Algorithms and Data Structures for Data Science

IEF and Multi-Dimensional Lists

CS 277

Brad Solomon

February 19, 2024

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Department of Computer Science

Illinois Interfaith Conference

Student-led conversations, dinner dialogues, interfaith service project, and more. Registration is free.

KEYNOTE SPEAKER

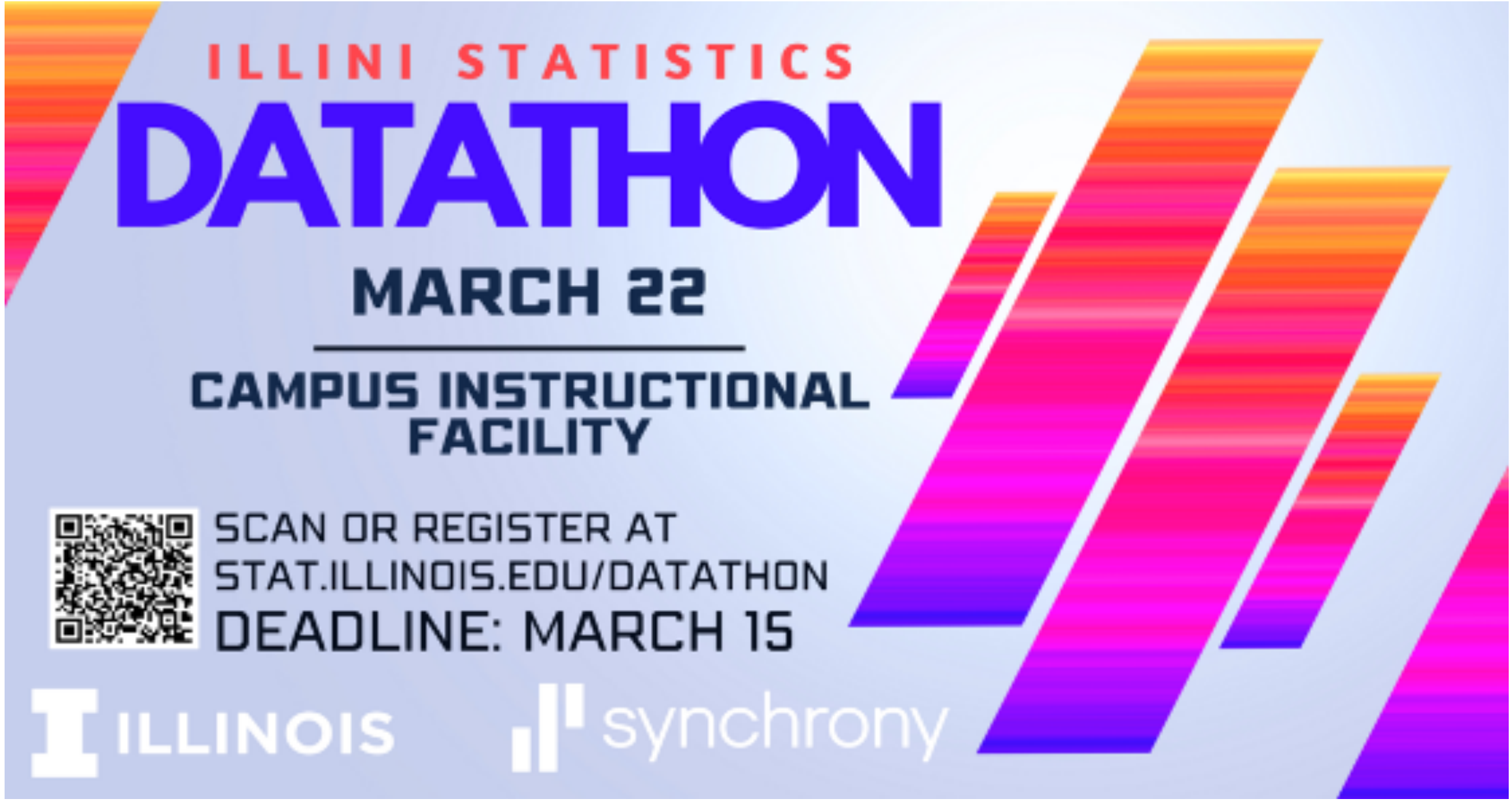
Nadia Bolz-Weber

Lutheran pastor and author of *Shameless: A Case for Not Feeling Bad About Feeling Good (About Sex)*

FEBRUARY 23 – 24, 2024

Co-sponsored by the Office of the Vice Chancellor for Diversity, Equity & Inclusion, Diversity & Social Justice Education, Illini Hillel, Interfaith in Action, the Department of Religion, Interfaith Alliance of Champaign County, Religious Workers Association, and the University YMCA

To learn more and register to attend, visit: go.illinois.edu/InterfaithConference.


ILLINI STATISTICS
DATATHON
MARCH 22
CAMPUS INSTRUCTIONAL FACILITY
SCAN OR REGISTER AT
STAT.ILLINOIS.EDU/DATATHON
DEADLINE: MARCH 15
ILLINOIS
synchrony

Informal Early Feedback

An anonymous survey about the class

If 70% of class completes, everyone gets bonus points

Please provide constructive criticism and positive feedback

Learning Objectives

Build a better understanding of course using IEF

Review exam

Review core 2D list concepts necessary for mp_automata

If time: Practice Big O in the context of sorting algorithms!

CS 277

An intermediate CS class that must cover:

- Algorithms Analysis and Optimization
- Elementary Data Structures

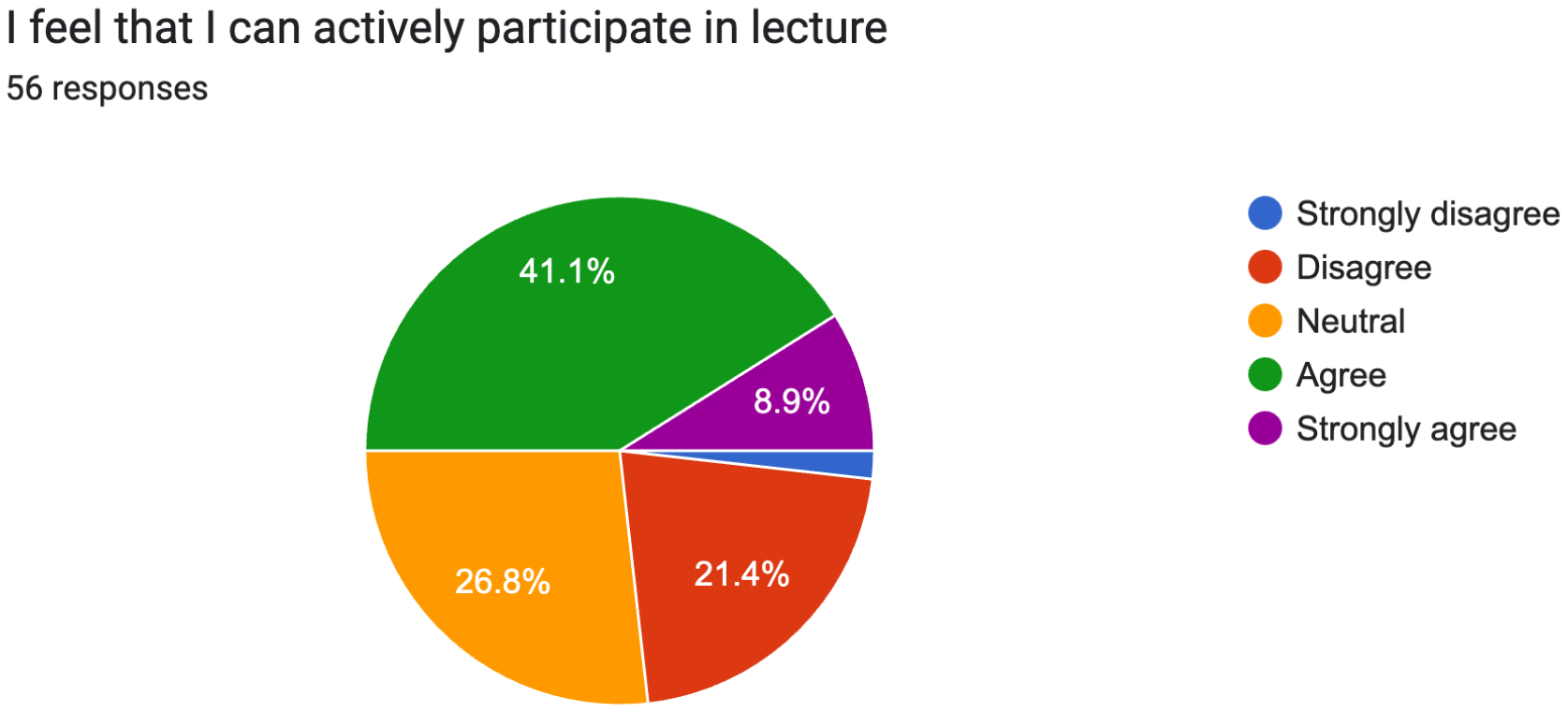
An introductory programming class that covers:

- Computational problem solving
- Programming techniques

CS 277 is relatively new and still finding its 'place' in the curriculum

IEF Lecture Feedback

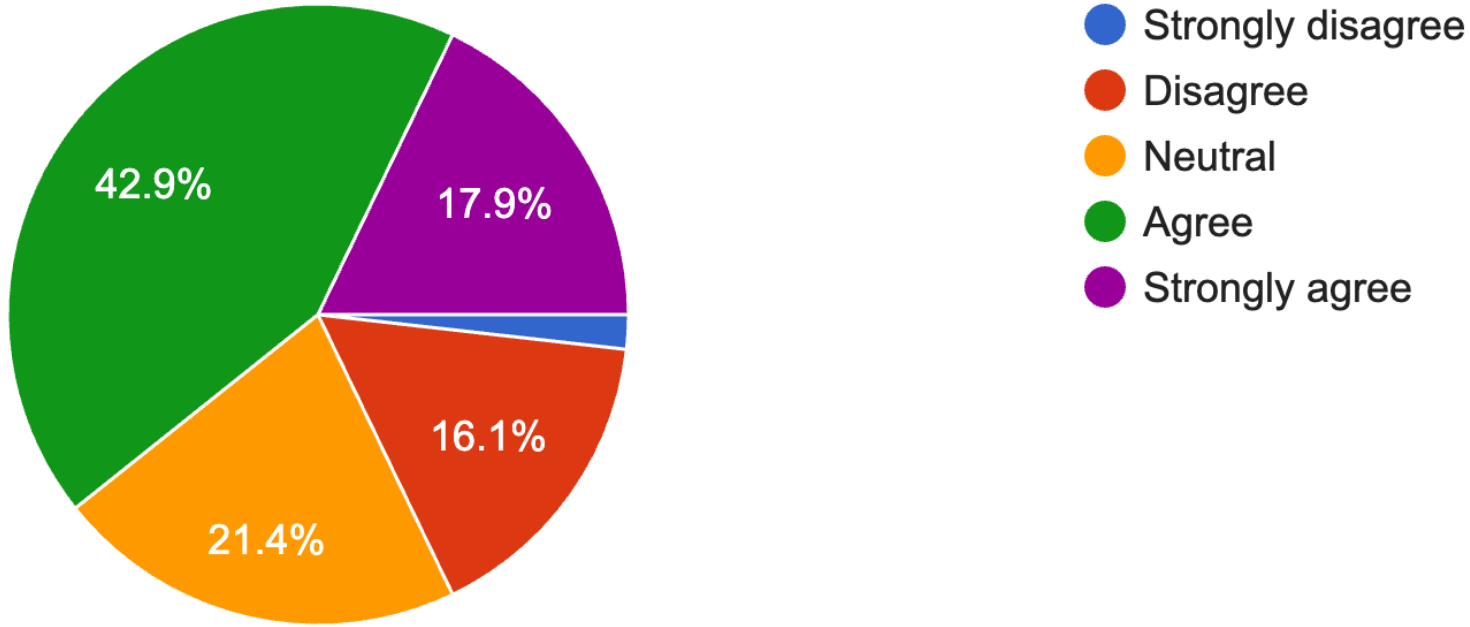
~22% feel they cannot actively participate in lecture **— what would help?**



IEF Lab Feedback

The lab lectures are helpful for completing the assignments.

56 responses

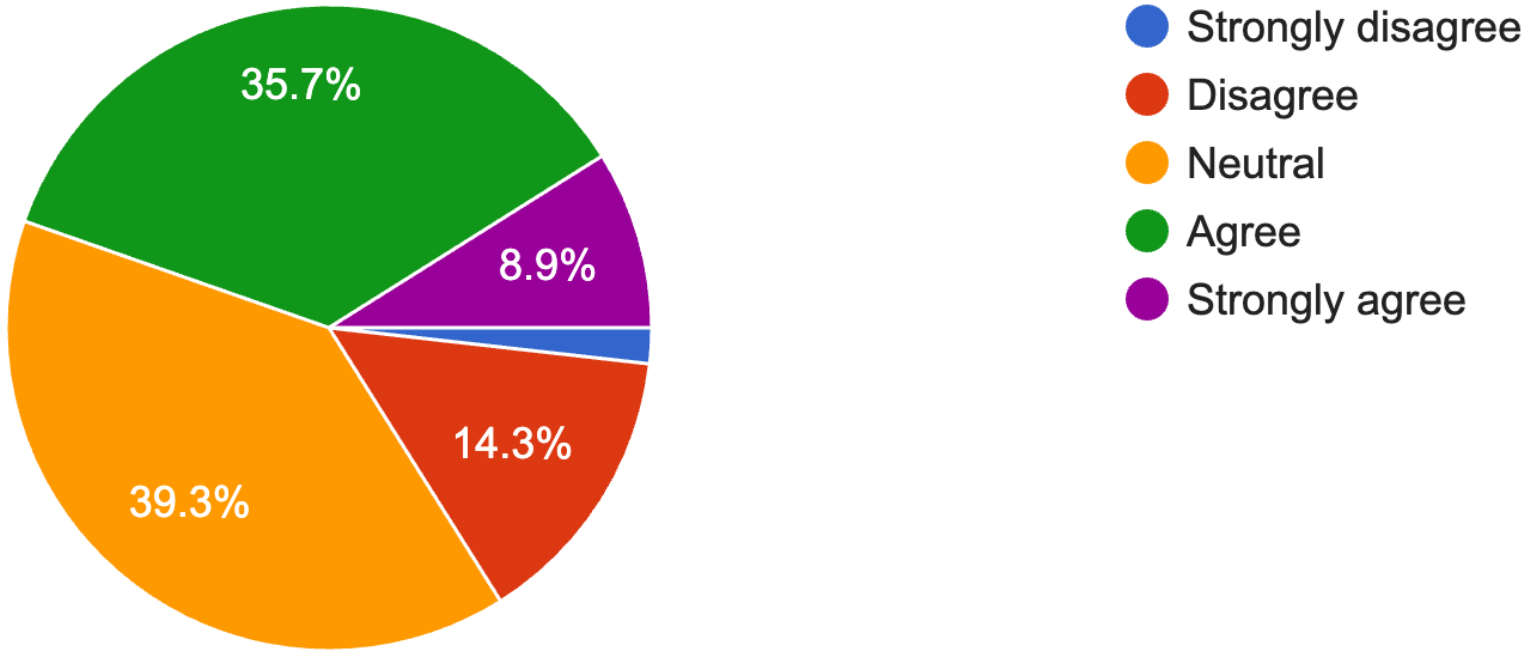


IEF Lab Feedback

Lets try regrouping as a class at the end of a lab section!

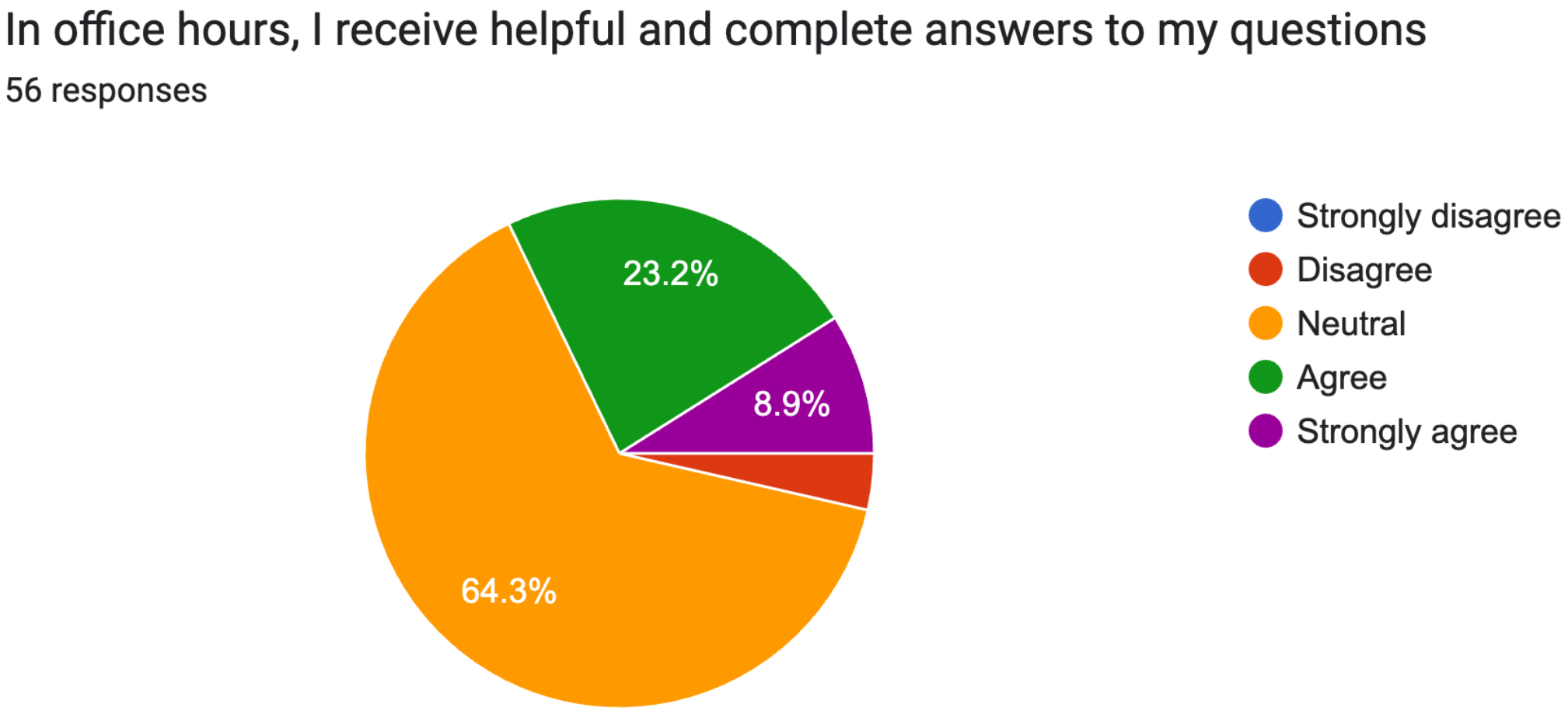
During lab, I receive helpful and complete answers to my questions

56 responses



IEF Course Resource Feedback

Every question looks roughly like this — **resource underutilized?**



IEF Short Answers (Positives)

People find both labs and MPs to be great sources of learning

People enjoy having office hour coverage throughout the week

People find the class is improving their coding and problem solving

(Some) people think the class is great and shouldn't be changed!

IEF Short Answers (Requests)

I wish there was more practice questions

I wish there was more in-class coding

Slow down the pace of the course

More extra credit assignments for advanced work

Requests for additional hints and help on assignments

IEF Short Answers (More Requests)

People want the annotated slides / annotated Python notebooks

More time for lab assignments

People want questions that look like the MPs in the lectures

IEF Short Answers (Negatives)

People don’t find the theory content relevant (prefer pure code)

People find the theory content great but dont like coding

People find some directions or lectures to be confusing

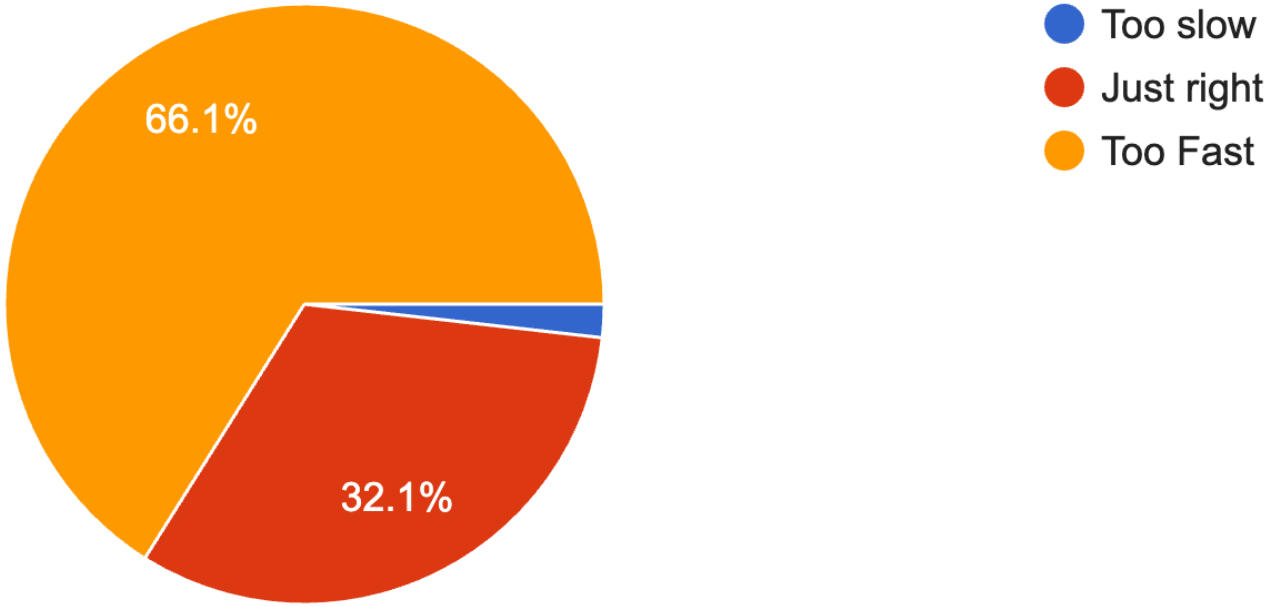
Concerns that the course is splitting focus too much

There is a steep learning curve in how to code

IEF Learning Objectives

The pacing of this course is

56 responses



Exam 0: Extra Credit Practice

We will go over two problems today (one from each set).

The other will be extra credit for one week.

PirateMap

Core concepts:

String Parsing

Data type Casting

Conditionals

Sub-problems:

Store x, y coordinates starting at 0, 0

Convert string to modification of variable

Get difference between two numbers

Similar to:

lab_fundamentals (getTotalTime(), electricBill())

Practice exam (Combine two lists)

Major Confusion: No clear signal

CalendarTime

Core concepts:

- String Parsing
- Data type Casting
- Conditionals

Sub-problems:

- Convert string to day and month
- Convert months to days
- Get difference between two numbers

Similar to:

- lab_fundamentals (getTotalTime(), electricBill())
- Practice exam (Combine two lists)

Major Confusion: Converting month to days

roboBuilder

Core concepts:

- 1D List Indexing
- Loops
- Conditionals

Sub-problems:

- Loop through recipe list
- Calculate price for recipe
- Update price lists

Similar to:

- lab_fundamentals (checkSorted(), removeOdds())
- mp_generate (Every list loop function in part 2)
- Practice exam (Combine two lists)

Major Confusion: List indexing

combineLists

Core concepts:

- 1D List Indexing
- Loops
- Conditionals

Sub-problems:

- Create string of appropriate format
- Iterate through lists by tracking index values
- Compute max of two integers

Similar to:

- lab_fundamentals (checkSorted(), removeOdds())
- mp_generate (Every generate function)
- Practice exam (Combine two lists)

Major Confusion: List indexing

Python List Indexing (and slicing)

Lets remind ourselves what the following code is actually doing!

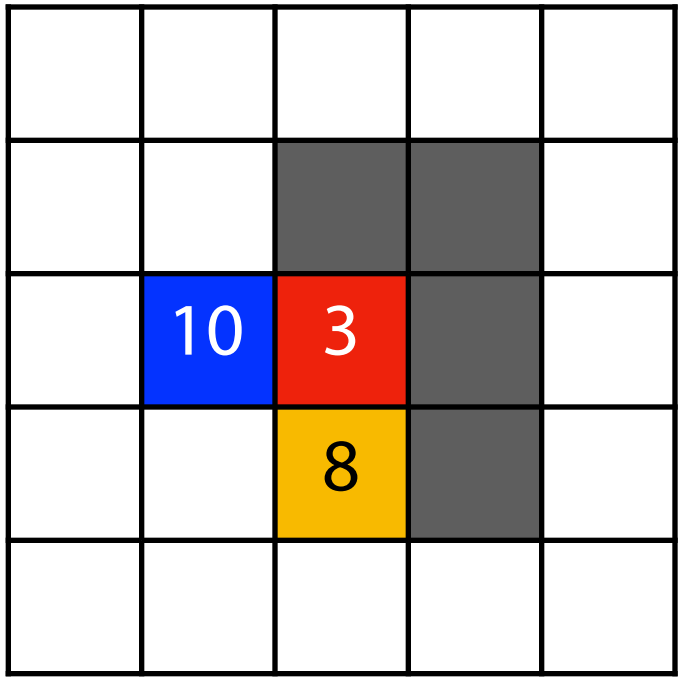
```
1  l = [5, 6, 7, 8]
2
3  # List indexing
4  for i in range(len(l)):
5      print(l[i])
6
7  i = 0
8  while(i < len(l)):
9      print(l[i])
10     i+=1
11
12 # List slicing
13 print(l[1:2])
14
15
16
17
18
```

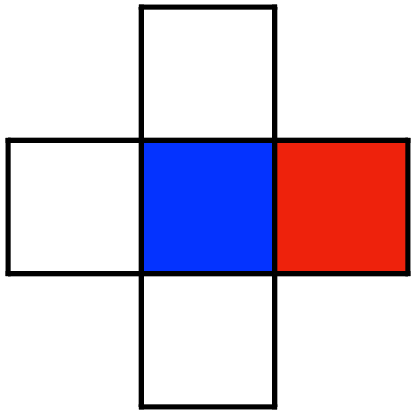
The Flood Fill Cellular Automata

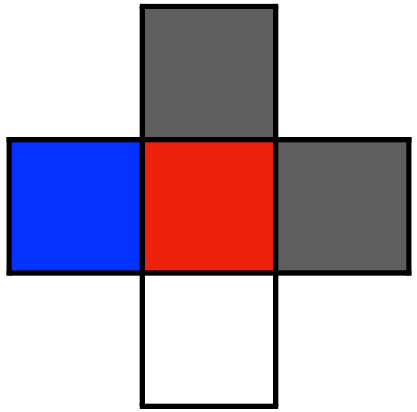
For each cell in the matrix, spread water evenly between all neighbors

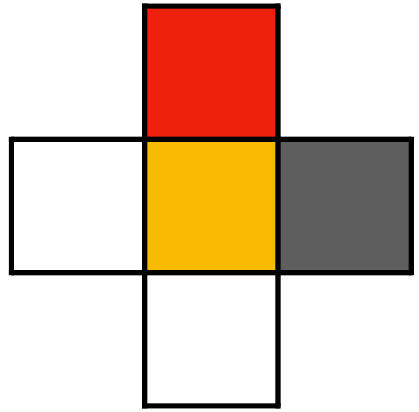
The key CA trick: Each square calculates simultaneously.

One cell perspective: Red's water is entirely based on three squares







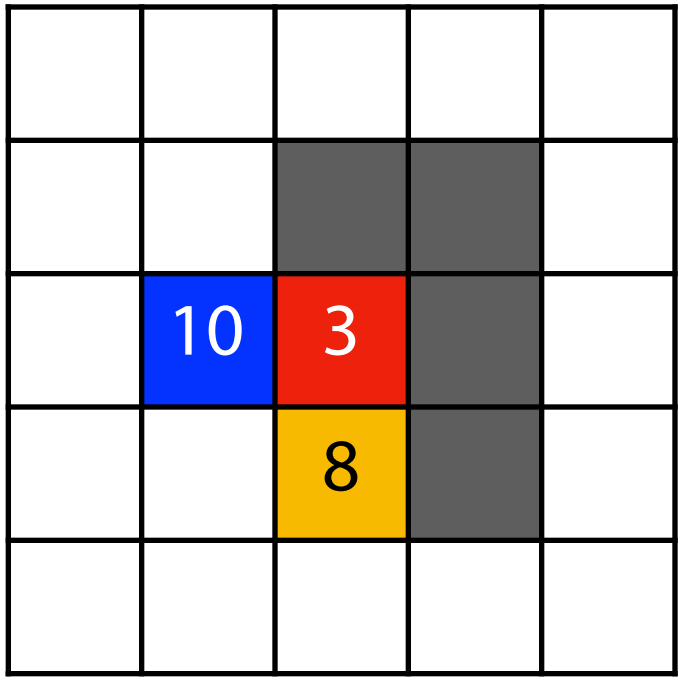


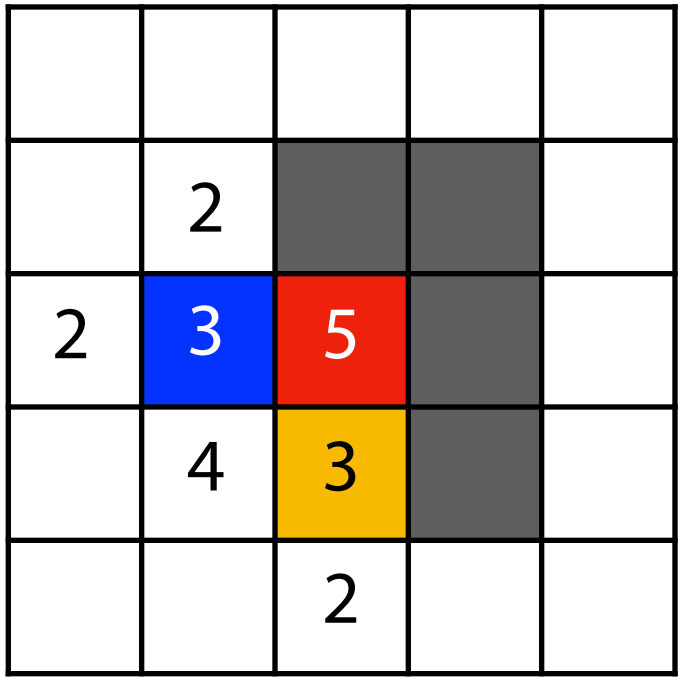
The Flood Fill Cellular Automata

For each cell in the matrix, spread water evenly between all neighbors

The key CA trick: Each square calculates simultaneously.

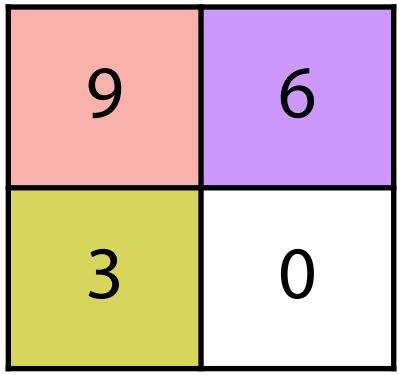
Water will diffuse to all nearby cells (and eventually stabilize)



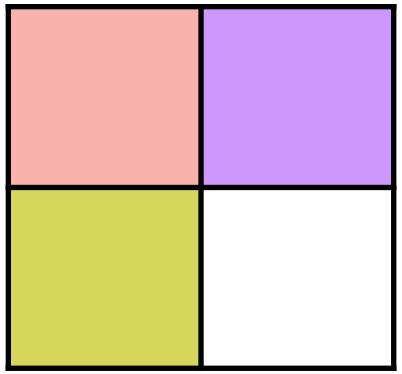


The Flood Fill Cellular Automata

`ff_update(matrix)`

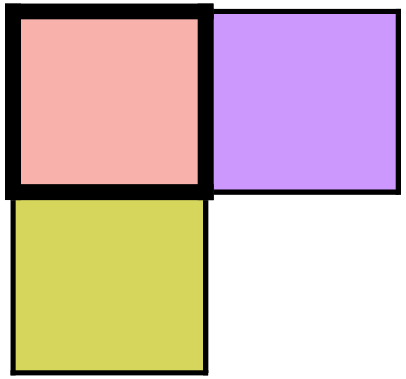


t=0

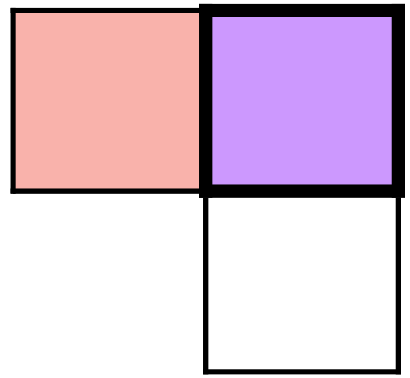


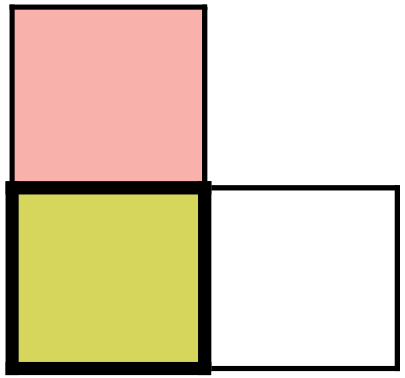
t=1

9 weight

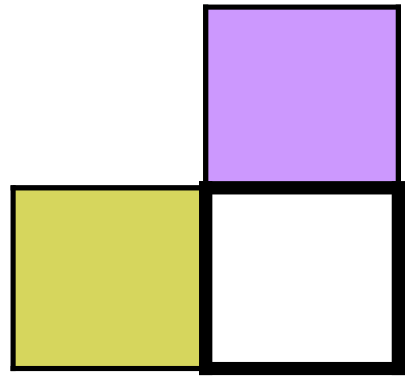


6 weight





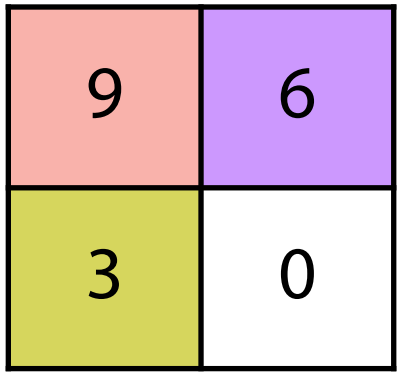
3 weight



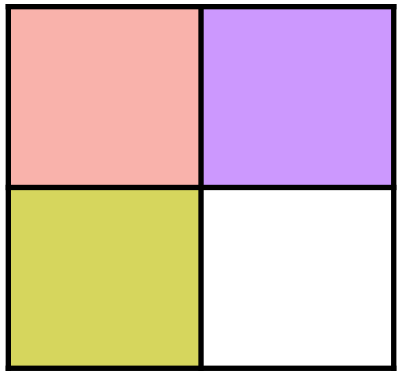
0 weight

The Flood Fill Cellular Automata

`ff_update(matrix)`

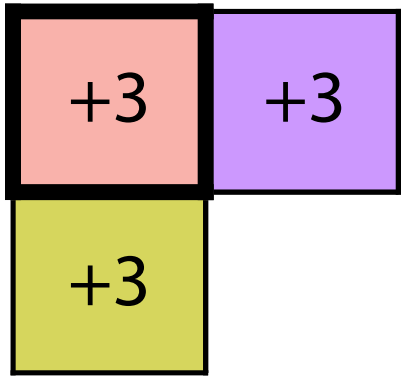


t=0

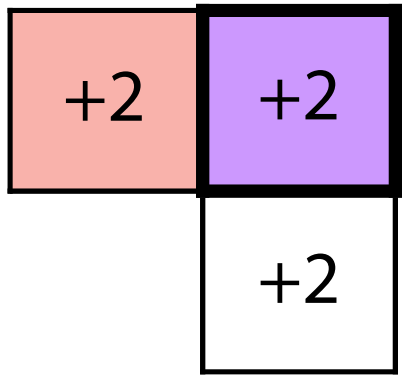


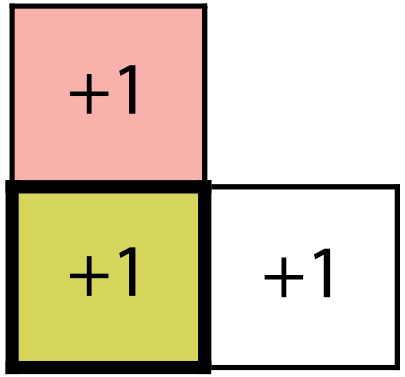
t=1

9 weight

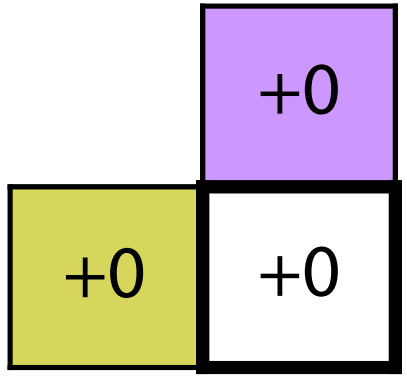


6 weight



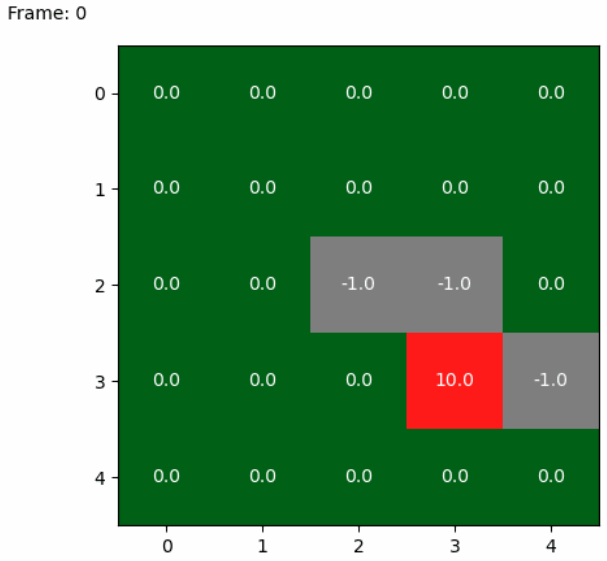


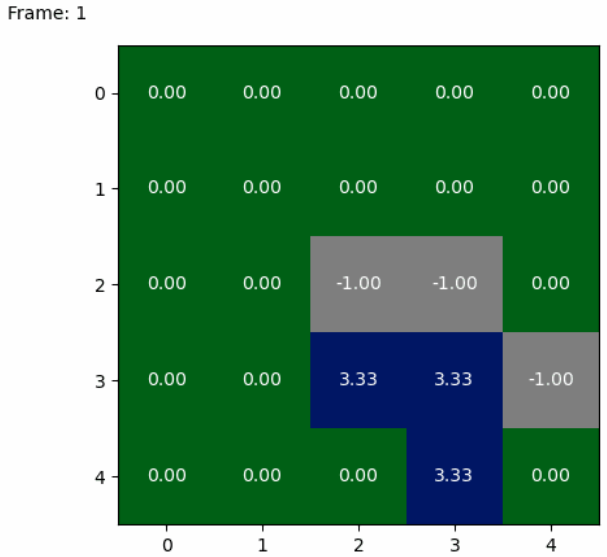
3 weight

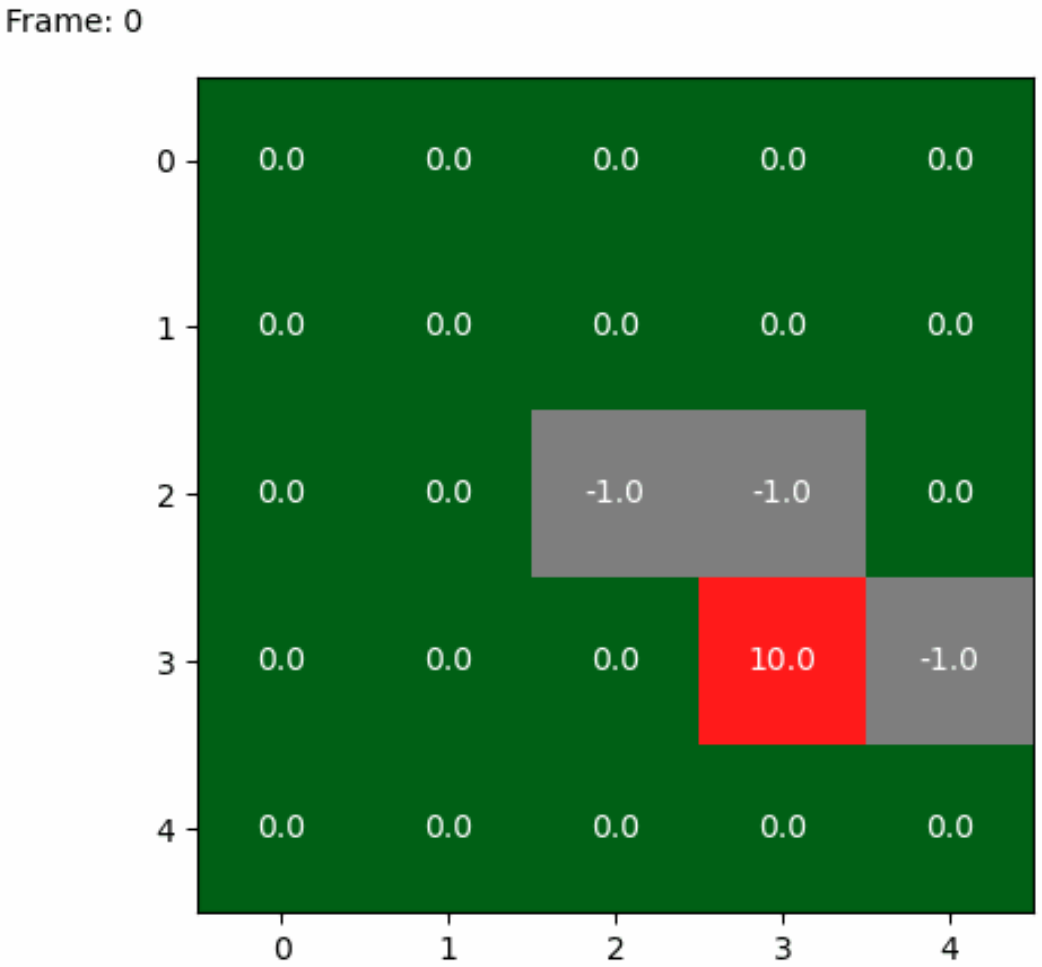


0 weight

The Flood Fill Cellular Automata







Conway's Game of Life Rules

All cells in a matrix update at the same time according to the following:

1. Any live cell with fewer than two live neighbors dies.

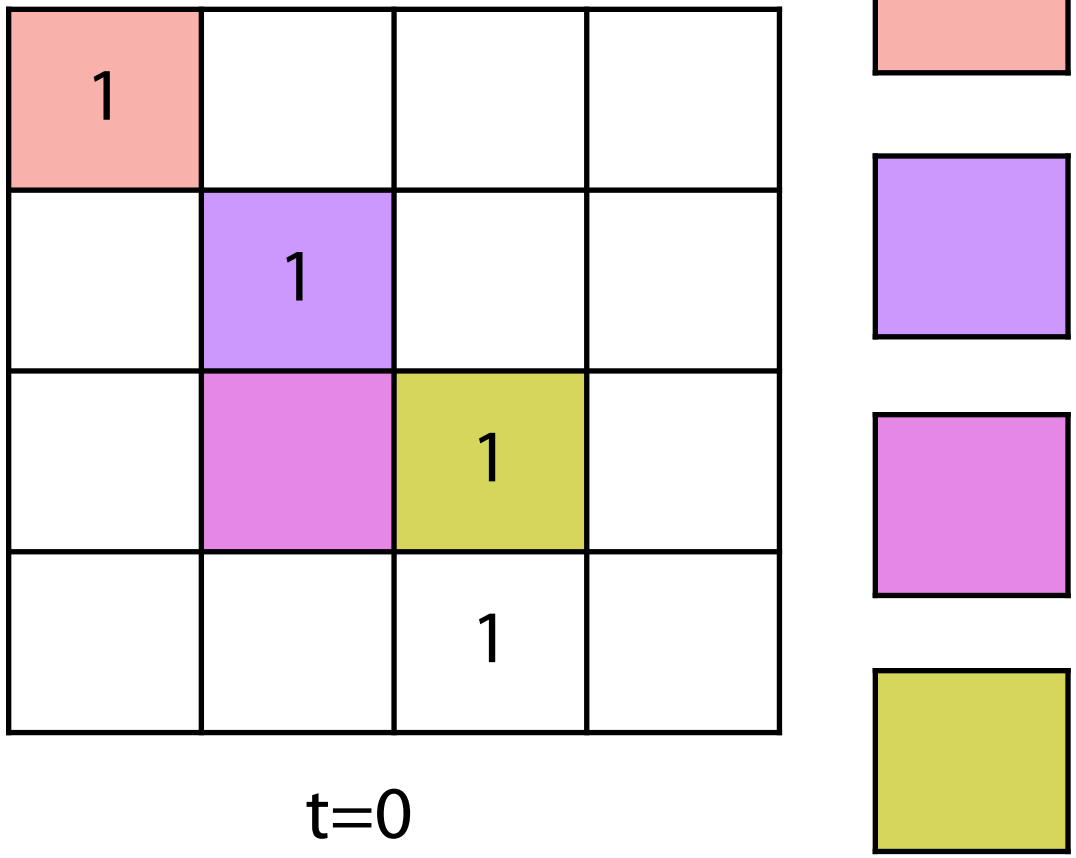
2. Any live cell with two or three live neighbors lives.

3. Any live cell with more than three live neighbors dies

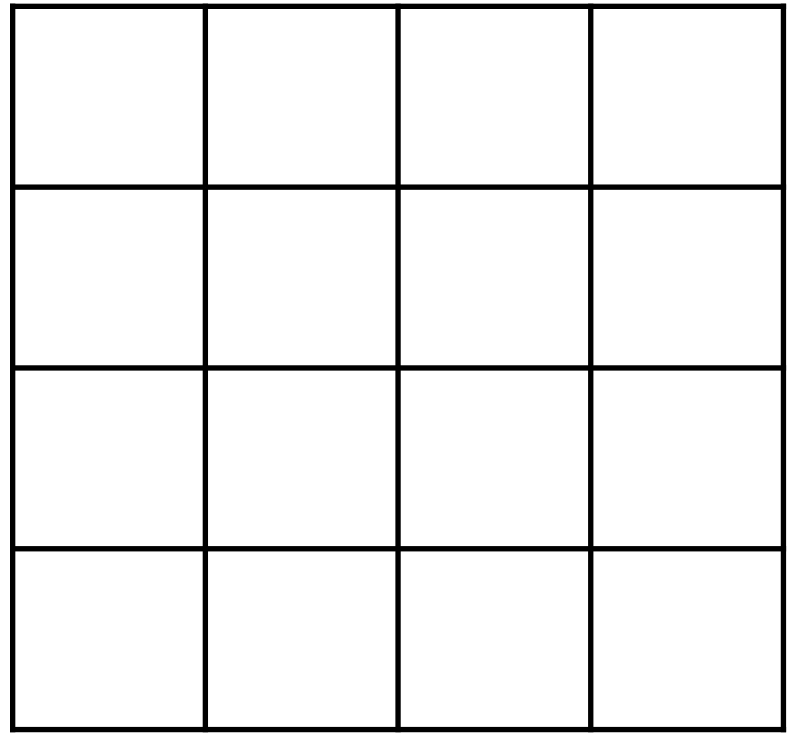
4. Any dead cells with exactly three live neighbors becomes a live cell.

The Game of Life Cellular Automata

```
gol_update(matrix)
```



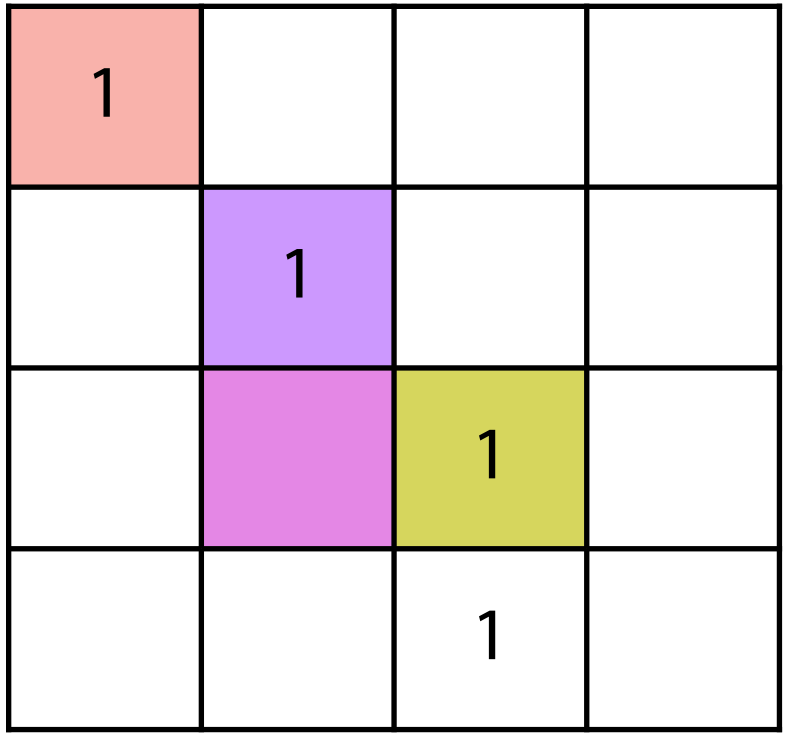
t=0



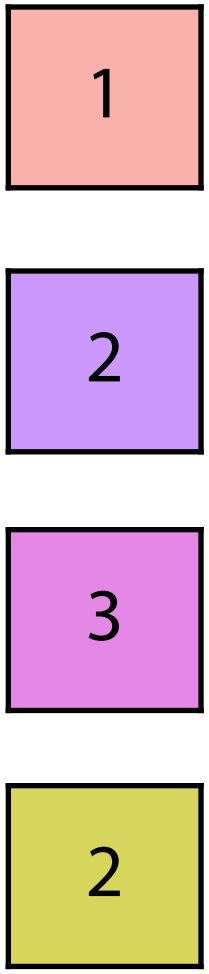
t=1

The Game of Life Cellular Automata

`gol_update(matrix)`



t=0

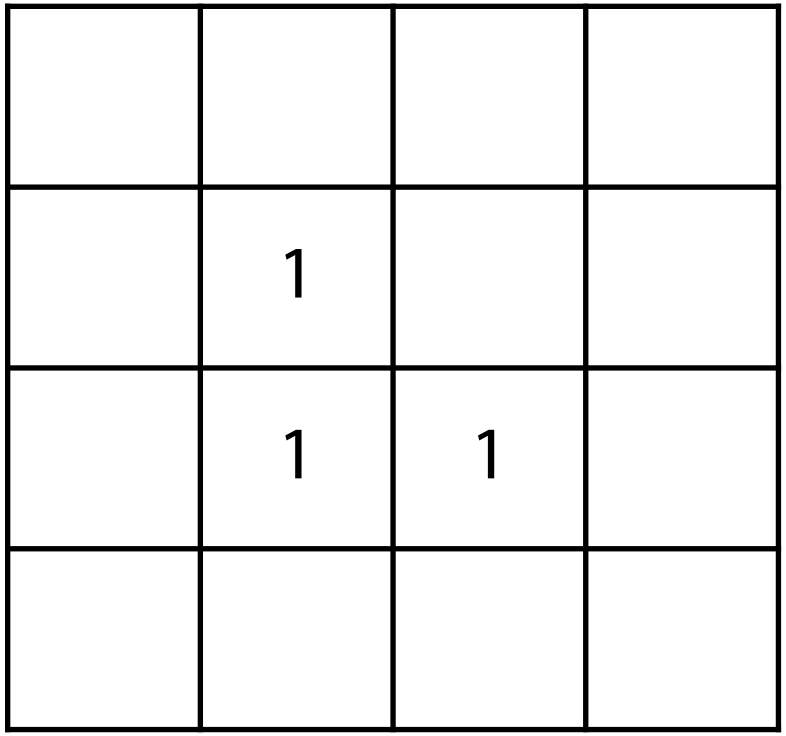


Dead

Alive

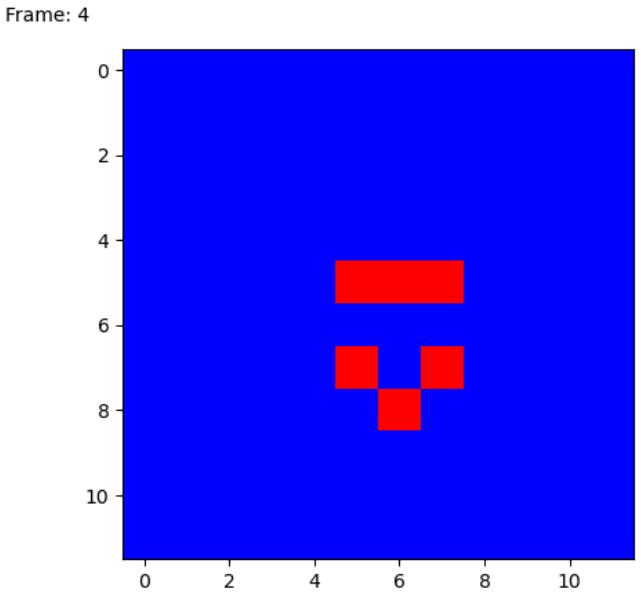
Alive

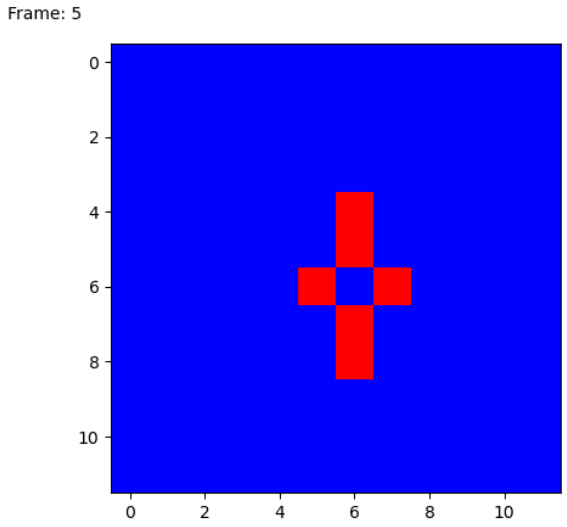
Alive

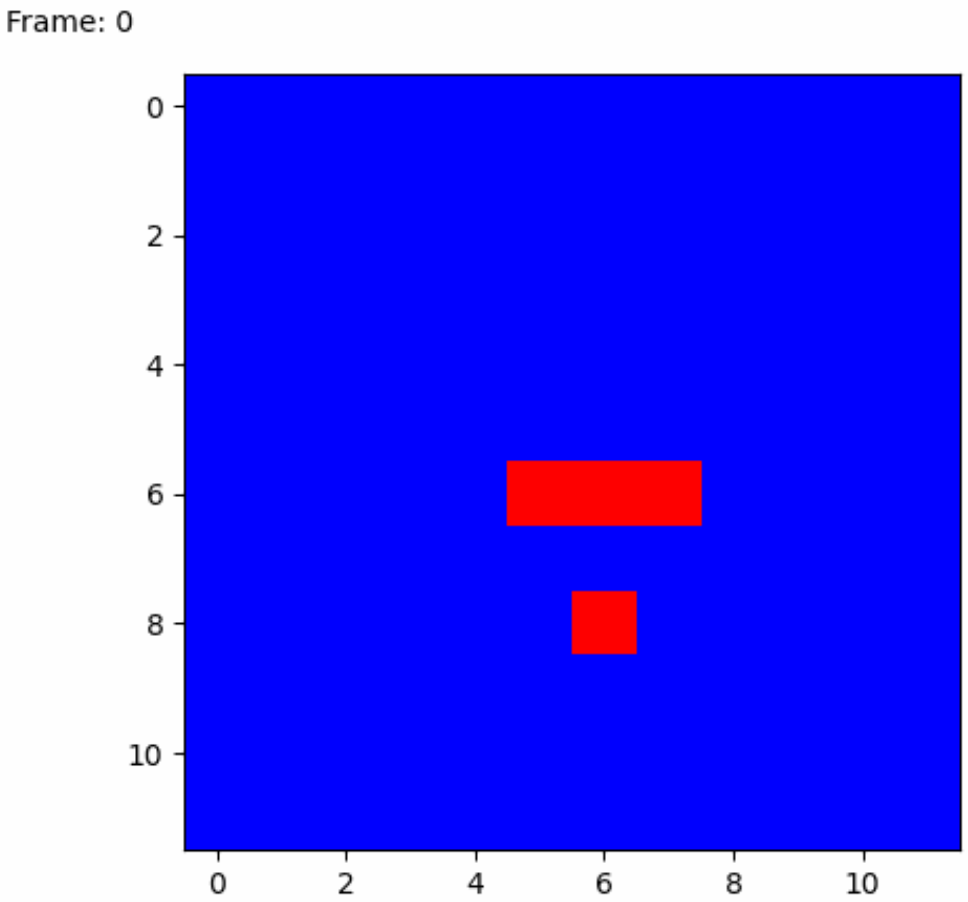


t=1

Conway's Game of Life



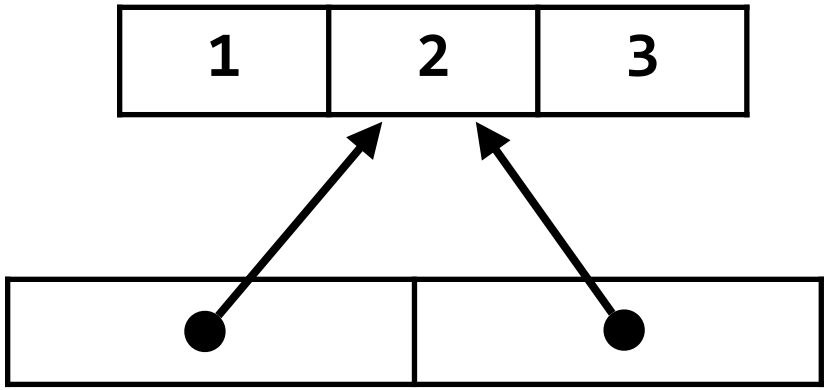


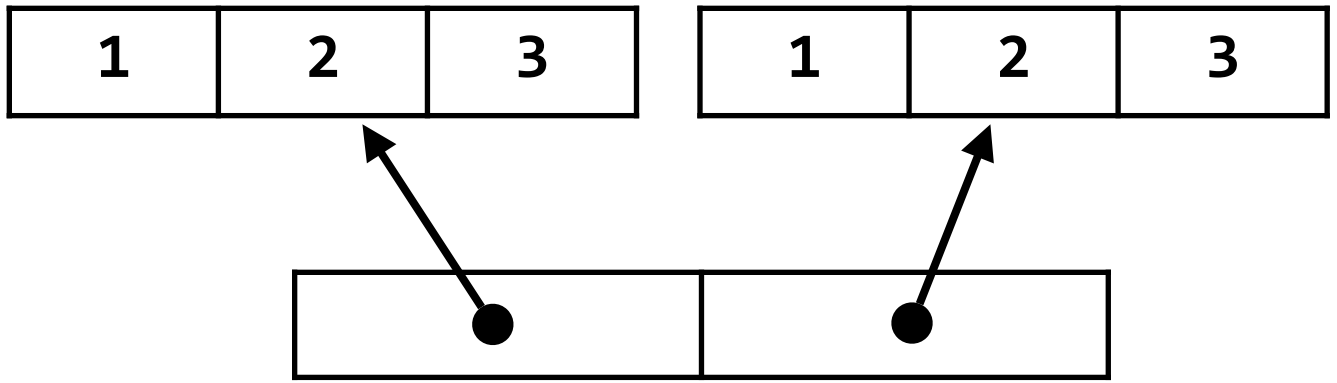


Programming Practice: 2D Lists

What happens if we append the same array twice?

```
1  def doubleAppend(inList):
2      return [inList, inList]
3
4  x = doubleAppend([1, 2, 3])
5
6  x[0][0]=9
7
```





Programming Practice: 2D List copying

What happens when we run the following code?

```
1  import copy
2
3  orig = [ [1,2,3], [4, 5, 6]]
4
5  copy = orig.copy()
6
7  orig[1][1]=9
8  copy[0][2]=7
9
10 print(orig)
11 print(copy)
12
13
14
15
16
17
18
```

Programming Practice: 2D List copying

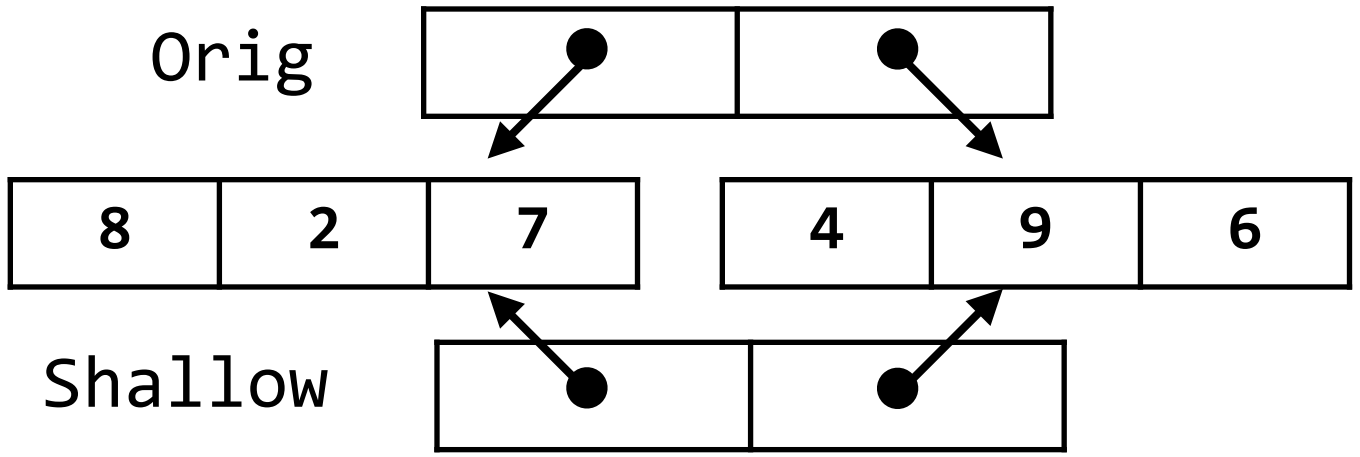
The new code shows that the list stored at 'copy' is an actual copy!

But it shares two of its three lists with original. Why?

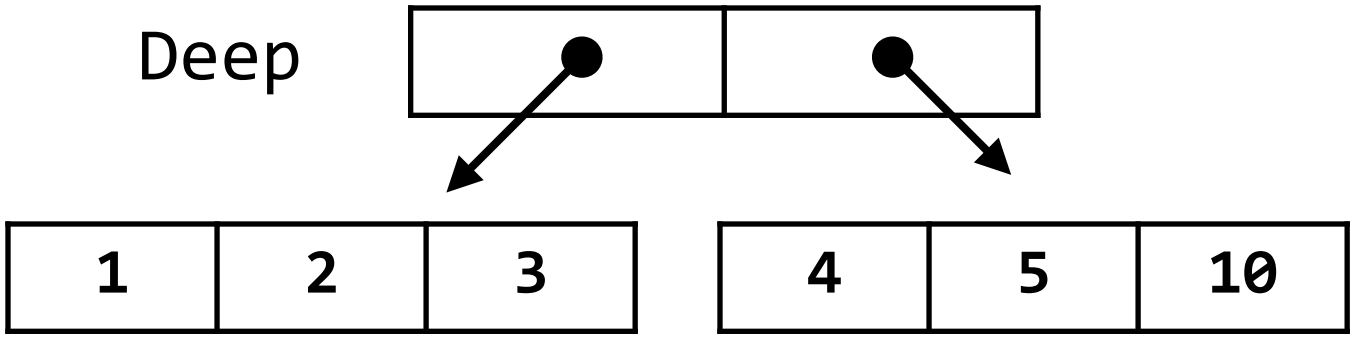
```
 1 import copy
 2
 3 orig = [ [1,2,3], [4, 5, 6]]
 4
 5 copy = orig.copy()
 6
 7 orig[1][1]=9
 8 copy[0][2]=7
 9
10 print(orig)
11 print(copy)
12
13 copy.append([7, 8, 9])
14
15 print(orig)
16 print(copy)
17
18
```

Shallow vs Deep Copying

A **shallow copy** creates a new list but copies all values directly



A **deep copy** creates new instances of every object in the list



Shallow vs Deep Copying

To protect yourself from accidental shallow copies, **copy individual values!**

Don't do this:

```
l1 = [ [1, 2], [3, 4] ]

l2 = l1[:]


l3 = l1.copy()



l4 = []
for i in l1:
    l4.append(i)



l1[0][0]=9

print(l1, l2, l3, l4)
```

Do this:

```
l1 = [ [1, 2], [3, 4] ]

l2 = copy.deepcopy(l1)


l3 = []
for row in l1:
    tmp = []
    for val in row:
        tmp.append(val)
    l3.append(tmp)


l4 = np.array(l1)

l1[0][0]=9

print(l1, l2, l3, l4)
```