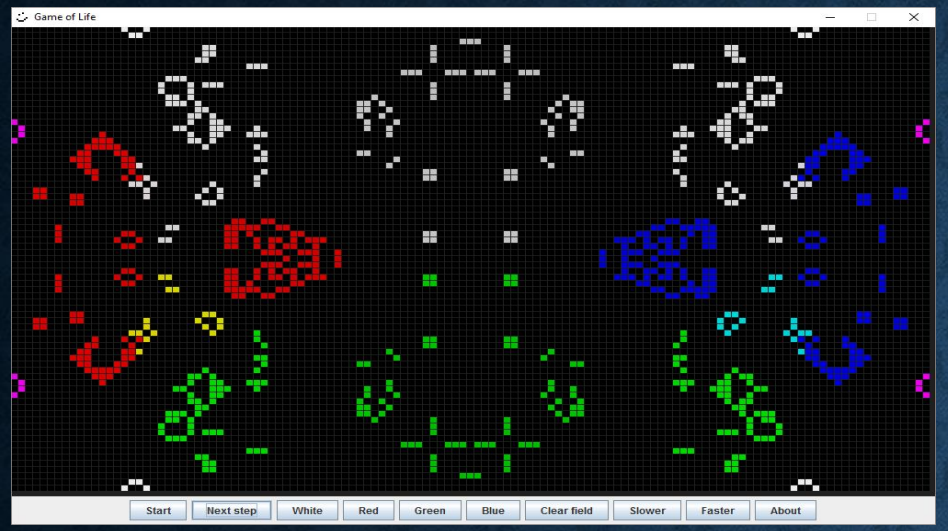
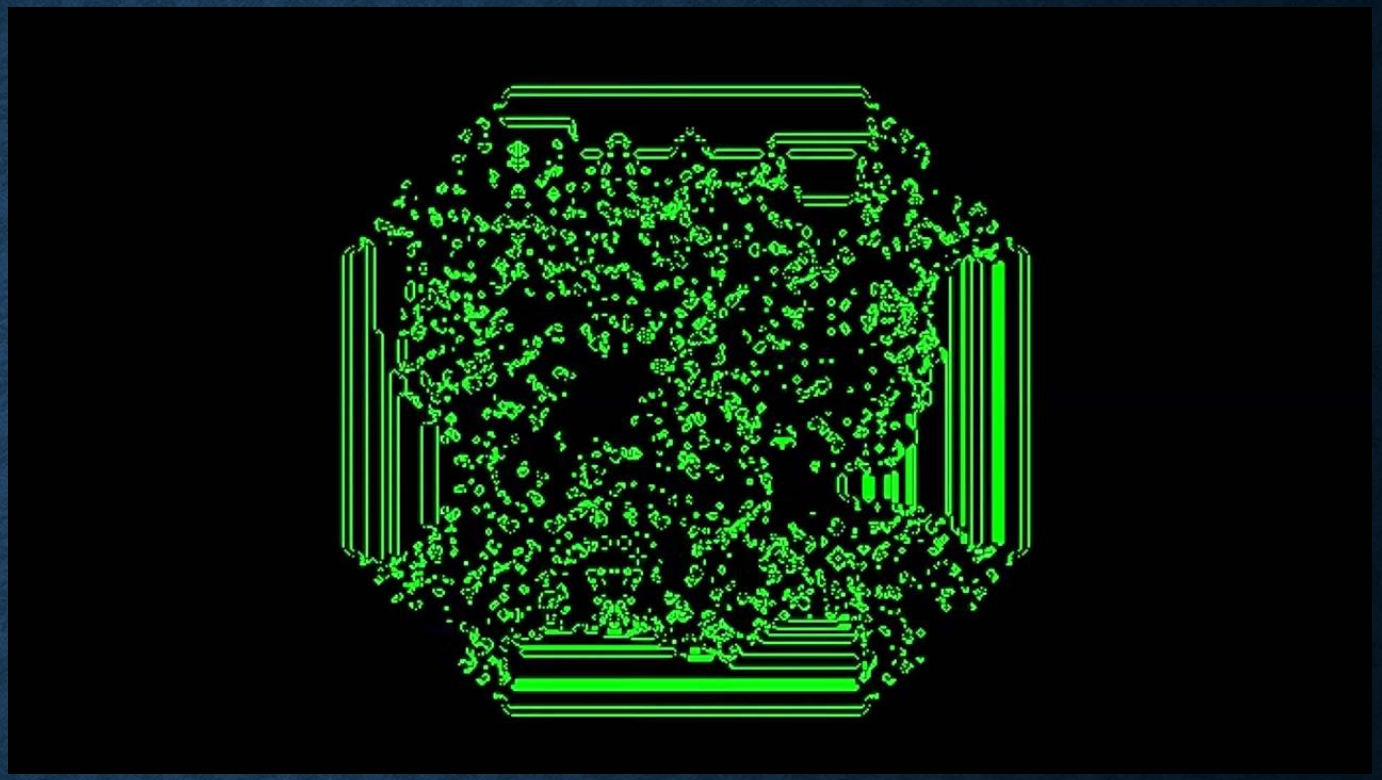
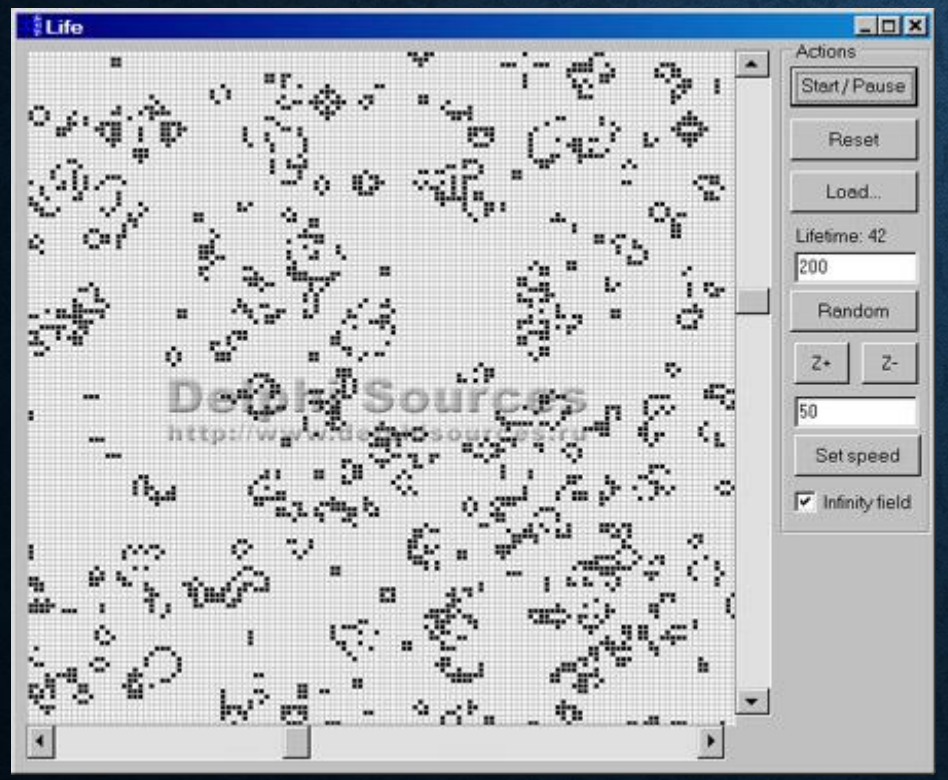
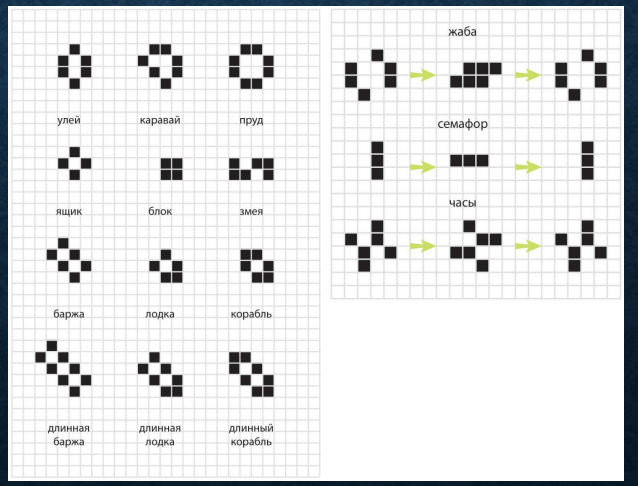
ИГРА ЖИЗНЬ. КЛЕТОЧНЫЕ АВТОМАТЫ.

Подготовил Фатеев Владислав. РИ-110942

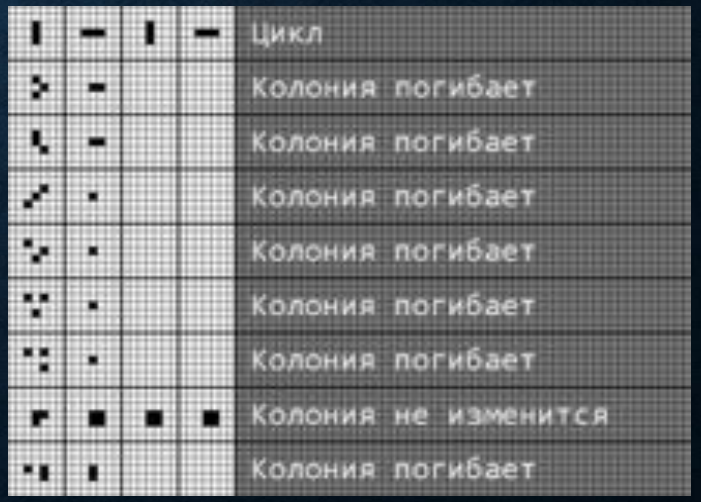

Game of Life
Start
Next step
White
Red
Green
Blue
Clear field
Slower
Faster
About




Life
Actions
Start / Pause
Reset
Load...
Lifetime: 42
200
Random
Z+
Z-
50
Set speed
Infinity field


улей
каравай
пруд
ящик
блок
змея
баржа
лодка
корабль
длинная баржа
длинная лодка
длинный корабль

жаба
семафор
часы


Цикл
Колония погибает
Колония погибает
Колония погибает
Колония погибает
Колония погибает
Колония погибает
Колония не изменится
Колония погибает

Модель Винера-Розенблюта

Выполнил: Долганов
Никита Сергеевич
АТ-01

Модель Винера-Розенблюта

Рассмотрим однородный ареал обитания некоторых травоядных животных

Пусть

- пищевых ресурсов для этих животных хватает на **время t1**
- На восстановление запасов пищи требуется **время t2**
- территория обитания – клетка со стороной 1
- каждая клетка может находиться в заселенном/ незаселенном/ возобновляемом **состоянии**

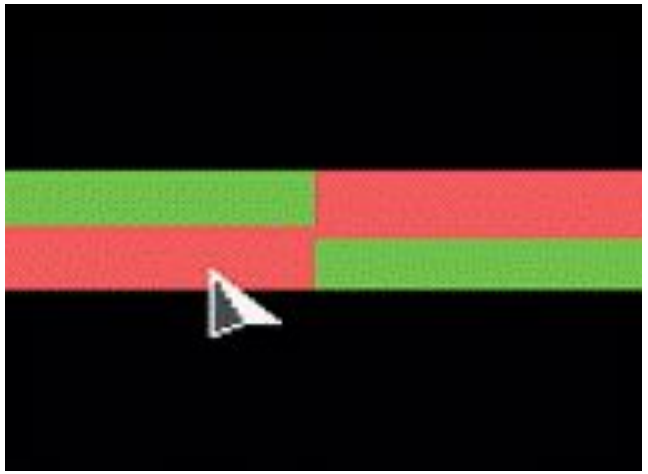
Правила

- Заселиться может только **незаселенная клетка**
- Через время t1 возбуждение клетки переходит в состояние **возобновления**
- Через время t2 возобновляемые клетки переходят в незаселенное состояние

Двухрукавная спиральная волна

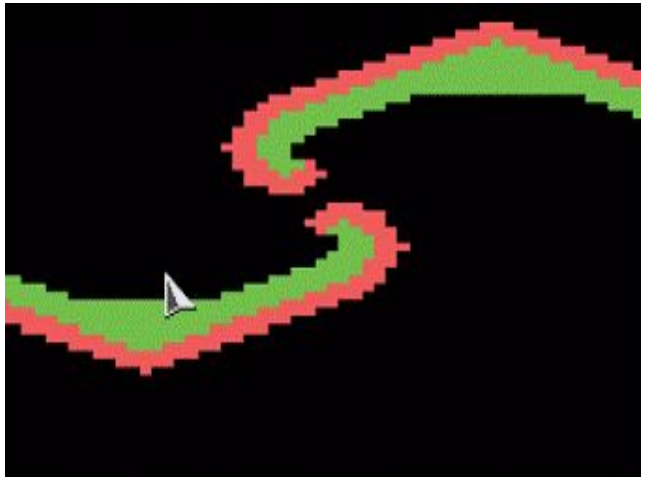
Если t1≈t2

Момент t = 1

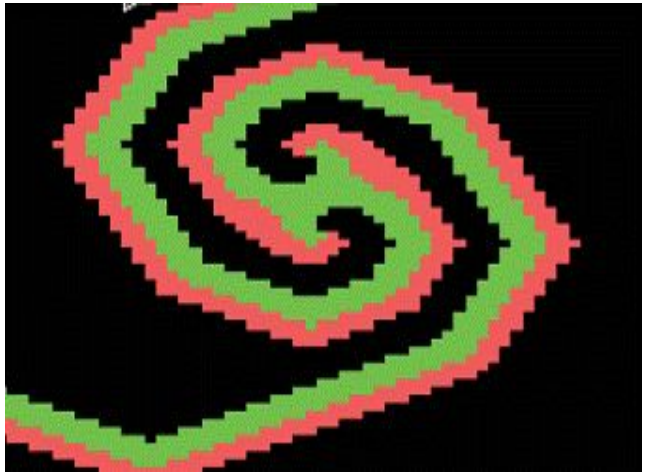


Принятое начальное состояние

Момент t = 21



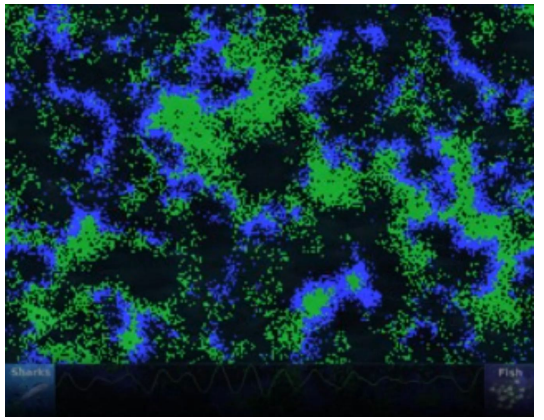
Момент t = 51



Модель Ва-тор

Выполнял Работу,студент Ат-01 группь
Воробьёв Алексей

- Модель Ва-Тор (от англ. Wa-Tor), предложенная Дьюдни в 1984 году, является клеточным автоматом, моделирующим сосуществование двух биологических видов – «хищников» и «жертв». Wa-Tor обычно реализуется в виде двухмерной сетки с тремя цветами: для рыб, для акул и для пустой воды. В этой модели жизненное пространство представляет собой клетчатую доску с тороидальными граничными условиями, то есть соседями справа у клеток крайнего правого столбца являются клетки крайнего левого столбца, а соседями сверху у клеток первой строки являются клетки последней строки.Акулы являются хищниками и едят рыбу. Как акулы, так и рыбы живут, передвигаются, размножаются и умирают в Ва-Торе в соответствии с

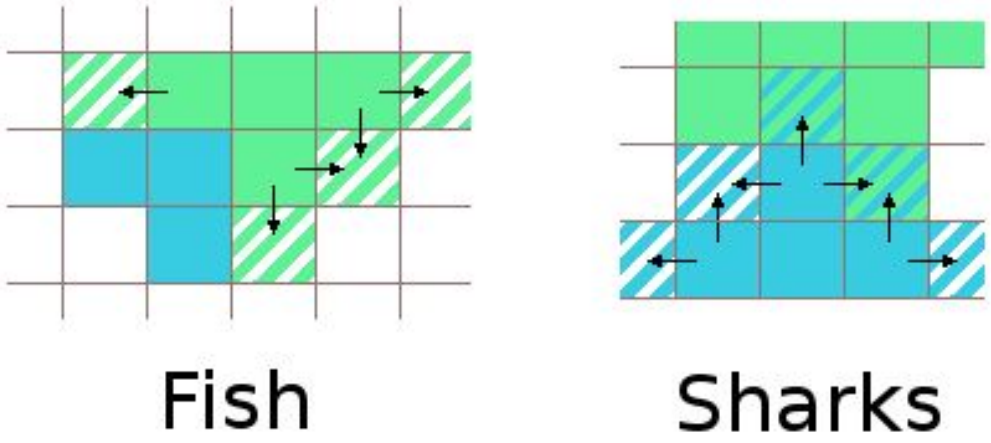


Выдержки из моделирования Ва-тор. Зеленым цветом показаны рыбы, синим акулы.

- Баланс этой экосистемы очень хрупок: популяции двух видов могут следовать сильно отличающимся друг от друга циклам в зависимости от заданных параметров (таких как циклы воспроизводства потомства и периода времени, в течение которого акула должна есть, чтобы избежать смерти), а также от начального положения каждого вида. Баланс может сильно меняться: от того, что оба вида находятся под угрозой исчезновения, до обилия одного или обоих.
- Когда добыча многочисленна, хищники могут быстро размножаться. Но это, в свою очередь, увеличивает количество добываемой добычи, и популяция жертвы уменьшается. Когда добыча становится меньше, хищники начинают голодать и умирать от голода, уменьшая свою популяцию и ослабляя давление на добычу. После этого жертва (а со временем и хищник) может вернуться к быстрому

Клеточный автомат задается следующим набором правил:

- Начальное количество рыб и акул помещается случайным образом в узлы прямоугольной сетки. Всем рыбам и акулам приписывается случайный возраст.
- На очередном временном шаге рассматривается по очереди каждая рыба. Определяется число ближайших незанятых соседних узлов и рыба передвигается в один из незанятых узлов случайным образом. Если все узлы заняты, рыба не перемещается.
- На очередном временном шаге рассматривается по очереди каждая акула. Если все ближайшие к акуле соседние узлы свободны, она перемещается в один из них случайным образом. Если хоть во одном из них находится рыба, акула перемещается в такой узел случайным образом и съедает рыбу.
- Если за N(a) шагов акула ничего не съедает, то она погибает. Если акула выживает в течение M(a) шагов, у нее появляется потомок. Новая акула помещается в предыдущую позицию родителя.
- Если рыба выживает в течение M(p) шагов, у нее появляется потомок. Новая рыба помещается в предыдущую позицию ро



Возможные результаты

В конечном итоге в Wa-Tor есть три возможных сценария:

- Идеальный баланс между рыбами и акулами, количество которых увеличивается и уменьшается, но никогда не исчезает.

Первый сценарий может быть очень трудным для реализации, когда достигается своего рода равновесие, при котором две популяции периодически колеблются в численности. В большинстве случаев количество рыбы сокращается до почти исчезающего состояния, затем популяция акул стремительно сокращается из-за нехватки корма. Это позволяет популяции рыб снова расти, пока популяция акул не сможет замедлить этот рост.

- Вымирание обоих видов.

Вымирание обоих видов происходит, когда количество акул превышает численность до такой степени, что они съедают всю рыбу. Поскольку рыба была единственным источником пищи для акул, они неминуемо умрут от голода.

- Исчезновение акул.

И наоборот, если первоначальное количество рыбы невелико или у акул очень короткий период голодания, реализуется второй сценарий. В этом случае акулы вымрут, оставив поле свободным для рыб.

Практическое применение

Для описания каждой рыбы программе необходимо запомнить три переменных: координаты x, y и возраст sday. Логично и очень удобно было бы каким-то образом сгруппировать эти три переменные. Для этого в языке С есть специальное средство - структуры.

```
struct fishtype {
int x,y;
int fday;
};
```

Этим определением мы ввели новый тип переменных, теперь можно описать переменные этого типа:

```
fishtype fish;
```

Группировка данных в структуры полезна тем, что когда нам нужно думать о рыбе как о целом, мы оперируем всей структурой. В то же время есть возможность работать и с полями структуры по отдельности, обращаясь к ним следующим образом: fish.x, fish.y, fish.fday.

Аналогичные определения можно сделать и для моделирования акулы, сгруппировав в структуру координаты акулы x и y, возраст акулы sday, и количество дней, прошедших после последней съеденной рыбы eat.

```
struct sharktype {
int x,y;
int fday;
int eat;
};
```

В данной ситуации, когда количество рыб и акул заранее неизвестно и может изменяться в очень широких пределах, удобнее распределять память компьютера динамически (по мере надобности в процессе выполнения программы). Поэтому вместо переменной типа sharktype удобнее описать переменную-указатель на этот тип:

```
sharktype *shark;
```

В этом случае обращение к полям структуры уже будет не через точку, а через стрелку: shark-> sday, shark->eat и т. д. Указатель занимает гораздо меньше памяти, чем сама структура, поэтому можно объявить большие массивы указателей с запасом. Необходимо только написать процедуры создания и удаления акул и рыб, не забывая аккуратно выделять и освобождать память для структур.

Листинг 5. Фрагмент программы моделирования клеточного автомата Ва-Тор fishtype *fish[32000]; // объявляем большой массив указателей на рыб. sharkty*shark pe [32000]; // и большой массив указателей на акул.

```
void MakeFish(int i,int x,int y) // процедура «рождения» новой рыбы
{
fish[i]=(FISH *)malloc(sizeof(FISH)); // выделяем память для размещения структуры
fish[i]->x=x; // помещаем рыбу в точку с заданными координатами
fish[i]->y=y;
fish[i]->fday=0; // ее возраст равен нулю
}
void MakeShark(int i,int x,int y) // процедура «рождения» акулы
{
shark[i]=(SHARK *)malloc(sizeof(SHARK)); // выделяем память для размещения
структуры
shark[i]->x=x; // помещаем акулу в точку с заданными
координатами
shark[i]->y=y;
shark[i]->sday=0; // возраст равен нулю
shark[i]->eat=0; // акула не ела 0 дней
}
void KillFish(int i) // «смерть» рыбы
{
free(fish[i]); // освобождаем память
fish[i]=NULL; // обнуляем значение указателя
}
void KillShark(int i) // «смерть» акулы
{
free(shark[i]); // освобождаем память
shark[i]=NULL; // обнуляем значение указателя
```