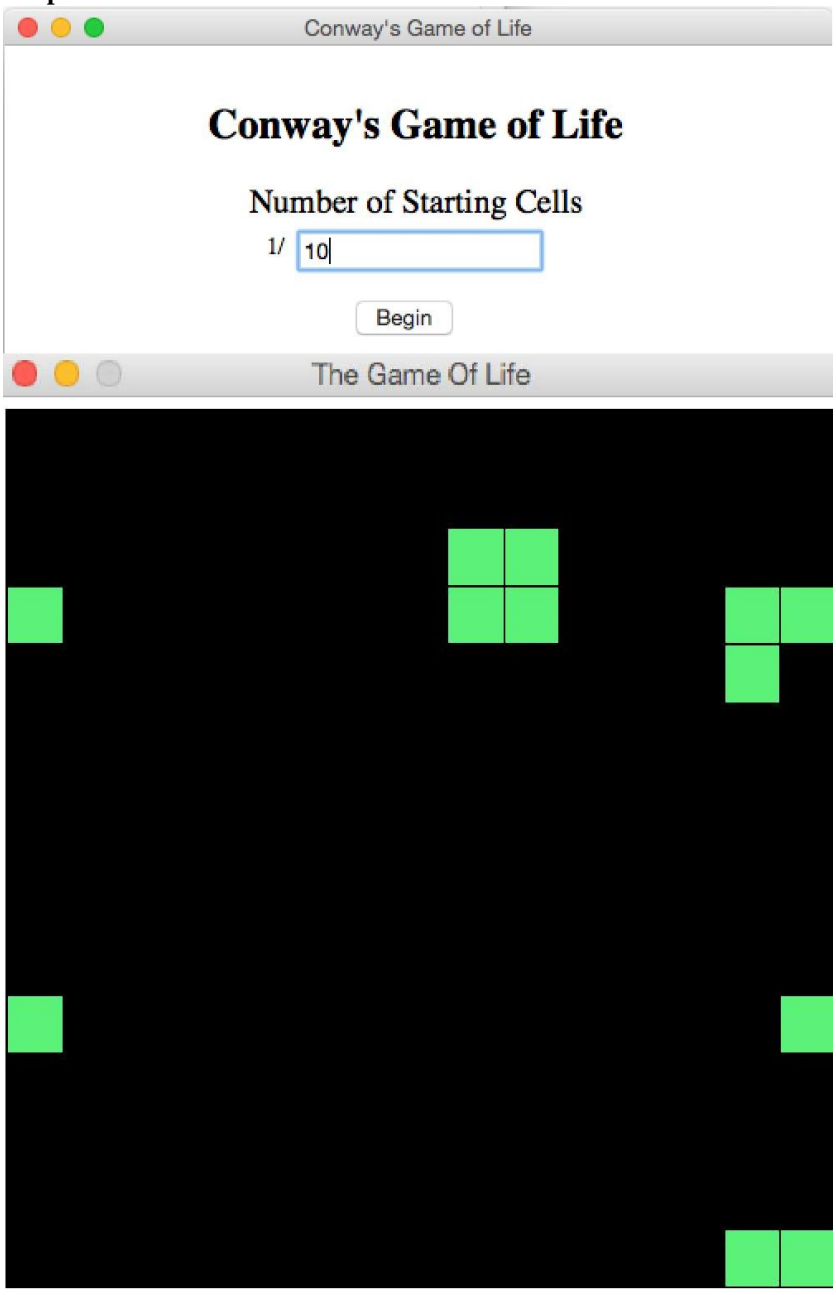
```
    return alives

def firstLoadCallBack():

    startUpCallBack()

    for i in range(size):
        a = []
        for r in range(size):
            a.append(0)
        p.append(a)


    for i in range(size):
        for r in range(size):
            for q in range(len(alives)):
                if alives[q][0] == r:
                    if alives[q][1] == i:
                        p[i][r] = 1
    return p


firstLoadCallBack()


def loaderCallBack(p):

    green = color_rgb(82, 245, 103)
    red = color_rgb(247, 34, 49)
    black = color_rgb(0, 0, 0)

    while True:

        for i in range(size):
            for r in range(size):
                point = Rectangle(Point(r, i), Point(r + 1, i + 1))
                if p[i][r] == 0:
                    point.setFill(black)
                if p[i][r] == 1:
                    point.setFill(green)
                point.draw(game)

        x = []

        for i in range(size):
            g = []
            for r in range(size):
                g.append(p[i][r])
            x.append(g)

        for i in range(size):
            for r in range(size):

                ss = 0

                if not i == (size - 1) and not i == 0:
                    if not r == (size - 1) and not r == 0:

                        ss = ss + p[i + 1][r - 1]
                        ss = ss + p[i + 1][r]
```

```
        ss = ss + p[i + 1][r + 1]
        ss = ss + p[i][r - 1]
        ss = ss + p[i][r + 1]
        ss = ss + p[i - 1][r - 1]
        ss = ss + p[i - 1][r]
        ss = ss + p[i - 1][r + 1]

if i == (size - 1):
    if not r == (size - 1) and not r == 0:
        ss = ss + p[0][r - 1]
        ss = ss + p[0][r]
        ss = ss + p[0][r + 1]
        ss = ss + p[i][r - 1]
        ss = ss + p[i][r + 1]
        ss = ss + p[i - 1][r - 1]
        ss = ss + p[i - 1][r]
        ss = ss + p[i - 1][r + 1]

if i == (size - 1):
    if r == (size - 1):
        ss = ss + p[0][r - 1]
        ss = ss + p[0][r]
        ss = ss + p[0][0]
        ss = ss + p[i][r - 1]
        ss = ss + p[i][0]
        ss = ss + p[i - 1][r - 1]
        ss = ss + p[i - 1][r]
        ss = ss + p[i - 1][0]

if i == (size - 1):
    if r == 0:
        ss = ss + p[0][size - 1]
        ss = ss + p[0][r]
        ss = ss + p[0][r + 1]
        ss = ss + p[i][size - 1]
        ss = ss + p[i][r + 1]
        ss = ss + p[i - 1][size - 1]
        ss = ss + p[i - 1][r]
        ss = ss + p[i - 1][r + 1]
if i == 0:
    if not r == (size - 1) and not r == 0:
        ss = ss + p[i][r - 1]
        ss = ss + p[i][r]
        ss = ss + p[i][r + 1]
        ss = ss + p[i][r - 1]
        ss = ss + p[i][r + 1]
        ss = ss + p[size - 1][r - 1]
        ss = ss + p[size - 1][r]
        ss = ss + p[size - 1][r + 1]

if i == 0:
    if r == (size - 1):
        ss = ss + p[i][r - 1]
        ss = ss + p[i][r]
        ss = ss + p[i][0]
        ss = ss + p[i][r - 1]
        ss = ss + p[i][0]
        ss = ss + p[size - 1][r - 1]
        ss = ss + p[size - 1][r]
        ss = ss + p[size - 1][0]
```

```
                    if i == 0:
                        if r == 0:
                            ss = ss + p[i][size - 1]
                            ss = ss + p[i][r]
                            ss = ss + p[i][r + 1]
                            ss = ss + p[i][size - 1]
                            ss = ss + p[i][r + 1]
                            ss = ss + p[size - 1][size - 1]
                            ss = ss + p[size - 1][r]
                            ss = ss + p[size - 1][r + 1]

                    if not i == (size - 1) and not i == 0:
                        if r == (size - 1):
                            ss = ss + p[i][r - 1]
                            ss = ss + p[i][r]
                            ss = ss + p[i][0]
                            ss = ss + p[i][r - 1]
                            ss = ss + p[i][0]
                            ss = ss + p[i][r - 1]
                            ss = ss + p[i][r]
                            ss = ss + p[i][0]

                    if not i == (size - 1) and not i == 0:
                        if r == 0:
                            ss = ss + p[i][size - 1]
                            ss = ss + p[i][r]
                            ss = ss + p[i][r + 1]
                            ss = ss + p[i][size - 1]
                            ss = ss + p[i][r + 1]
                            ss = ss + p[i][size - 1]
                            ss = ss + p[i][r]
                            ss = ss + p[i][r + 1]

                    if p[i][r] == 1:
                        if ss < 2:
                            x[i][r] = 0
                        if ss > 3:
                            x[i][r] = 0
                        if not ss > 3 and not ss < 2:
                            x[i][r] = 1
                    if p[i][r] == 0:
                        if ss == 3:
                            x[i][r] = 1
                        if not ss == 3:
                            x[i][r] == 0

                    #print ss

                p = []

                for i in range(size):
                    g = []
                    for r in range(size):
                        g.append(x[i][r])
                    p.append(g)

        loaderCallBack(p)

    def startCallBack():
        gameCallBack(int(eEntry.get()))
```

```
bBegin = Button(wMain, text = "Begin", command = startCallBack)
bBegin.place(x = 210, y = 150)

wMain.minsize(width = 500, height = 210)
wMain.maxsize(width = 500, height = 210)
wMain.title("Conway's Game of Life")
wMain.mainloop()
```

Output:



The window with this output appeared. As you clearly see, it is a Gauss distribution