```
“””
Cap 8
“””
from __future__ import print_function, division
#%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

import thinkplot
from matplotlib import rc
rc('animation', html='jshtml')
import warnings
import matplotlib.cbook
warnings.filterwarnings("ignore",category=matplotlib.cbook.mplDeprecation)
from thinkstats2 import RandomSeed
RandomSeed(17)

from Cell2D import Cell2D, Cell2DViewer
from scipy.signal import correlate2d


def add_island(a, height=0.1):
    """Aggiunge un’isola nelcentrodell’array.

    height: altezza dell’isola
    """
    n, m = a.shape
    radius = min(n, m) // 20
    i = n//2
    j = m//2
    a[i-radius:i+radius, j-radius:j+radius] += height

class ReactionDiffusion(Cell2D):
    """Reaction-Diffusion Cellular Automaton."""

    kernel = np.array([[.05, .2, .05],
                       [ .2, -1, .2],
                       [.05, .2, .05]])

    options = dict(mode='same', boundary='wrap')

    def __init__(self, n, params, noise=0.1):
        """Inizializza gli attributi.

        n: numero dirighe
        params: tupla di (Da, Db, f, k)
        """
        self.params = params
        self.array = np.ones((n, n), dtype=float)
        self.array2 = noise * np.random.random((n, n))
        add_island(self.array2)

    def step(self):
        """Esegue un passodi tempo."""
        A = self.array
        B = self.array2
        ra, rb, f, k = self.params

        cA = correlate2d(A, self.kernel, **self.options)
        cB = correlate2d(B, self.kernel, **self.options)
        reaction = A * B**2
        self.array += ra * cA - reaction + f * (1-A)
        self.array2 += rb * cB + reaction - (f+k) * B
class RDViewer(Cell2DViewer):
```

```
    """Genera immagini e animazioni."""
    cmapu = plt.get_cmap()
    cmapv = plt.get_cmap('Blues')

    options = dict(alpha=0.7,
                   interpolation='bicubic')

    def __init__(self, viewee):
        """Inizializza gli attributi.
             viewee: gli oggetti da rappresentare
        """
        self.viewee = viewee
        self.imu = None
        self.imv = None

    def draw(self, grid=False):
        """Disegna le celle."""
        au = self.viewee.array.copy()
        av = self.viewee.array2.copy()

        n, m = av.shape
        plt.axis([0, m, 0, n])
        plt.xticks([])
        plt.yticks([])

        self.options['extent'] = [0, m, 0, n]
        self.imu = plt.imshow(au, cmap=self.cmapu, **self.options)
        self.imv = plt.imshow(av, cmap=self.cmapv, **self.options)

    def animate_func(self, i):
        """raws one frame of the animation."""
        if i > 0:
             self.step(iters=100)

        self.imu.set_array(self.viewee.array)
        self.imv.set_array(self.viewee.array2)
        return (self.imu, self.imv)

params1 = 0.5, 0.25, 0.035, 0.057   # white spots
params2 = 0.5, 0.25, 0.055, 0.062   # coral
params3 = 0.5, 0.25, 0.039, 0.065   # blue spots
params4 = 1.0, 0.50,  0.0367, 0.0649

rd = ReactionDiffusion(n=100, params=params4)
viewer = RDViewer(rd)
anim = viewer.animate(frames=100)
anim

thinkplot.preplot(cols=3)
viewer.step(1000)
viewer.draw()
thinkplot.subplot(2)
viewer.step(2000)
viewer.draw()
thinkplot.subplot(3)
viewer.step(4000)
viewer.draw()
```