Review

- Recursion
- Factorial (Iterative and Recursive versions)
- Call Stack (Last-in, first-out Queue)
- Tracing recursive functions
- Fibonacci Sequence – Recursive Implementation
- Recursive Maze Generation

One can declare an array of any type

```
int myInt;          →   int[] myInts;
float myFloat;      →   float[] myFloats;
String myStr;       →   String[] myStrs;
```

… just add []

To create and size the array, use the `new` keyword

```
myInts = new int[10];
myFloats = new float[20]
myStrs = new String[30];
```

One can declare an array of custom classes

```
Mammoth[] mammoths;              // declare array variable

void setup() {
  mammoths = new Mammoth[30]; // create + size array
}

class Mammoth {
  String name;
  String sound;

  Mammoth( String name, String sound ) {
    this.name = name;
    this.sound = sound;
  }
}
```

If this is a float…

```
float myFloat;
```

and this is an array of floats…

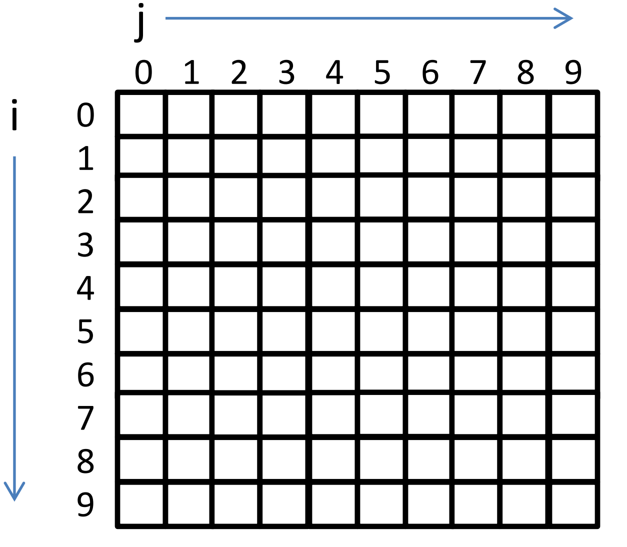
```
float[] myFloats;
```

what is this?

```
float[][] myFloats2;
```

Declare, size, and fill a 2D array

```
void setup() {

  float[][] myFloats2 = new float[10][10];

  for (int i=0; i<10; i++)
  {
    for (int j=0; j<10; j++)
    {
      myFloats2[i][j] = random(100);
    }
  }

}
```



ex1.pde

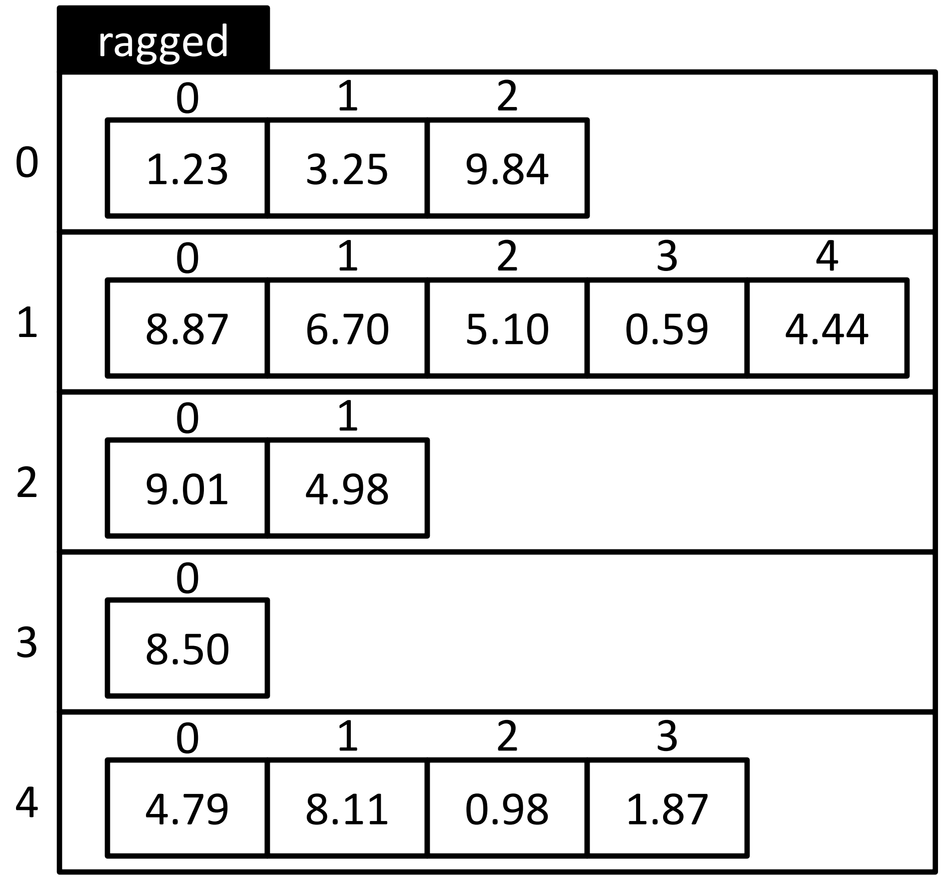
```
float[][] vals;

void setup() {

  vals = new float[20][300];

  for (int i=0; i<20; i++) {

    println( vals[i].length );   // What is going on here?

  }
}
```

```
300
300
300
300
300
300
```

ex2.pde

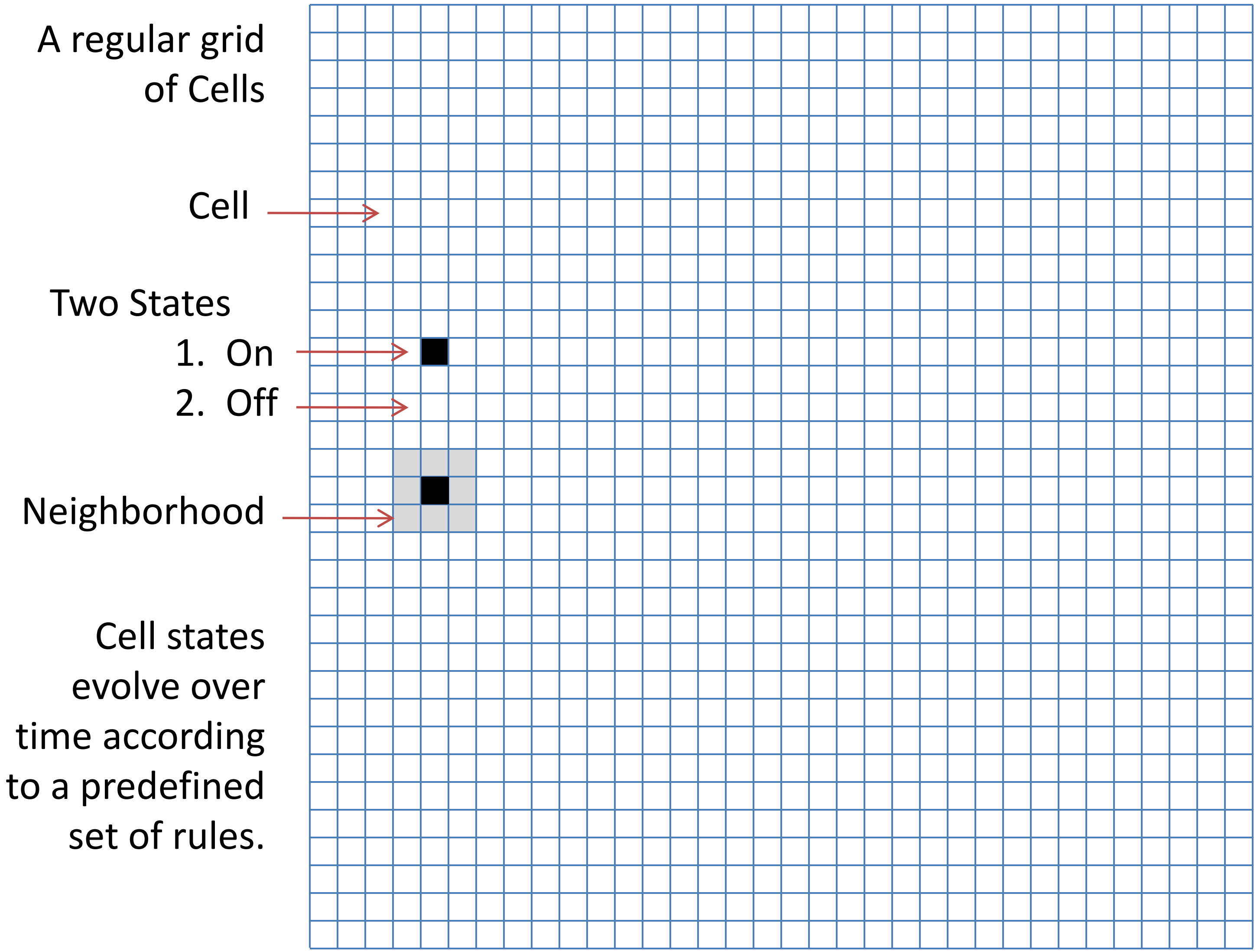
“Ragged” Arrays

```
float[][] ragged;

void setup() {

  ragged = new float[5][];

  for (int i=0; i<5; i++) {
    int n = int(random(10));
    ragged[i] = new float[n];
  }

  for (int i=0; i<5; i++) {
    println(ragged[i].length);
  }

}
```

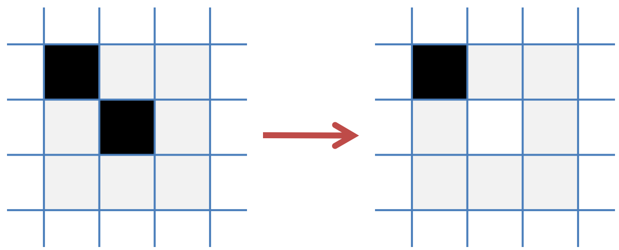


ex3.pde

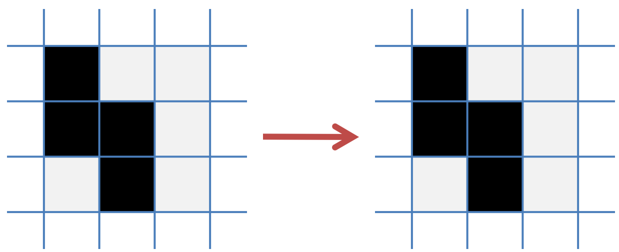
Cellular Automata



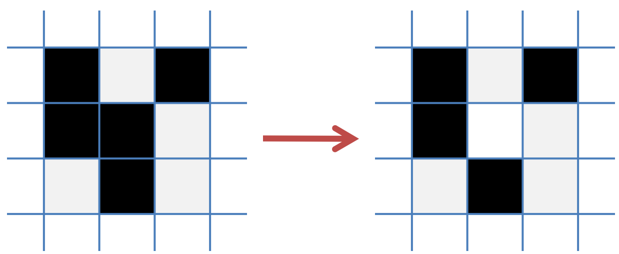
Sample Set of Rules – Conway's Game of Life



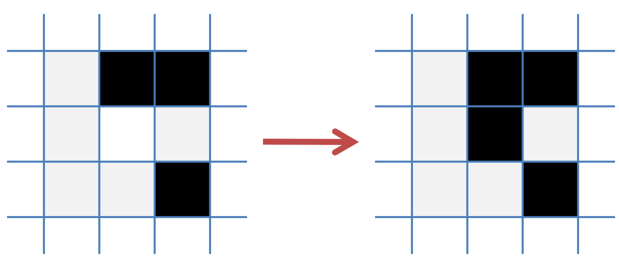
1. Any live cell with fewer than two live neighbors dies, as if caused by under-population.



2. Any live cell with two or three live neighbors lives on to the next generation.



3. Any live cell with more than three live neighbors dies, as if by overcrowding.

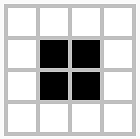
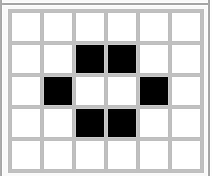
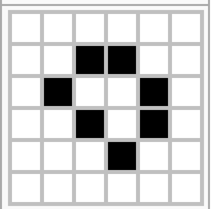
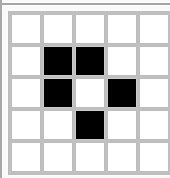


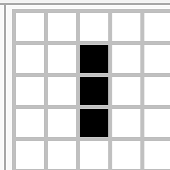
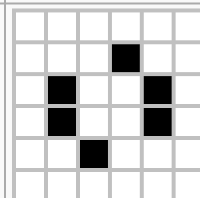
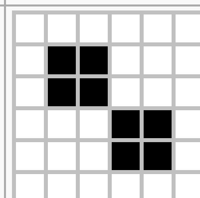
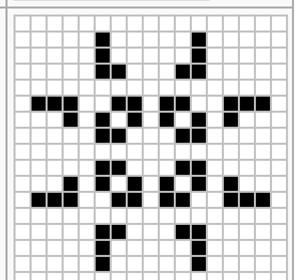
4. Any dead cell with exactly three live neighbors becomes a live cell, as if by reproduction.

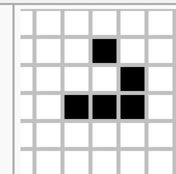
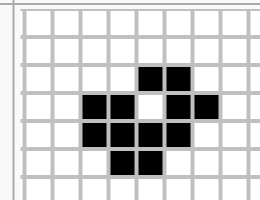
An example of "Emergence"

http://en.wikipedia.org/wiki/Conway%27s_game_of_life

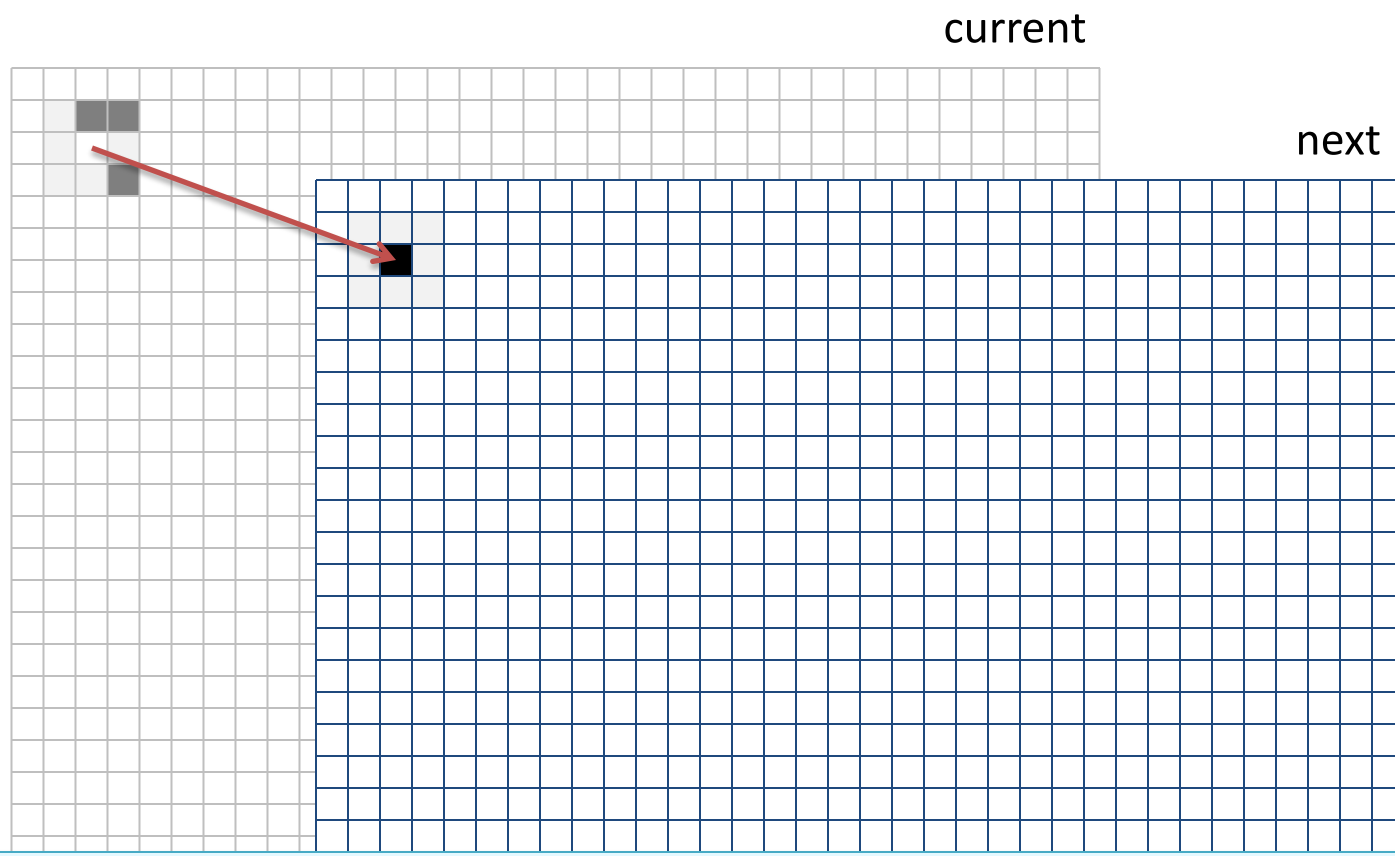
Interesting Patterns – Conway's Game of Life

Still lives	
Block	
Beehive	
Loaf	
Boat	

Oscillators	
Blinker (period 2)	
Toad (period 2)	
Beacon (period 2)	
Pulsar (period 3)	

Spaceships	
Glider	
Lightweight spaceship (LWSS)	

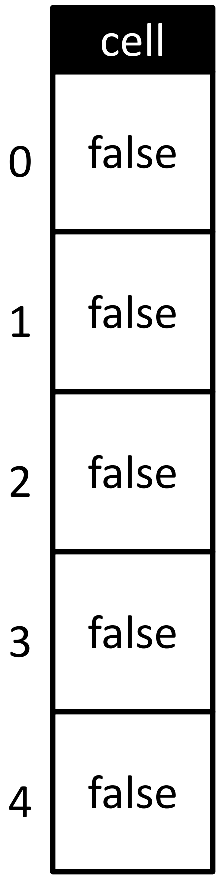
http://en.wikipedia.org/wiki/Conway%27s_game_of_life



Top-level procedure

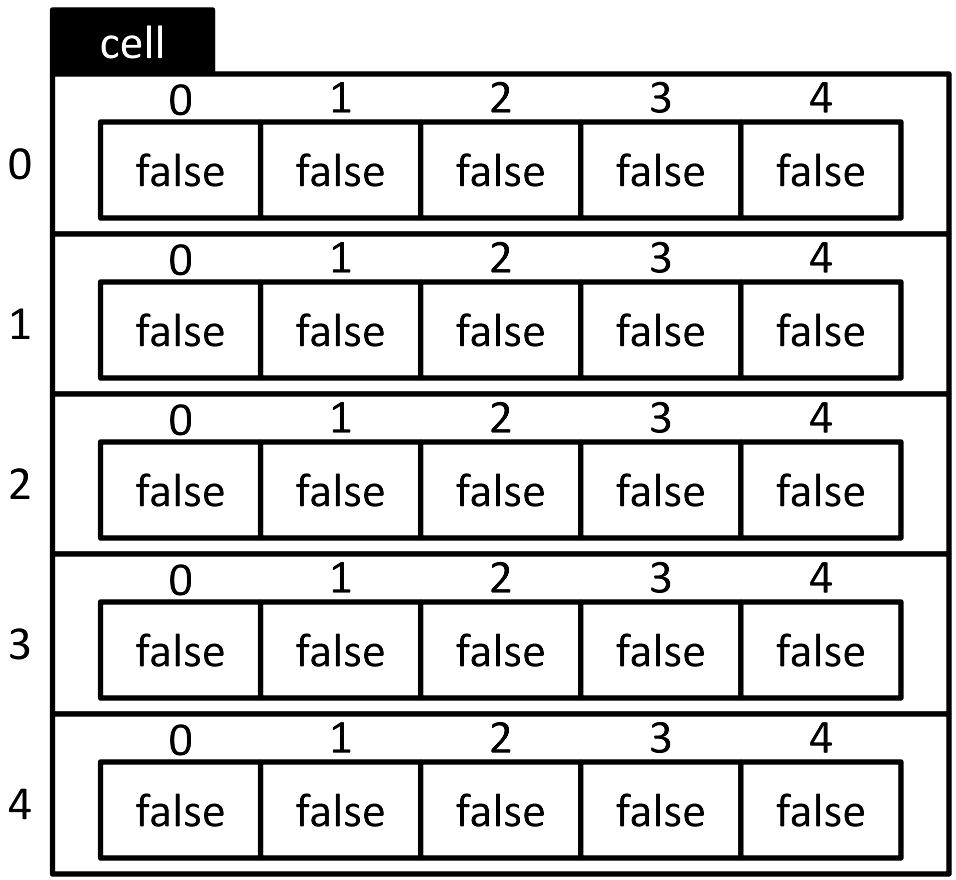
1. Draw the current grid
2. Advance game by applying rules to all cells of current and filling next
3. Swap current and next grid

```
int N = 5;
boolean[] cell = new boolean[N];
```



← One-dimensional array

```
int N = 5;
boolean[][] cell = new boolean[N][N];
```

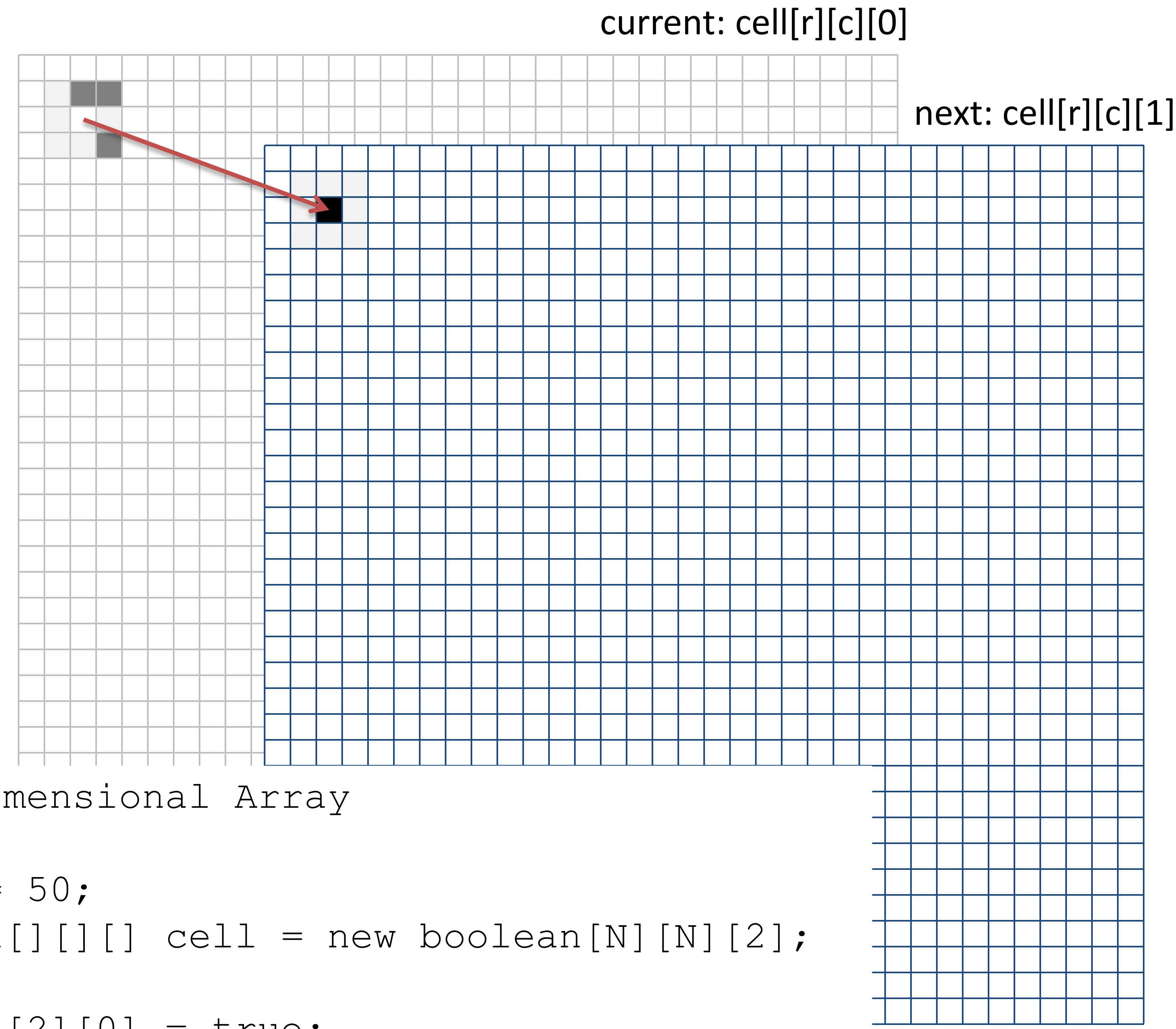


← Two-dimensional array

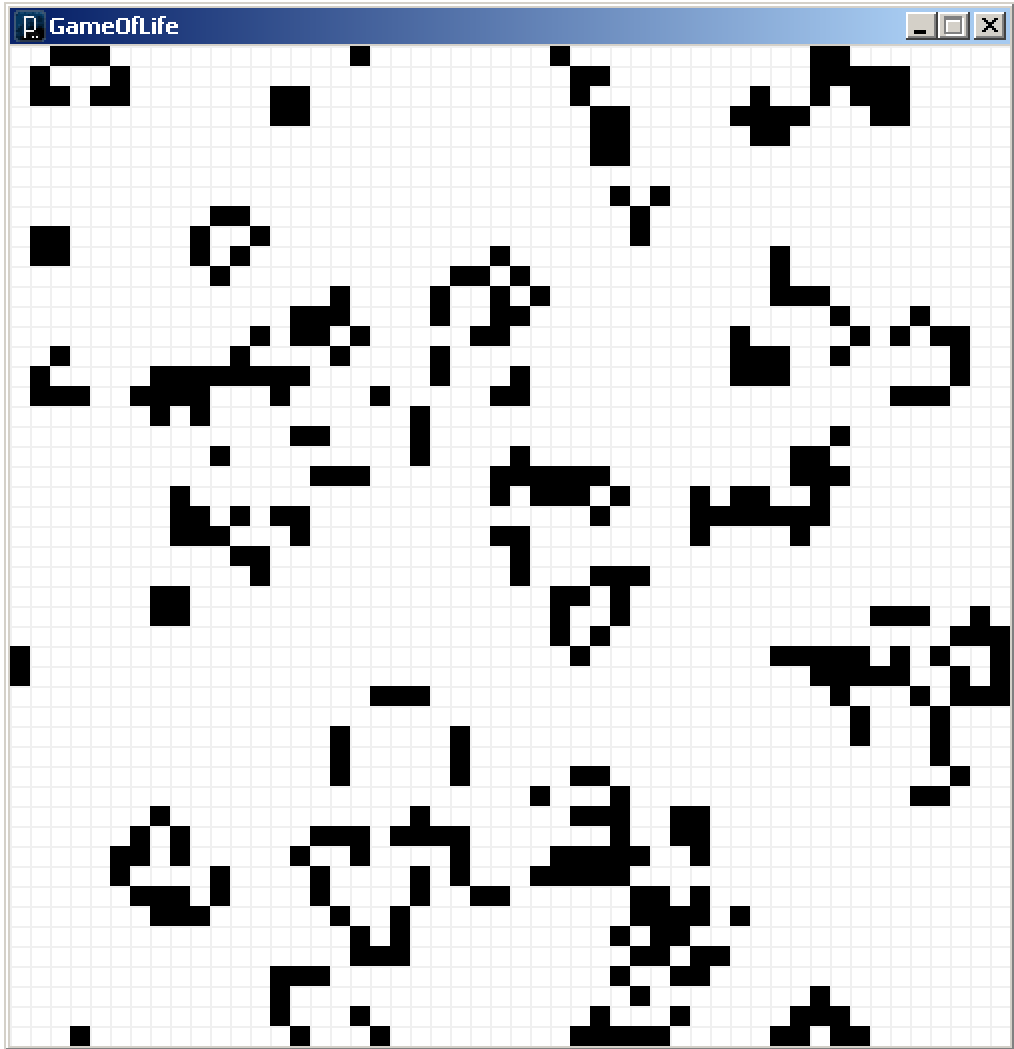
… an array of arrays

```
int N = 5;
boolean[][] cell = new boolean[N][N];

cell[1][2] = true;
```

cell	0	1	2	3	4
0	false	false	false	false	false
1	false	false	**true**	false	false
2	false	false	false	false	false
3	false	false	false	false	false
4	false	false	false	false	false




```
// 3-Dimensional Array

int N = 50;
boolean[][][] cell = new boolean[N][N][2];

cell[1][2][0] = true;
```


GameOfLife