**Міністерство освіти і науки України**

**Черкаський національний університет імені Богдана Хмельницького**

**Інститут фізики, математики та комп'ютерно-інформаційних систем**

**Кафедра прикладної математики та інформатики**

**З В І Т**

**з лабораторної роботи**

студента 4-го курсу спеціальності «Прикладна математика»

***Годованюк Матвія Ігоровича***

**Черкаси – 2016 р.**

\*Коди программ у кінці файла

Гра життя

Місце дії цієї гри — «всесвіт» — це площина, поділена на клітинки. Кожна клітинка на цій поверхні може знаходитись в двох станах: бути живою або бути мертвою. Клітинка має вісім сусідів. Розподіл живих клітинок на початку гри називається першим поколінням. Кожне наступне покоління утворюється на основі попереднього за наведеними нижче правилами.

Правила

якщо у живої клітини два чи три сусіди – то вона лишається жити;

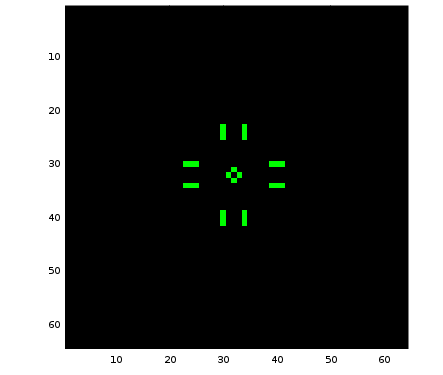
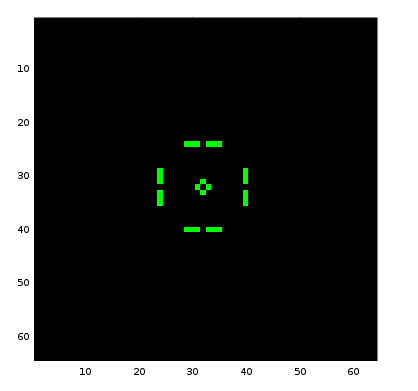
якщо у живої клітини один чи немає сусідів – то вона помирає від «самотності»;

якщо у живої клітини чотири та більше сусідів – вона помирає від «перенаселення»;

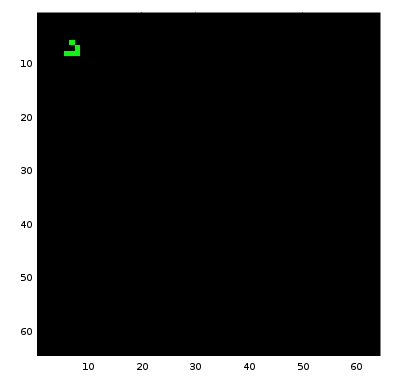
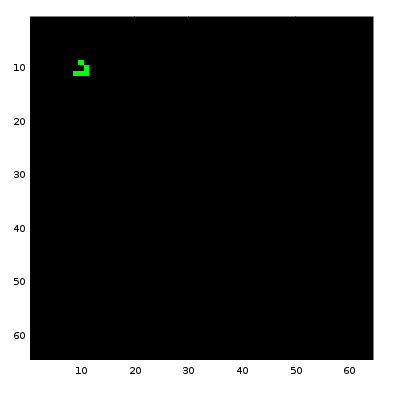
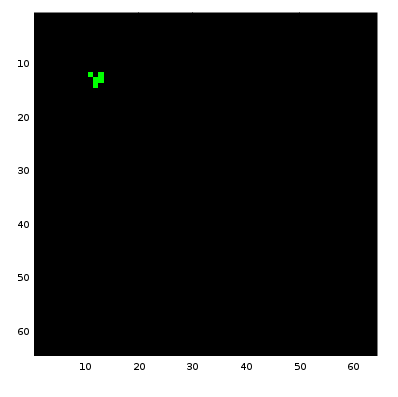
якщо у мертвої клітини рівно три сусіди – то вона оживає.

Приклади станів

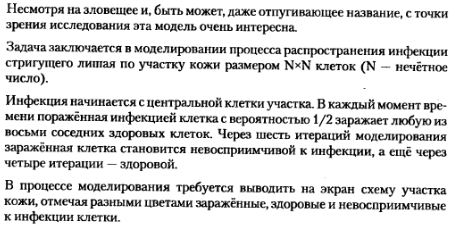
Пульсуючий



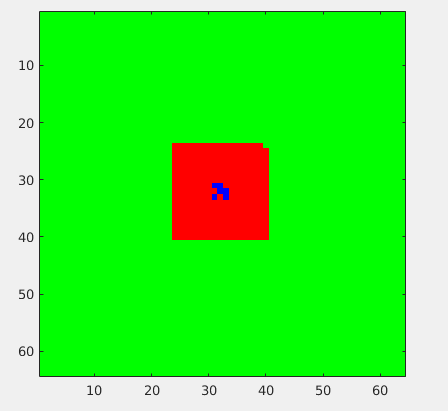
Рухивий

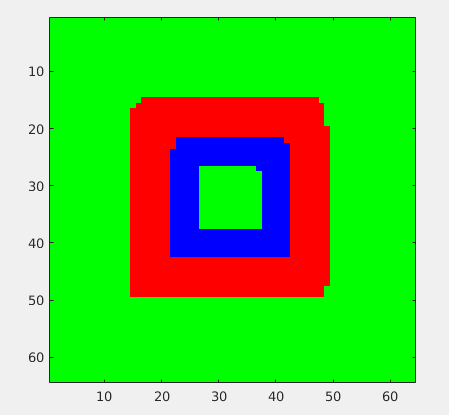


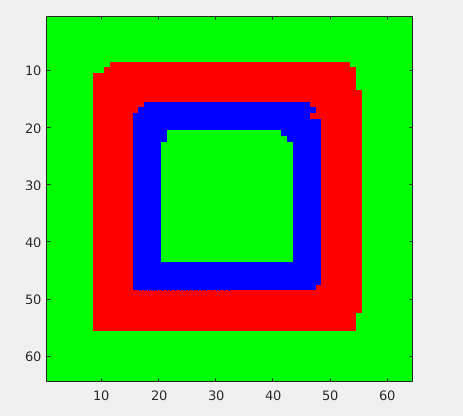
**Стригучий лишай**

****

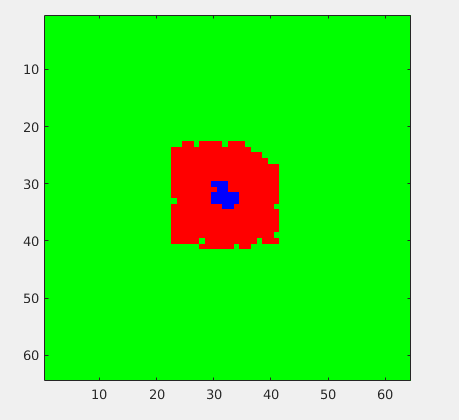
p = 0.9

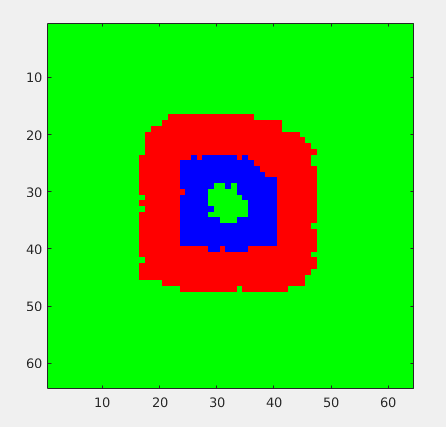


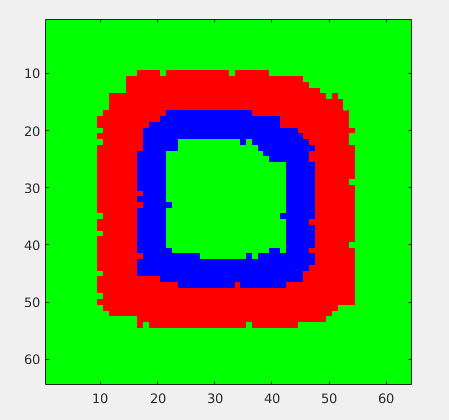




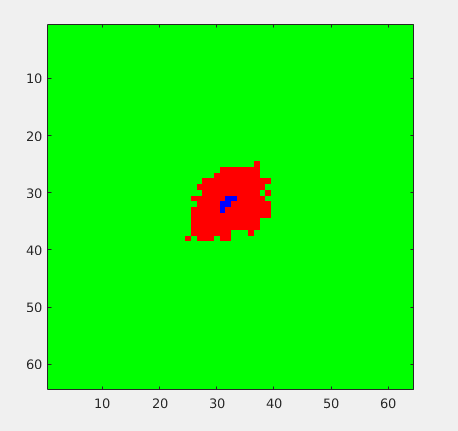
p = 0.6

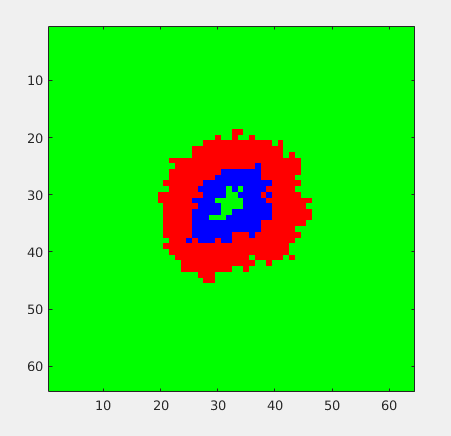


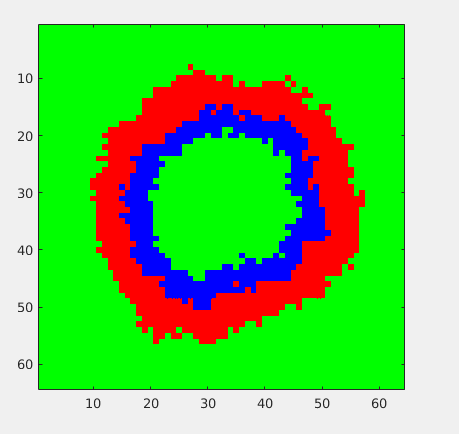




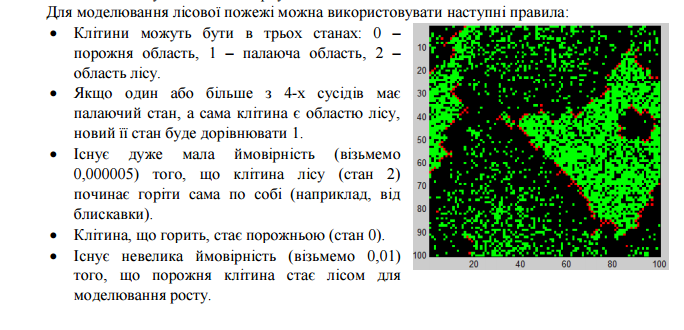
p = 0.3



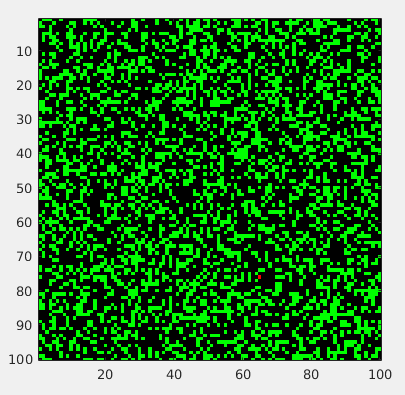


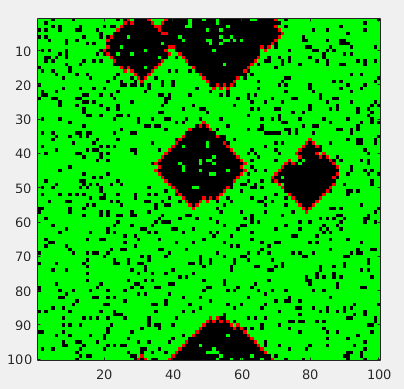


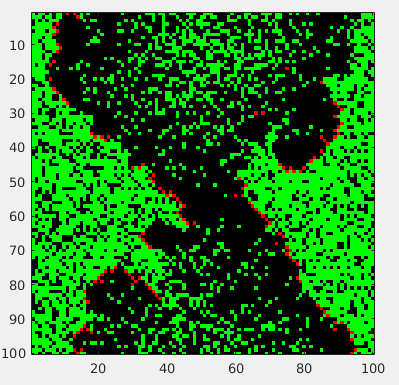
**Розповсюдження лісової пожежі**

****

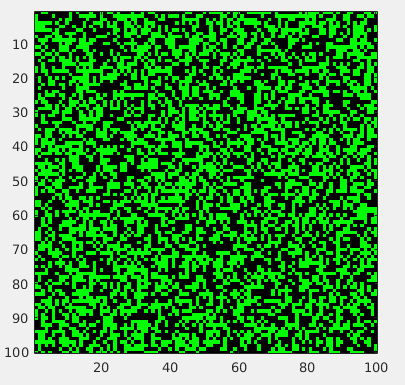
p = 0.000005

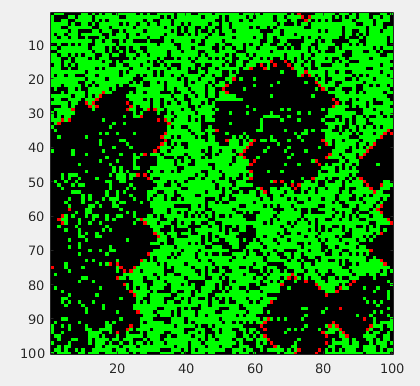


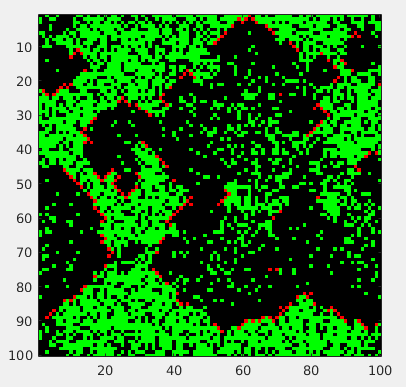




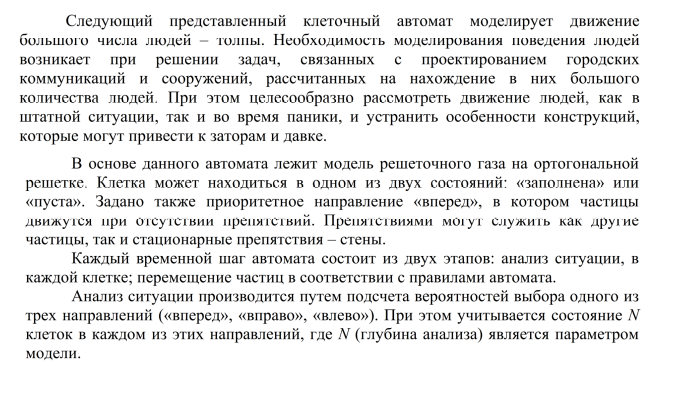
p = 0.000015

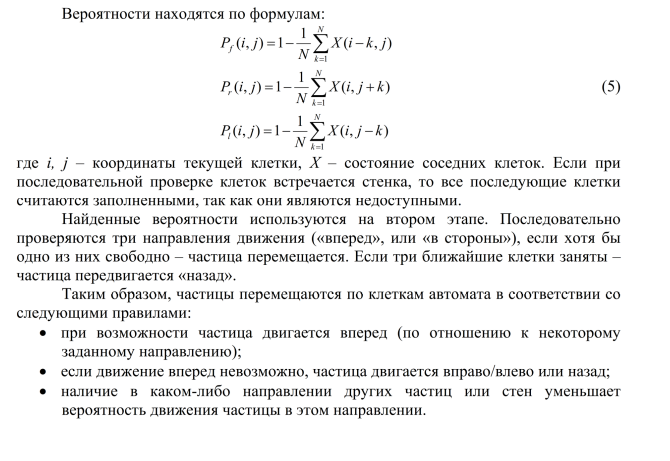


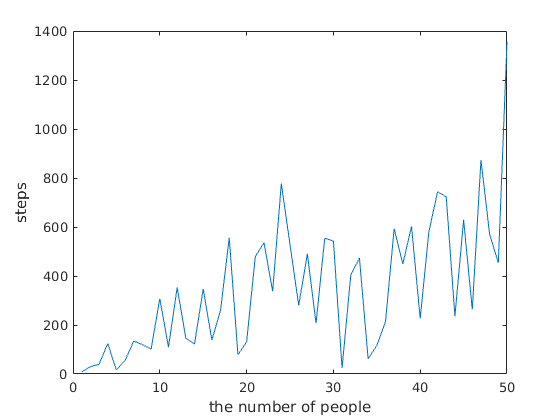




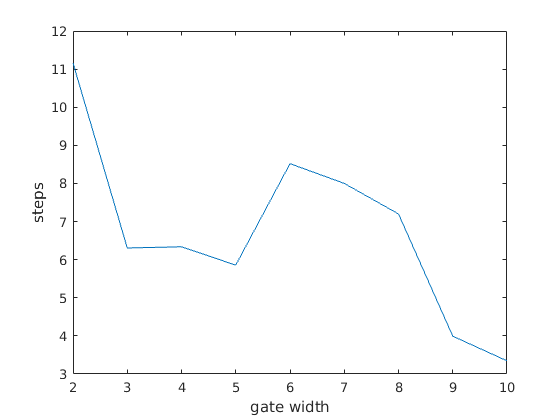
**Моделювання руху натовпу людей**

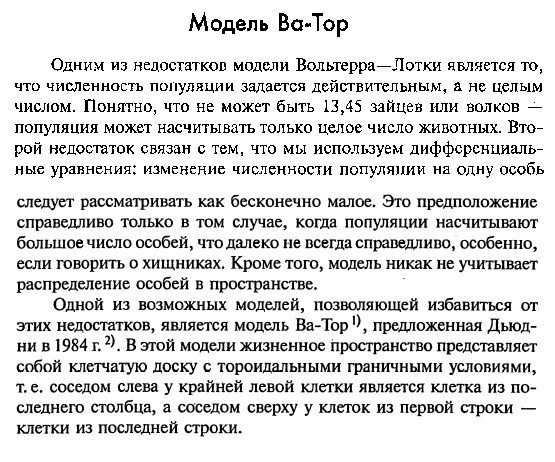
****

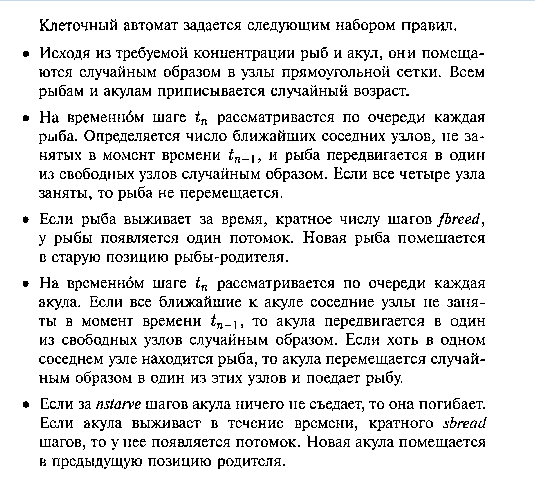
Графік залежності часу(кількості зроблених кроків) від кількості людей. Загалом час залежить від розташуванням чим ближче до прямої яка веде до виходу буде точка тим вище ії шанси на те, щоб вийти в інакшому випадку є можливість застрягнути у куртку и не мати змоги вийти. Тож іноді симуляція зупинялась так, як утворювался безкінечний цикл.



Залежність часу(кількості зроблених кроків) від довжини брами зпівпадає з очікуваннями. Так з збільшенням довжини кількість часу зменьшуется. Найкращим було б зробити два виходи хоча і маленьки, так час який потребуется на вихід з приміщення буде кращим.





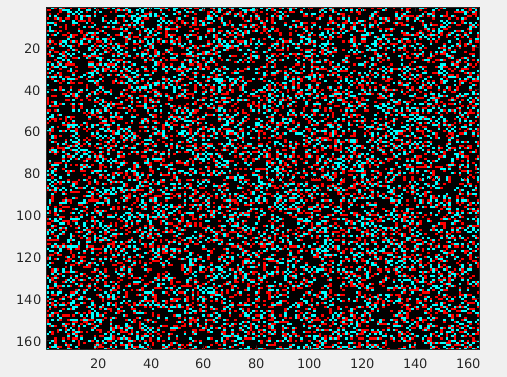
Експеремент

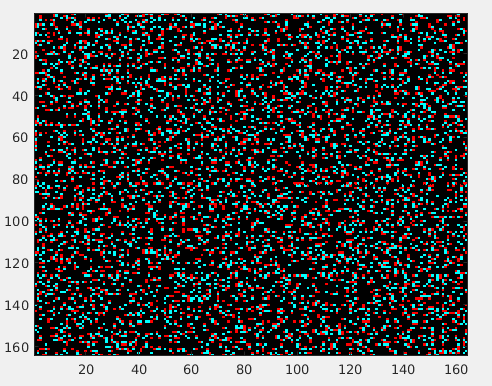
fbreed = 1;

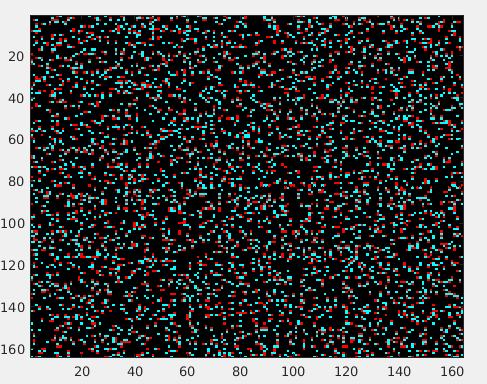
nstarve = 6;

sbreed = 1;

energy\_point = 4;







блакитним позначаються риби, червоним акули. За данних параметрів акули швидко розмножуются та ідять рибу од чого іі стає меньш.

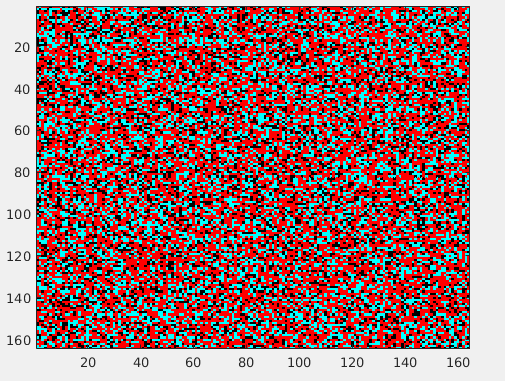
Експеремент 2

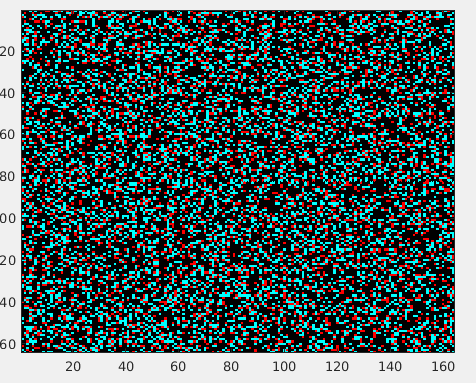
fbreed = 1;

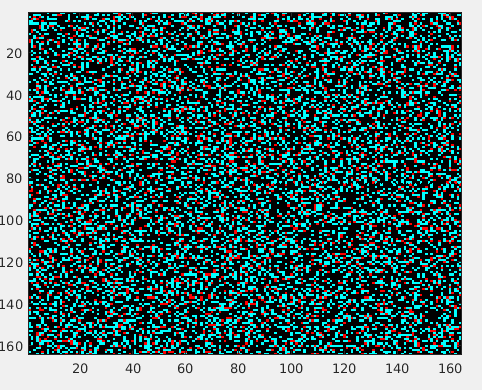
nstarve = 6;

sbreed = 6;

energy\_point = 7;







За даних параметрів акули вже не так швидко множаться дають шанс рибам збільшити свою популяцію.

Коди програм

life.m

% клеточный автомат "Жизнь"

clear all

clc

n = 64;

% инициализация масивов

z = zeros(n,n);

cells = z;

sum = z;

% задаем начальную конфигурацию

%cells(n/2,.25\*n:.75\*n) = 1;

%cells(.25\*n:.75\*n,n/2) =1 ;

cells(2, 3) = 1;

cells(3 ,4) = 1;

cells(4 ,2) = 1;

cells(4 ,3) = 1;

cells(4 ,4) = 1;

%cells = (rand(n,n))<.5;

% создаем картинку

% cat - объеденяет масив

imh = image(cat(3,z,cells,z));

set(imh,'erasemode', 'none');

axis equal

axis tight

% pause;

% задаем индексы внутрених точек

x= 2:n-1;

y= 2:n-1;

% проверяем правило

for i=1:100

sum(x,y) = cells(x, y-1) +cells(x,y+1)+...

cells(x-1, y) +cells(x+1,y)+...

cells(x-1, y-1)+cells(x-1,y+1)+...

cells(x+1,y-1)+cells(x+1,y+1);

cells = (sum==3)|(sum==2 & cells);

set(imh,'cdata',cat(3,z,cells,z))

drawnow

pause(.9);

end

forestFire.m

%

%forest fire

clf

clear all

n=100;

Plightning = .000015;

Pgrowth = .01; %.01

z=zeros(n,n);

o=ones(n,n);

veg=z;

sum=z;

imh = image(cat(3,z,veg\*.02,z));

set(imh, 'erasemode', 'none')

axis equal

axis tight

for i=1:3000

sum = (veg(1:n,[n 1:n-1])==1) + (veg(1:n,[2:n 1])==1) + ...

(veg([n 1:n-1], 1:n)==1) + (veg([2:n 1],1:n)==1) ;

veg = ...

2\*(veg==2) - ((veg==2) & (sum>0 | (rand(n,n)<Plightning))) + ...

2\*((veg==0) & rand(n,n)<Pgrowth) ;

set(imh, 'cdata', cat(3,(veg==1),(veg==2),z) )

drawnow

pause(0.1)

end

ringworm.m

clc

clear all

n=64;

old = zeros(n,n);

new = zeros(n,n);

count = zeros(n,n);

old(32, 32) = 1;

imh = image(cat(3,old,old,old));

set(imh, 'erasemode', 'none')

axis equal

axis tight

% 0 = health

% 1 = sick

% 2 = not sick

for k=1:100

for i=2:n-1

for j=2:n-1

if old(i,j)==1

if count(i,j) == 6

count(i,j) = 0;

new(i,j) = 2;

else

for x=i-1:i+1

for y=j-1:j+1

if rand()<0.3 && old(x,y)==0

new(x,y) = 1;

end

end

end

count(i,j)= count(i,j)+1;

end

end

if old(i,j)==2

if count(i,j) == 4

count(i,j) =0;

new(i,j)=0;

else

count(i,j)= count(i,j)+1;

end

end

end

end

set(imh, 'cdata', cat(3,(old==1),(old==0),(old==2)))

drawnow

old = new;

pause (0.9)

end

crowd\_sim.m

clc;clear all;

%init

n = 64;

old = ones(n,n);

%depth of field

d = 2;

%board

old(:,1) = zeros(n,1);

old(:,n) = old(:,1);

old(1,:) = old(:,1);

old(n,:) = old(:,1);

%gate

old(1,n/2-1:n/2+1) = 1;

%wall

old(n/2-5,n/2-3:n/2+3) = 0;

old(n/2+5,n/2-3:n/2+3) = 0;

% empty template

template = old;

%crowd

old(5,n-5) = 2;

old(n/2,n/2) = 2;

%old = round((n-5).\*rand(5,2)+5);

new = template;

imh = image(cat(3,old,old,old));

%main loop

while true

%analisys

%loop over all board

for i=1+d:n-d

for j=1+d:n-d

%if cell is human work with it

if old(i,j)==2

%calculate probability to walk up left right

pu = 1 - 1/d \* sum( old( i-d : i-1, j ) );

pl = 1 - 1/d \* sum( old( i, j-d : j-1 ) );

pr = 1 - 1/d \* sum( old( i, j+1 : j+d ) );

% chose max probability

[~,idx] = min([ pu, pl, pr]);

%make step

% f\_step = @(t)[cos(deg2rad(t)),sin(deg2rad(t));-sin(deg2rad(t)),cos(deg2rad(t))];

% step = round([0 1]\*f\_step(m\_idx\*270));

% new(i+step(1),j+step(2)) = 2;

if idx == 1

new(i-1,j) = 2;

elseif idx == 2

new(i,j-1) = 2;

elseif idx == 3

new(i,j+1) = 2;

end

end

end

end

% update old state

old = new;

% clear new

new = template;

%draw

set(imh,'cdata',cat(3,(old==2),old,old))

drawnow

pause(.9);

end

wa\_tor.m

clc;clear all;close all;

% init

N = 164;

old = zeros(N,N);

new = old;

f\_step = @(t)[cos(deg2rad(t)),sin(deg2rad(t));-sin(deg2rad(t)),cos(deg2rad(t))];

fbreed = 1;

nstarve = 6;

sbreed = 1;

energy\_point = 4;

% fbreed = 1;

% nstarve = 6;

% sbreed = 1;

% energy\_point = 4;

% add sharks and fishes with rand age

for i=1:N

for j=1:N

r = rand;

if r>0.8

old(i,j) = 0;

elseif r<=0.8 && r >=0.3

old(i,j) = -1;

else

old(i,j) = 1;

end

end

end

old(1,1) = -1;

old(1,2) = 1;

imh = image(cat(3,(old<0),(old>0),old));

pause(3.9);

while true

% for step =1:2

for i=1:N

for j=1:N

iup = [ (i-1<1)\*N+(i>=1)\*(i-1) j ];

iright = [ i (j+1>N)+(j+1<=N)\*(j+1)];

idown = [ (i+1>N)+(i+1<=N)\*(i+1) j ];

ileft = [ i (j-1<1)\*N+(j>=1)\*(j-1) ];

new\_idx = [ iup; iright; idown; ileft ];

% fishes move

if old(i,j)>0

% on new layer move fish to new position

up = (old(iup(1),iup(2))==0);

right = 2\*(old(iright(1),iright(2))==0);

down = 3\*(old(idown(1),idown(2))==0);

left = 4\*(old(ileft(1),ileft(2))==0);

pos = [up,right,down,left];

pos = find(pos~=0);

if ~isempty(pos)

%there is free space around

rand\_idx = round((length(pos)-1)\*rand + 1);

new(new\_idx(pos(rand\_idx), 1),new\_idx(pos(rand\_idx), 2)) = old(i,j) + 1;

% if old(i,j)/fbreed == 1 -> put on old fish place one child

if mod( new(new\_idx(pos(rand\_idx), 1),new\_idx(pos(rand\_idx), 2)),fbreed) > 1

new(i,j) = 1;

new(new\_idx(pos(rand\_idx), 1),new\_idx(pos(rand\_idx), 2)) = mod( new(new\_idx(pos(rand\_idx), 1),new\_idx(pos(rand\_idx), 2)),fbreed);

end

else

%there is no free scpace around

new(i,j) = old(i,j) + 1;

if mod( new(i,j),fbreed) > 1

new(i,j) = 1;

end

end

end

% sharks move

if old(i,j) < 0

% if fish is near the shark, shark moves on it place

% find fishes around shark

up = ( old(iup(1), iup(2)) > 0 );

right = 2\*( old(iright(1), iright(2)) > 0 );

down = 3\*( old(idown(1), idown(2)) > 0 );

left = 4\*( old(ileft(1), ileft(2)) > 0 );

% make array of around state

pos = [up,right,down,left];

% let stay only places where fish is

pos = find(pos~=0);

% check if any fish around

if ~isempty(pos)

% we have at least one fish around

% make random choise of future place

% shark defently not die from starve

rand\_idx = round((length(pos)-1)\*rand + 1);

new(new\_idx(pos(rand\_idx), 1),new\_idx(pos(rand\_idx), 2)) = old(i,j) - energy\_point;

% check if shark could produce breed

if abs(new(new\_idx(pos(rand\_idx), 1),new\_idx(pos(rand\_idx), 2))) > sbreed

if mod(abs(new(new\_idx(pos(rand\_idx), 1),new\_idx(pos(rand\_idx), 2))),sbreed) > 0

new(i,j) = -1;

new(new\_idx(pos(rand\_idx), 1),new\_idx(pos(rand\_idx), 2)) = mod(abs(new(new\_idx(pos(rand\_idx), 1),new\_idx(pos(rand\_idx), 2))),sbreed);

end

end

else

%check if shar still have energy point

old(i,j) = old(i,j) + 1;

if old(i,j) < 0

% there is no fish around, let find free place for

% shark

up = (old(iup(1),iup(2))~=0);

right = 2\*(old(iright(1),iright(2))~=0);

down = 3\*(old(idown(1),idown(2))~=0);

left = 4\*(old(ileft(1),ileft(2))~=0);

pos = [up,right,down,left];

pos = find(pos~=0);

% check if there are free places around shark

if ~isempty(pos)

% there are free places around shark

rand\_idx = round((length(pos)-1)\*rand + 1);

new(new\_idx(pos(rand\_idx), 1),new\_idx(pos(rand\_idx), 2)) = old(i,j) - 1;

else

% there is no free places around shark, it will

% stay on the same place

new(i,j) = old(i,j) - 1;

end

end

end

end

end

end

old = new;

new = zeros(N,N);

set(imh,'cdata',cat(3,(old<0),(old>0),old))

drawnow

pause(1.5);

end