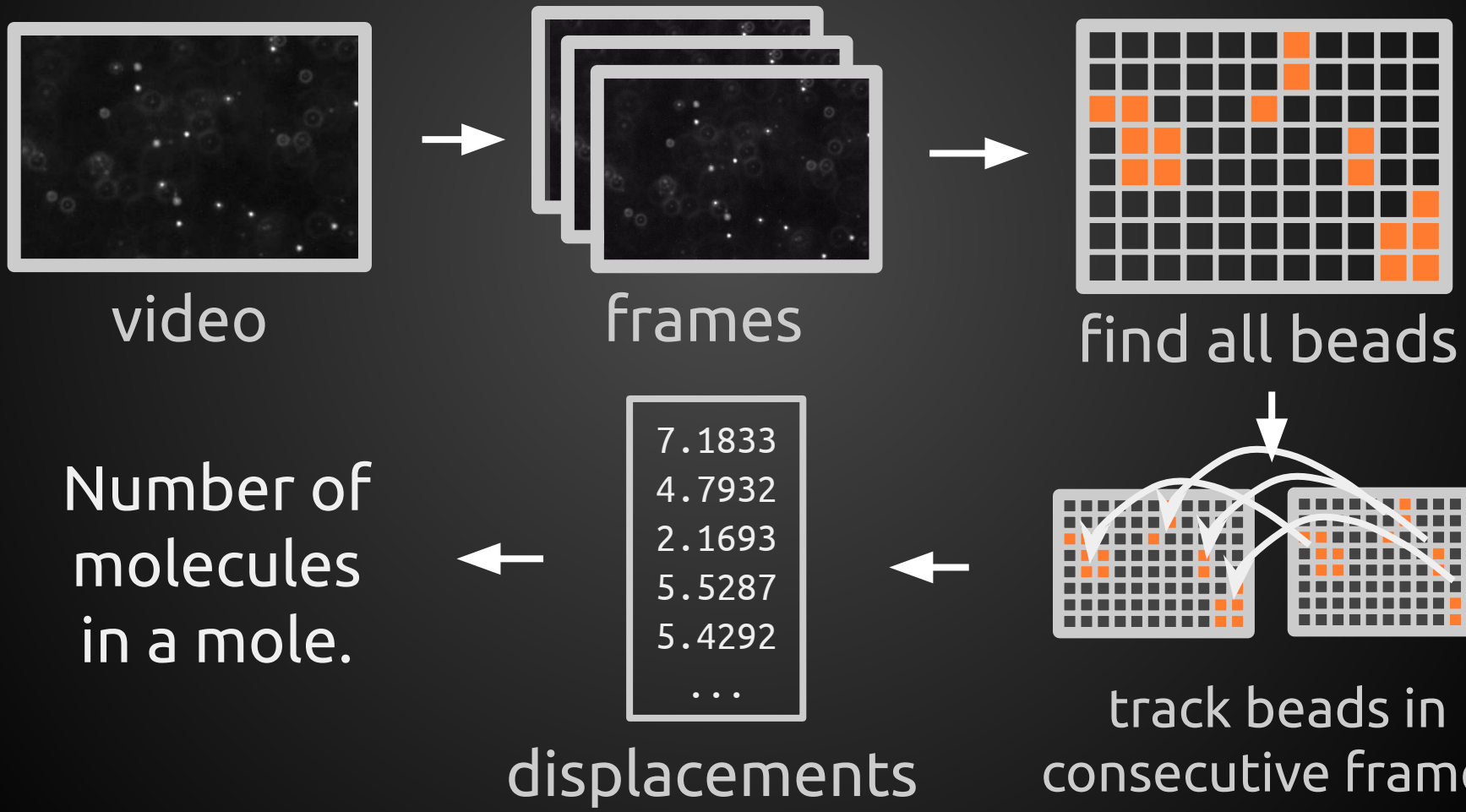
Atomic Nature of Matter

Use microscopy video to calculate Avogadro's #.



Beads represented as "Blobs"

Our goal is to analyze a collection of microscopy videos of beads suspended in water and find Avogadro's #.

Beads represented as "Blobs"

Our goal is to analyze a collection of microscopy videos of beads suspended in water and find Avogadro's #.

- Luckily, the first step is done for us; we're given each video as a sequence of images (.5s apart).

Beads represented as "Blobs"

Our goal is to analyze a collection of microscopy videos of beads suspended in water and find Avogadro's #.

- Luckily, the first step is done for us; we're given each video as a sequence of images (.5s apart).
  - So, our input isn't a video, it's n images (e.g., n = 200).

Beads represented as "Blobs"

Our goal is to find Avogadro's # by analyzing microscopy videos of beads suspended in water and

- We're given each video as a sequence of images (.5 seconds apart).
  - So, our input isn't a video, it's n images (e.g., n = 200).
- Our first step is to find all the beads in one (1) image.

Beads represented as `Blobs`

Our goal is to find Avogadro's # by analyzing microscopy videos of beads suspended in water and

- We're given each video as a sequence of images (.5 seconds apart).
  - So, our input isn't a video, it's n images (e.g., n = 200).
- Our first step is to find all the beads in 1 image.
  - To do that, we'll make an abstract data type called **Blob** to store beads and a client called **BeadFinder**.

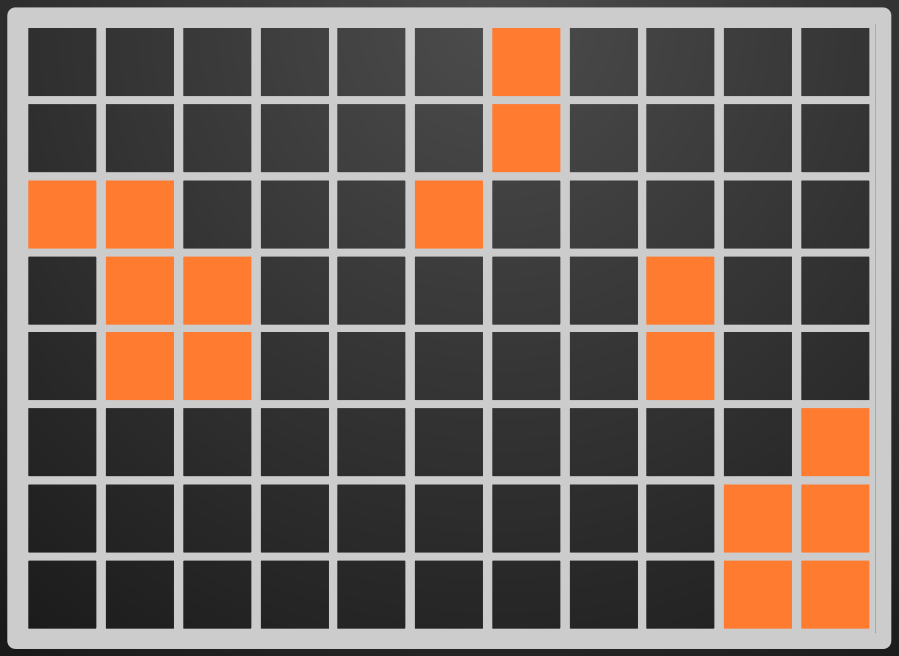
What is a Blob?

A blob is a contiguous group of pixels, connected in the four cardinal directions (i.e., not diagonally).

What is a Blob?

A blob is a contiguous group of pixels, connected in the four cardinal directions (i.e., not diagonally).

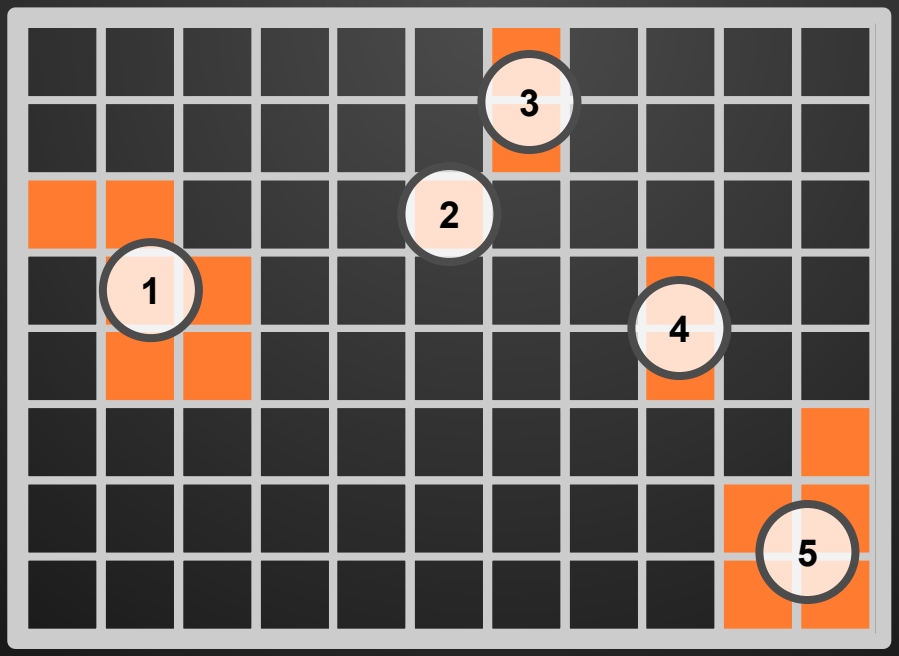
Q: How many blobs are there in this frame?



What is a Blob?

A blob is a contiguous group of pixels, connected in the four cardinal directions (i.e., not diagonally).

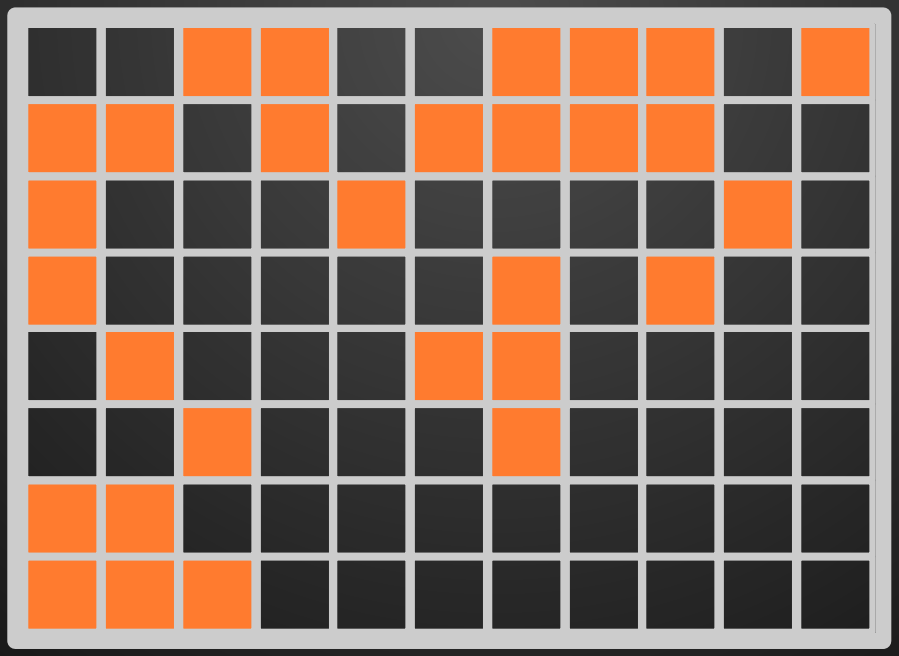
Q: How many blobs are there in this frame?



What is a Blob?

A blob is a contiguous group of pixels, connected in the four cardinal directions (i.e., not diagonally).

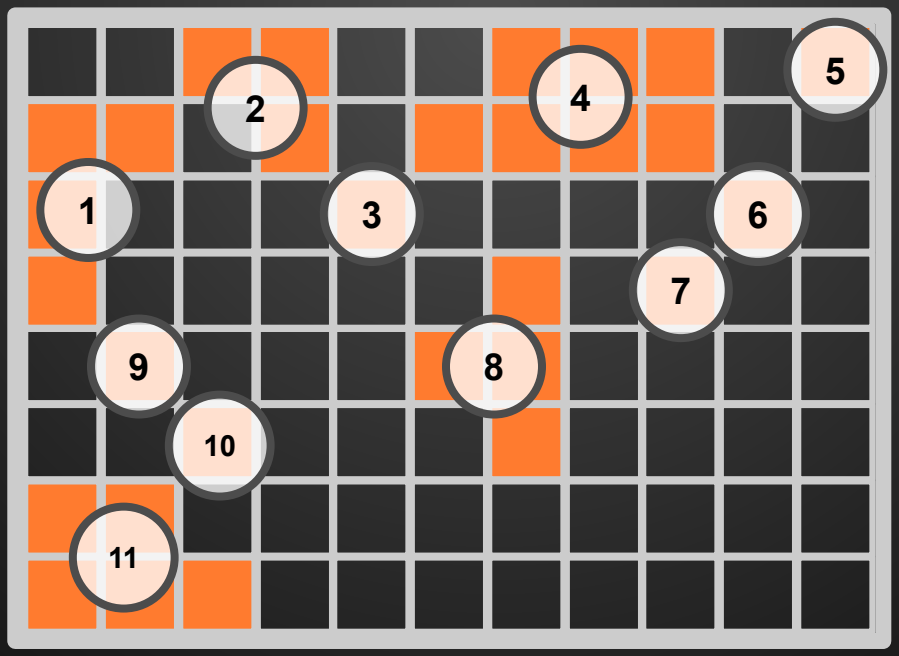
Q: How many blobs are there in this frame?



What is a Blob?

A blob is a contiguous group of pixels, connected in the four cardinal directions (i.e., not diagonally).

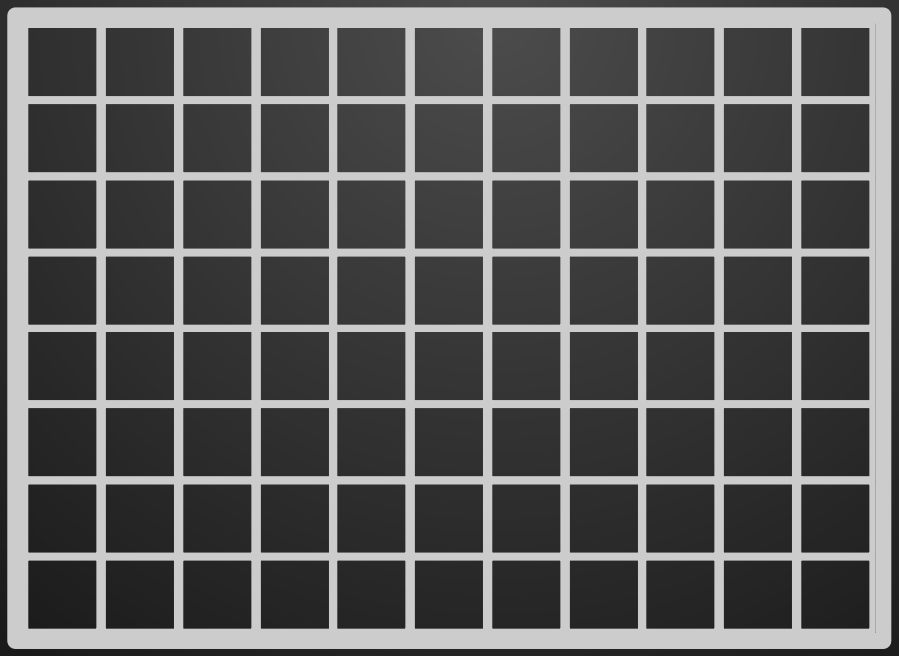
Q: How many blobs are there in this frame?



What is a Blob?

A blob is a contiguous group of pixels, connected in the four cardinal directions (i.e., not diagonally).

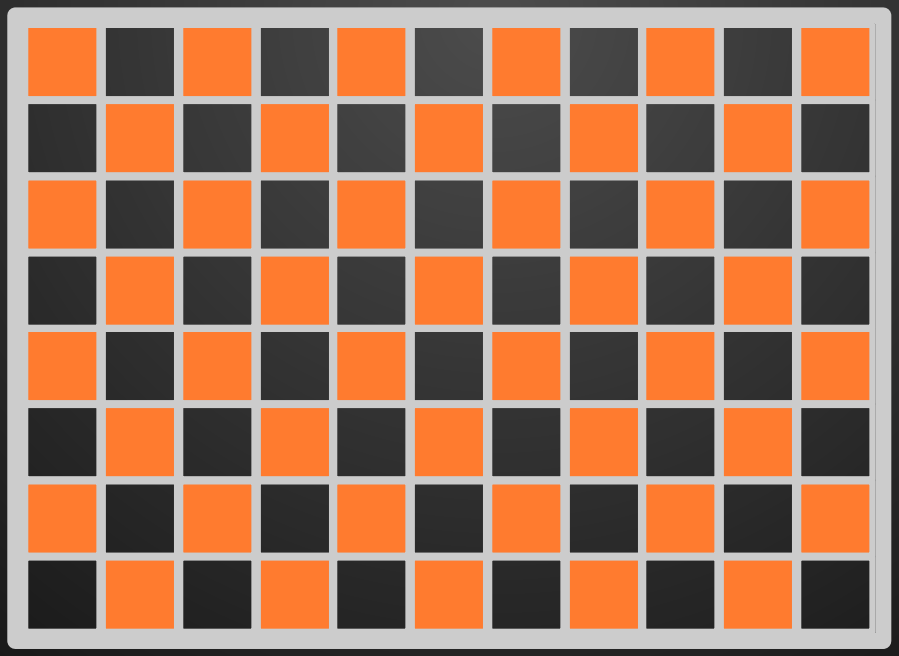
Q: Draw a frame with the max. # of Blobs.



What is a Blob?

A blob is a contiguous group of pixels, connected in the four cardinal directions (i.e., not diagonally).

Q: Draw a frame with the max. # of Blobs.



Center of Mass

The center of mass of a Blob is a point (x, y) whose x-coordinate is the avg. of all the x-coords and whose y-coordinate is the avg. of all the y-coords.

Center of Mass

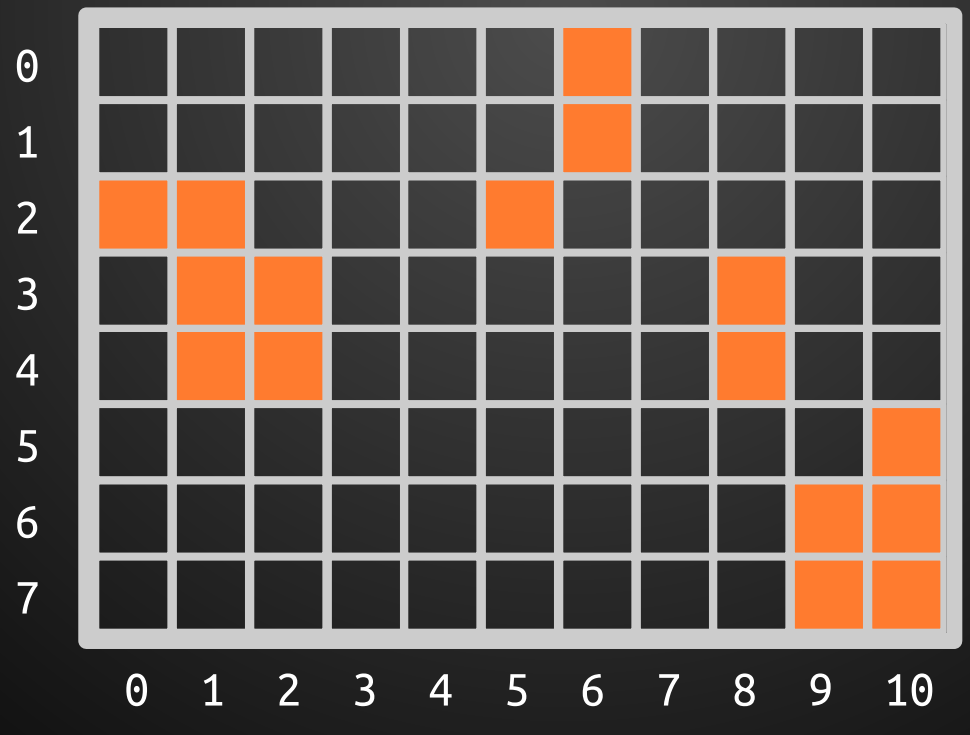
The center of mass of a Blob is a point (x, y) whose x-coordinate is the avg. of all the x-coords and whose y-coordinate is the avg. of all the y-coords.

Q: What is the center of mass of the **top** blob?

Center of Mass

The center of mass of a Blob is a point (x, y) whose x-coordinate is the avg. of all the x-coords and whose y-coordinate is the avg. of all the y-coords.

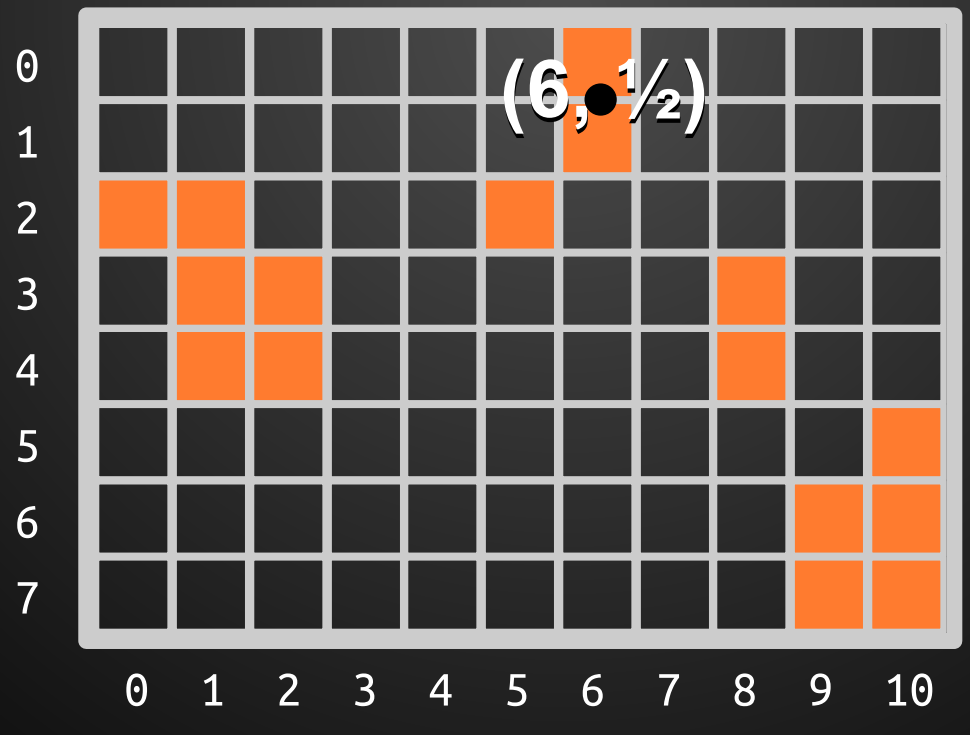
Q: What is the center of mass of the **top** blob?



Center of Mass

The center of mass of a Blob is a point (x, y) whose x-coordinate is the avg. of all the x-coords and whose y-coordinate is the avg. of all the y-coords.

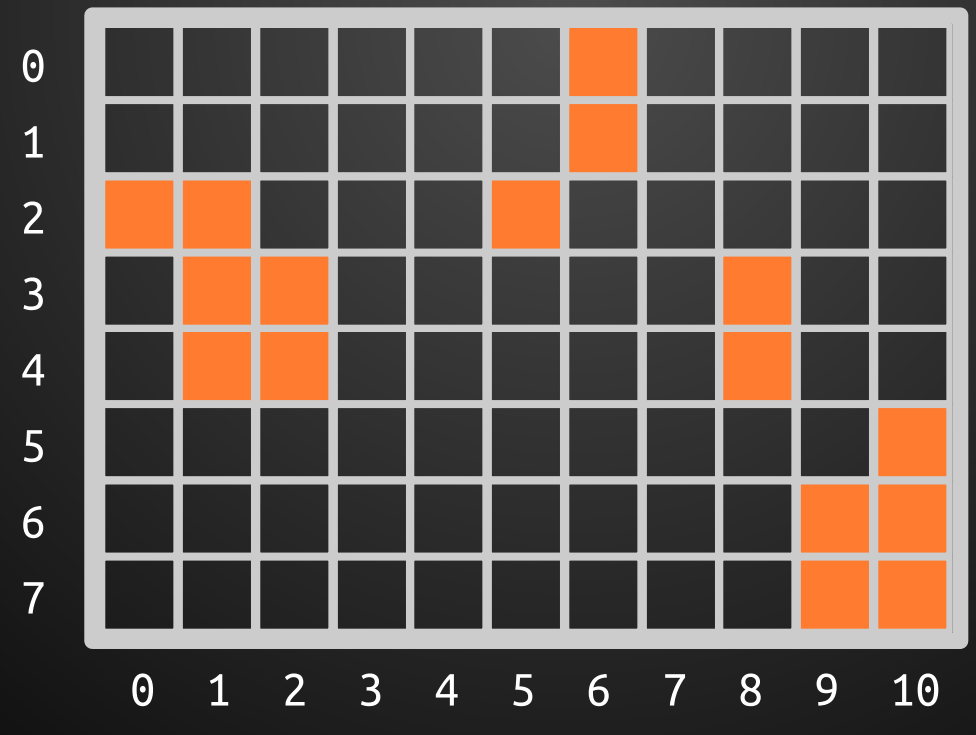
Q: What is the center of mass of the **top** blob?



Center of Mass

The center of mass of a Blob is a point (x, y) whose x-coordinate is the avg. of all the x-coords and whose y-coordinate is the avg. of all the y-coords.

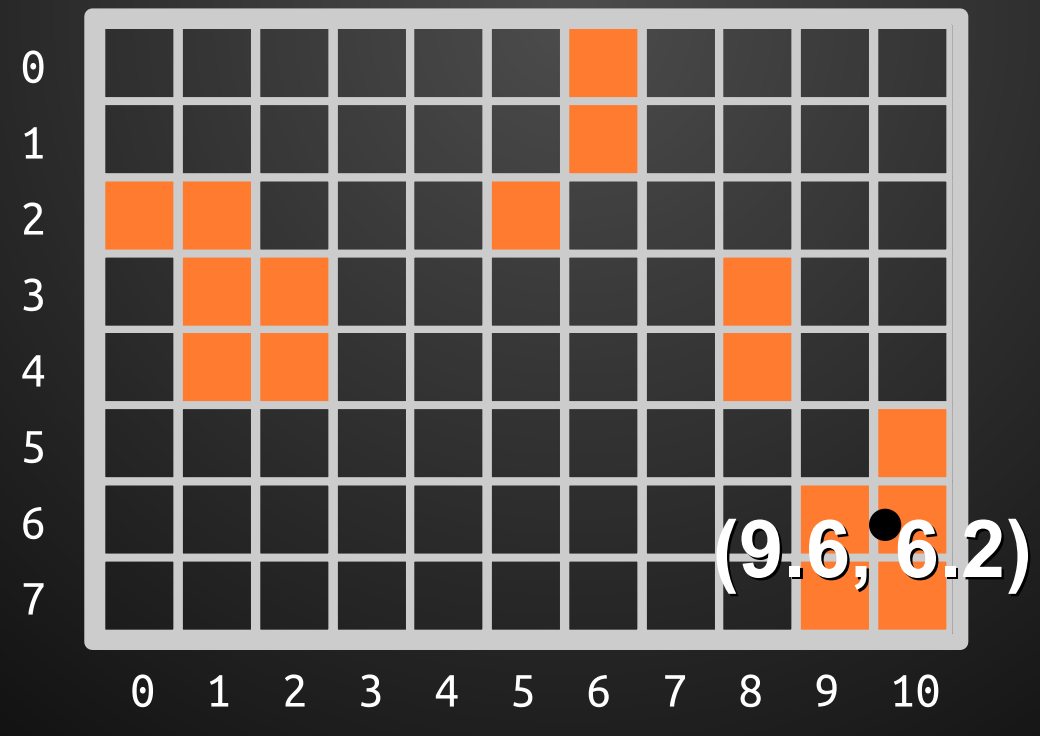
Q: What is the center of mass of **bottom** blob?



Center of Mass

The center of mass of a Blob is a point (x, y) whose x-coordinate is the avg. of all the x-coords and whose y-coordinate is the avg. of all the y-coords.

Q: What is the center of mass of **bottom** blob?



Blob.java

- Implement:
  - `Blob()`

Blob.java

- Implement:
  - `Blob()`
  - `void add(int i, int j)`

Blob.java

- Implement:
  - `Blob()`
  - `void add(int i, int j)`
  - `int mass()`

Blob.java

- Implement:
  - `Blob()`
  - `void add(int i, int j)`
  - `int mass()`
  - `double distanceTo(Blob b)`

Blob.java

- Implement:
  - `Blob()`
  - `void add(int i, int j)`
  - `int mass()`
  - `double distanceTo(Blob b)`
  - `String toString()`

Blob.java

- Implement:
  - `Blob()`
  - `void add(int i, int j)`
  - `int mass()`
  - `double distanceTo(Blob b)`
  - `String toString()`
  - `void main(String[] args)`

Blob.java

- Implement:
  - `Blob()`
  - `void add(int i, int j)`
  - `int mass()`
  - `double distanceTo(Blob b)`
  - `String toString()`
  - `void main(String[] args)`
- What instance variables do we need?

Blob.java

- Implement:
  - `Blob()`
  - `void add(int i, int j)`
  - `int mass()`
  - `double distanceTo(Blob b)`
  - `String toString()`
  - `void main(String[] args)`
- What instance variables do we need?
  - Don't store all the coordinates!

Blob Tips

- In toString(), how do I format my output?

Blob Tips

- In toString(), how do I format my output?
  - Use this code:

    String.format("%2d (%8.4f, %8.4f)", mass, cx, cy);

Blob Tips

- In toString(), how do I format my output?
  - Use this code:

    String.format("%2d (%8.4f, %8.4f)", mass, cx, cy);
- In main(), how do I test my program?

Blob Tips

- In toString(), how do I format my output?
  - Use this code:

    String.format("%2d (%8.4f, %8.4f)", mass, cx, cy);
- In main(), how do I test my program?
  - Think of, say, 3 equidistant points on a line.

Blob Tips

- In toString(), how do I format my output?
  - Use this code, available on the checklist:

```
String.format("%2d (%8.4f, %8.4f)", mass, cx, cy);
```

- In `main()`, how do I test my program?
  - Think of, say, three (3) equidistant points on a line.
  - If you `add()` all three (3) points, which will be the center?

Blob Tips

- In toString(), how do I format my output?
  - Use this code, available on the checklist:

    String.format("%2d (%8.4f, %8.4f)", mass, cx, cy);
- In main(), how do I test my program?
  - Think of, say, 3 equidistant points on a line.
  - If you add() all 3 points, which will be the center?
  - You can think of other tests, but **don’t forget** to test.

Blob Tips

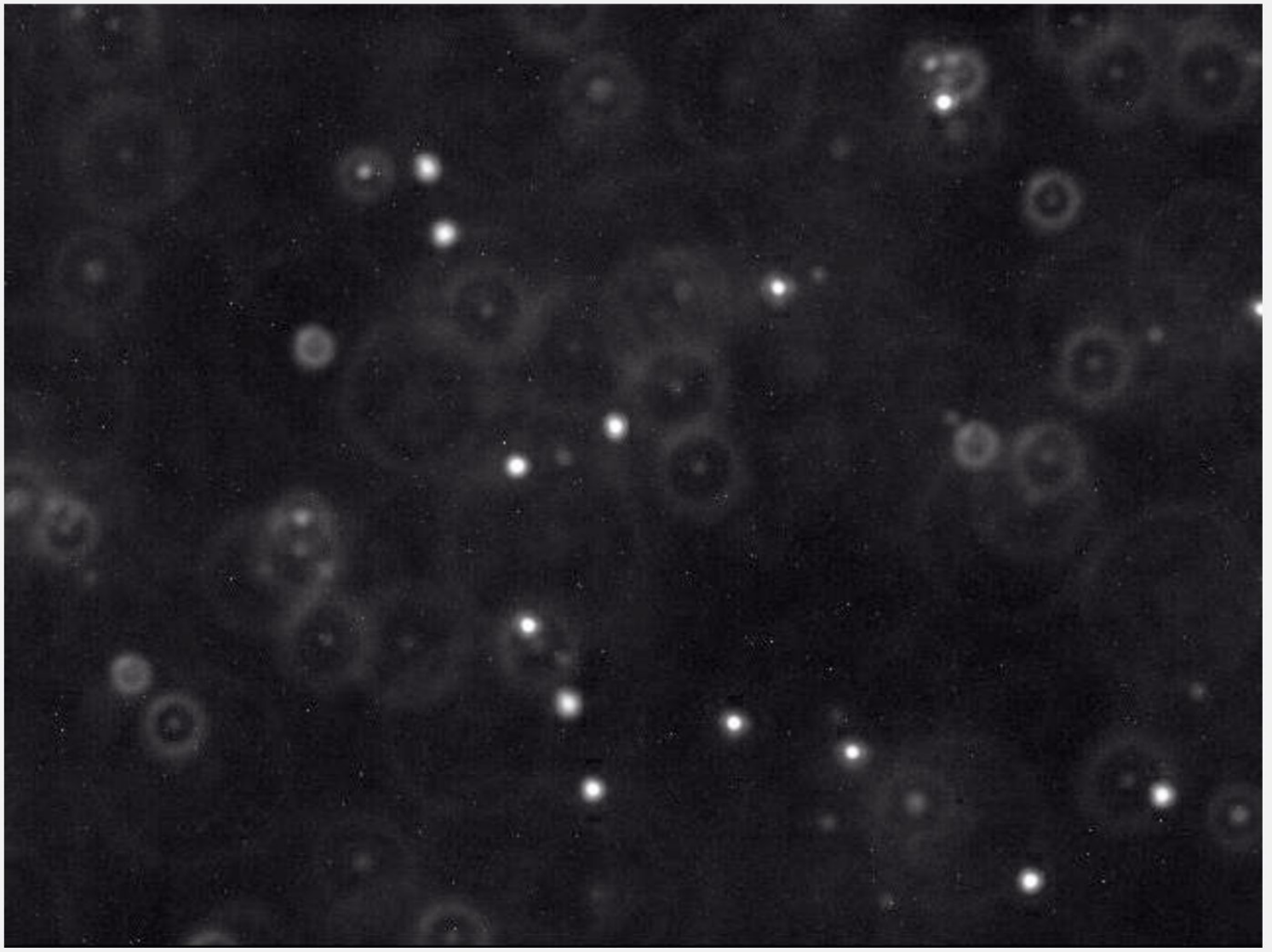
- In toString(), how do I format my output?
  - Use this code, available on the checklist:

    String.format("%2d (%8.4f, %8.4f)", mass, cx, cy);
- In main(), how do I test my program?
  - Think of, say, 3 equidistant points on a line.
  - If you add() all 3 points, which will be the center?
  - You can think of other tests, but **don't forget** to test.
  - Also, don't forget to *effectively* test *all* your methods.

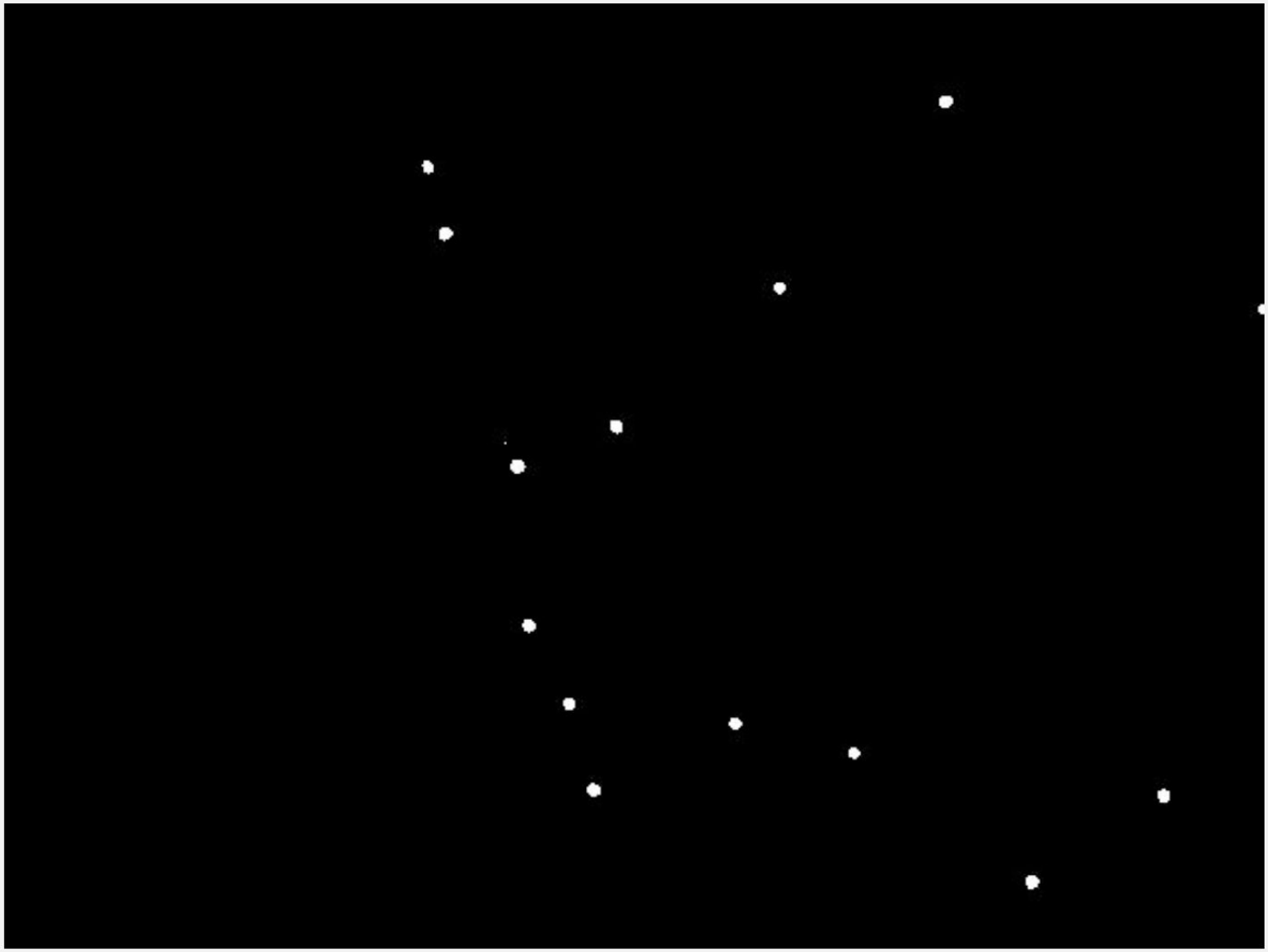
What does BeadFinder do?

```
/*  finds all blobs in the specified picture using
    luminance threshold tau */
public BeadFinder(Picture picture, double tau)


/*  returns all beads (blobs with >= min pixels) */
public Blob[] getBeads(int min)


/*  test client description in spec */
public static void main(String[] args)
```

BeadFinder: Original Image

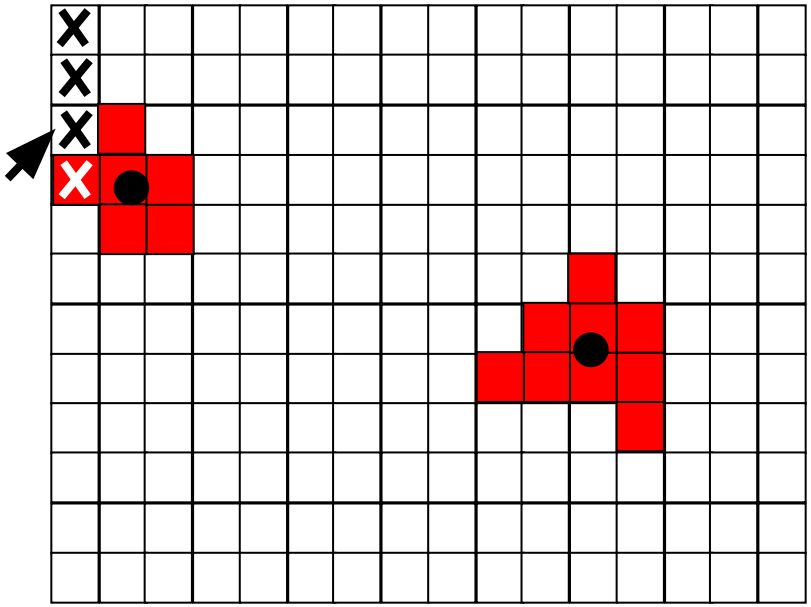


BeadFinder: Applying Luminance Threshold tau



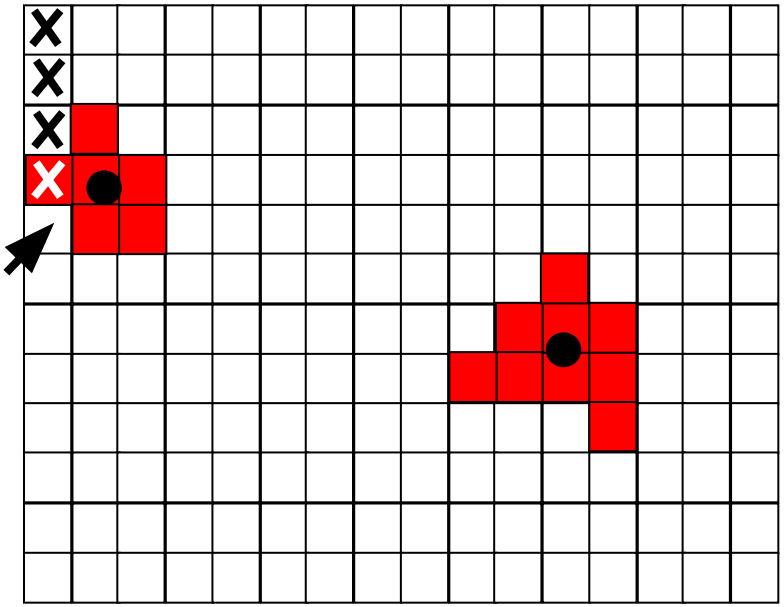
BeadFinder - Depth First Search

- Use boolean[][] array to mark visited
- Traverse image pixel by pixel
  - Dark pixel
    - Mark as visited, continue
  - Light pixel
    - Create new blob, call DFS
- DFS algorithm
  - Base case: simply return if
    - Pixel out-of-bounds
    - Pixel has been visited
    - Pixel is dark (and mark as visited)
  - Add pixel to current blob, mark as visited
  - Recursively visit up, down, left, and right neighbors



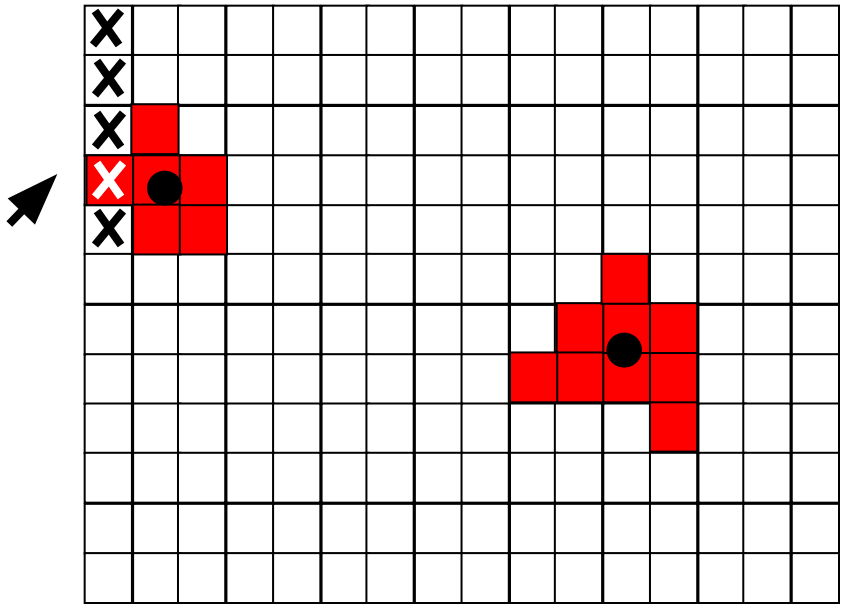
BeadFinder - Depth First Search

- Use boolean[][] array to mark visited
- Traverse image pixel by pixel
  - Dark pixel
    - Mark as visited, continue
  - Light pixel
    - Create new blob, call DFS
- DFS algorithm
  - Base case: simply return if
    - Pixel out-of-bounds
    - Pixel has been visited
    - Pixel is dark (and mark as visited)
  - Add pixel to current blob, mark as visited
  - Recursively visit up, down, left, and right neighbors



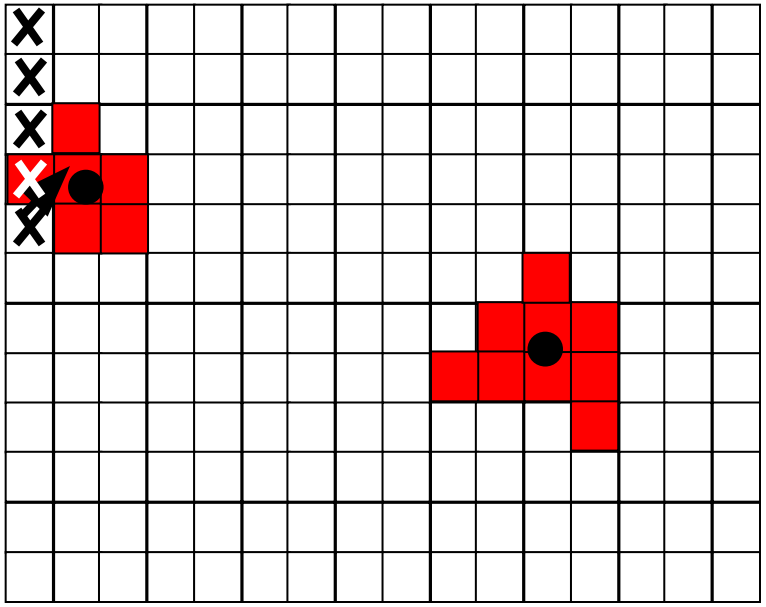
BeadFinder - Depth First Search

- Use boolean[][] array to mark visited
- Traverse image pixel by pixel
  - Dark pixel
    - Mark as visited, continue
  - Light pixel
    - Create new blob, call DFS
- DFS algorithm
  - Base case: simply return if
    - Pixel out-of-bounds
    - Pixel has been visited
    - Pixel is dark (and mark as visited)
  - Add pixel to current blob, mark as visited
  - Recursively visit up, down, left, and right neighbors



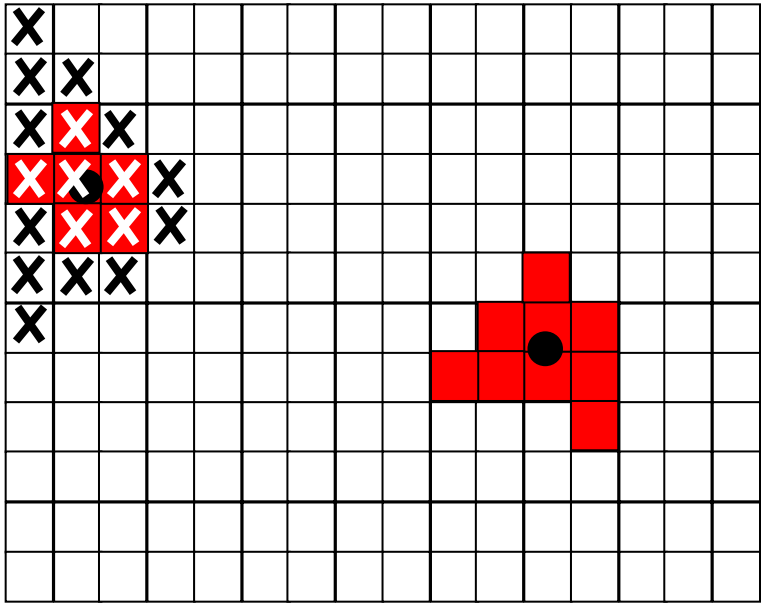
BeadFinder - Depth First Search

- Use boolean[][] array to mark visited
- Traverse image pixel by pixel
  - Dark pixel
    - Mark as visited, continue
  - Light pixel
    - Create new blob, call DFS
- DFS algorithm
  - Base case: simply return if
    - Pixel out-of-bounds
    - Pixel has been visited
    - Pixel is dark (and mark as visited)
  - Add pixel to current blob, mark as visited
  - Recursively visit up, down, left, and right neighbors



BeadFinder - Depth First Search

- Use boolean[][] array to mark visited
- Traverse image pixel by pixel
  - Dark pixel
    - Mark as visited, continue
  - Light pixel
    - Create new blob, call DFS
- DFS algorithm
  - Base case: simply return if
    - Pixel out-of-bounds
    - Pixel has been visited
    - Pixel is dark (and mark as visited)
  - Add pixel to current blob, mark as visited
  - Recursively visit up, down, left, and right neighbors



BeadFinder Challenges

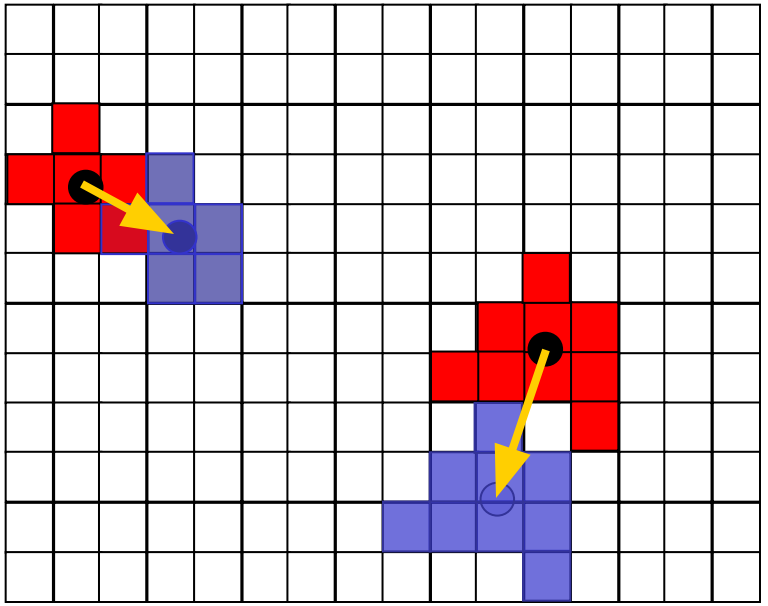
- Data structure for the collection of blobs
  - Store them any way you like
  - But be aware of memory use and timing

BeadFinder Challenges

- Data structure for the collection of blobs
  - Store them any way you like
  - But be aware of memory use and timing
- Array of blobs?
  - But how big should the array be?
- Linked list of blobs?
  - Memory efficient, but harder to implement
  - Avoid traversing whole list to add a blob!
- Anything else?
  - Submit your (extra) object classes

BeadTracker.java

- Track beads between successive images
- Single main function
  - Take in a series of images
  - Output distance traversed by all beads for each time-step
    - For each bead found at time t+1, find closest bead at time t and calculate distance
      - Not the other way around!
      - Don’t include if distance > 25 pixels (new bead)



BeadTracker Challenges

- Reading multiple input files
  - `java-introcs BeadTracker 25 180.0 25.0 run_1/*.jpg`
  - Expands files in alphanumeric order
  - End up as `args[0], args[1], …`
- Avoiding running out of memory
  - How?
- Recompiling
  - Recompile if Blob or BeadFinder change

BeadTracker Challenges

- Reading multiple input files
  - `java-introcs BeadTracker 25 180.0 25.0 run_1/*.jpg`
  - Expands files in alphabetical order
  - End up as `args[0], args[1], …`
- Avoiding running out of memory
  - Do *not* open all picture files at same time
  - Only two need to be open at a time
- Recompiling
  - Recompile if Blob or BeadFinder change

BeadTracker

Calculates the displacement of each bead between consecutive frames.

BeadTracker

Calculates the displacement of each bead between consecutive frames.

- Instead of tracking one bead's journey through all the images, let's look at all the consecutive frames and for each bead in the frame at time t+1, find the nearest bead in frame at time t.

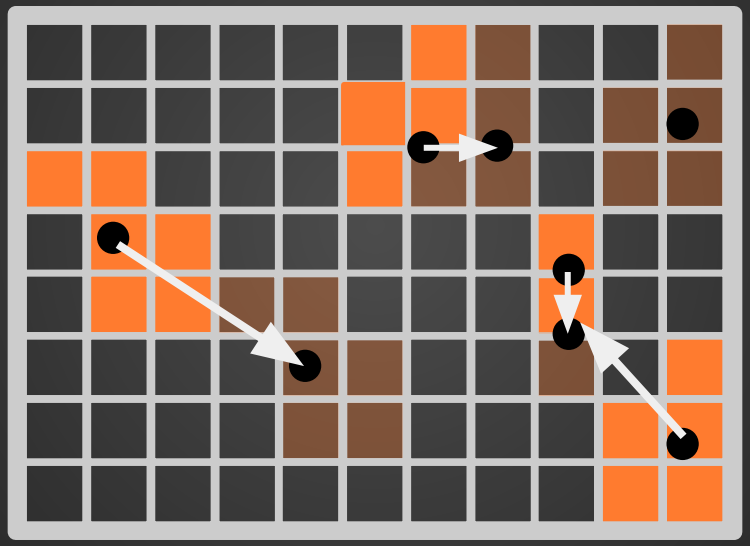
BeadTracker

Calculates the displacement of each bead between consecutive frames.

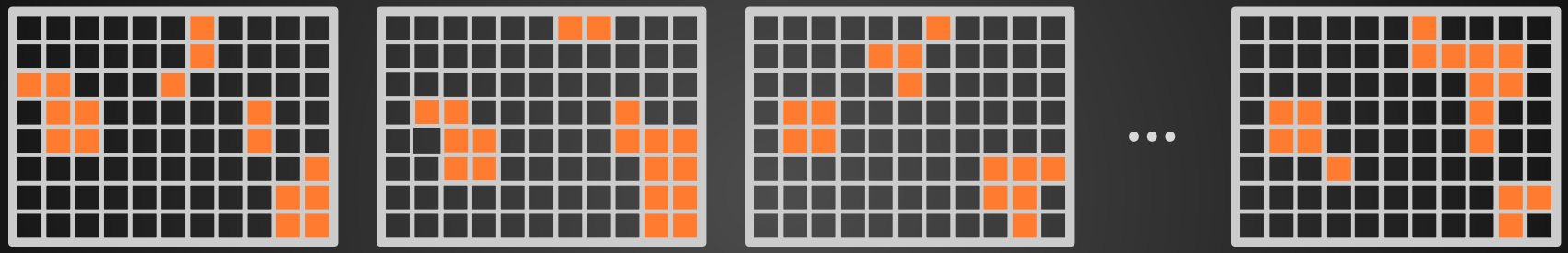
- Instead of tracking one bead's journey through all the images, let's look at all the consecutive frames and for each bead in the frame at time t+1, find the nearest bead in frame at time t.
- If it's within a maximal distance (i.e., <= delta), we'll assume it's the same bead. Otherwise, we assume that bead went out of focus in the previous frame.

BeadTracker

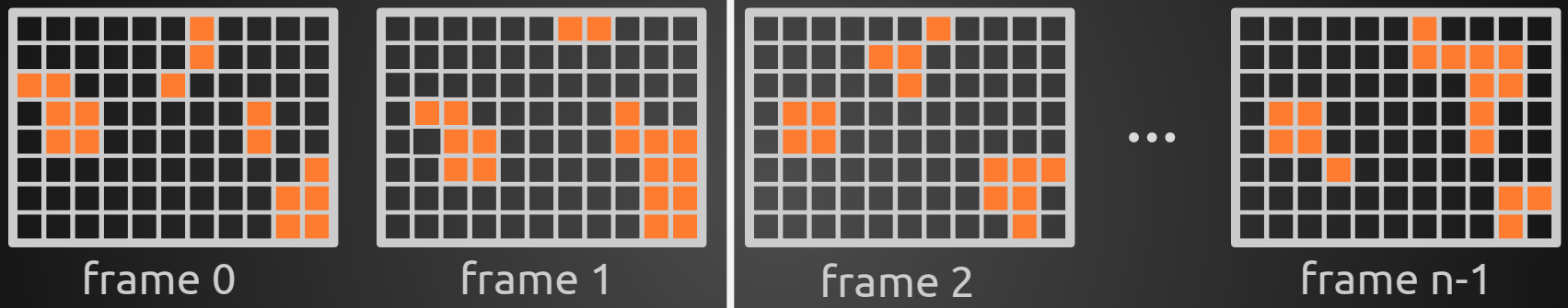
An example:



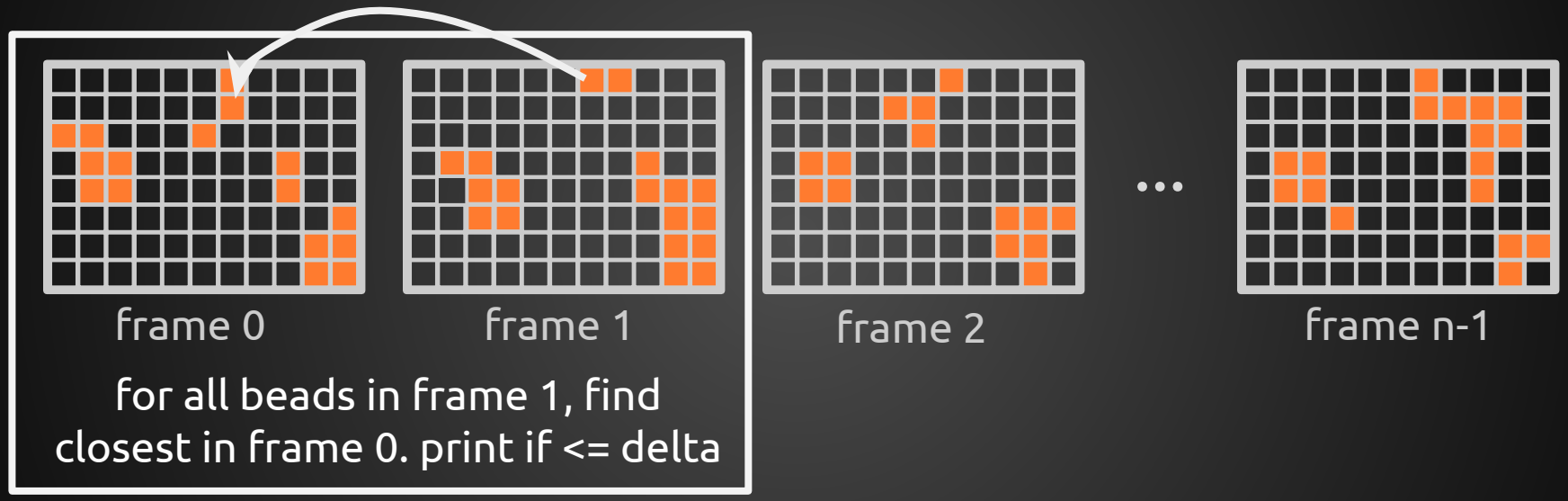
BeadTracker



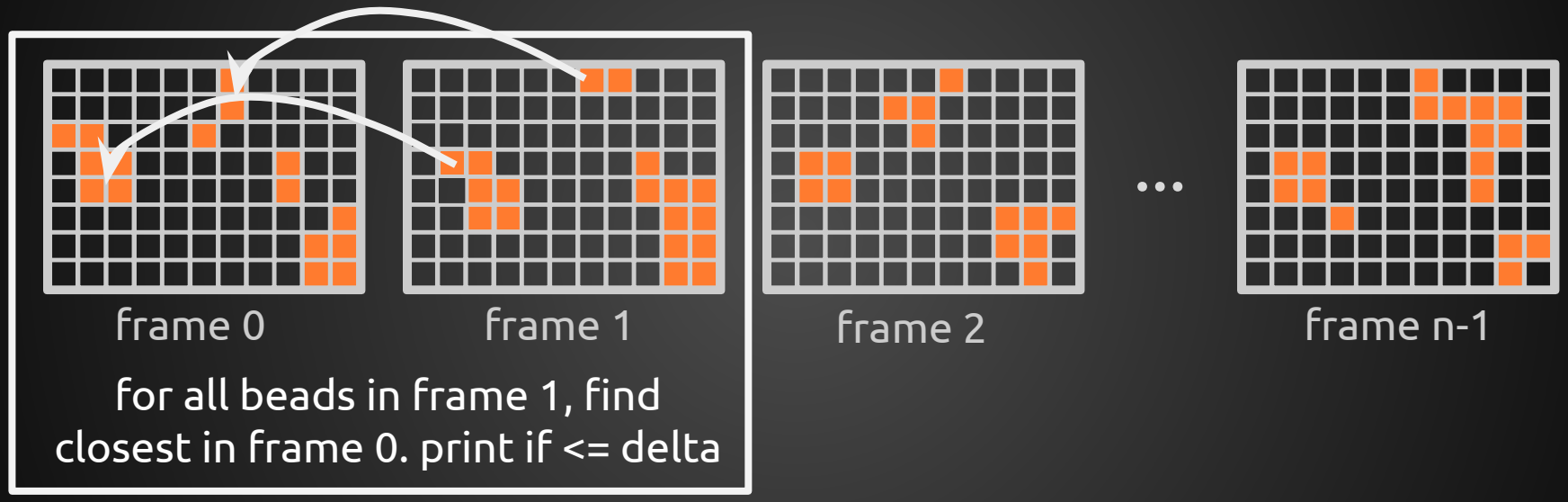
BeadTracker



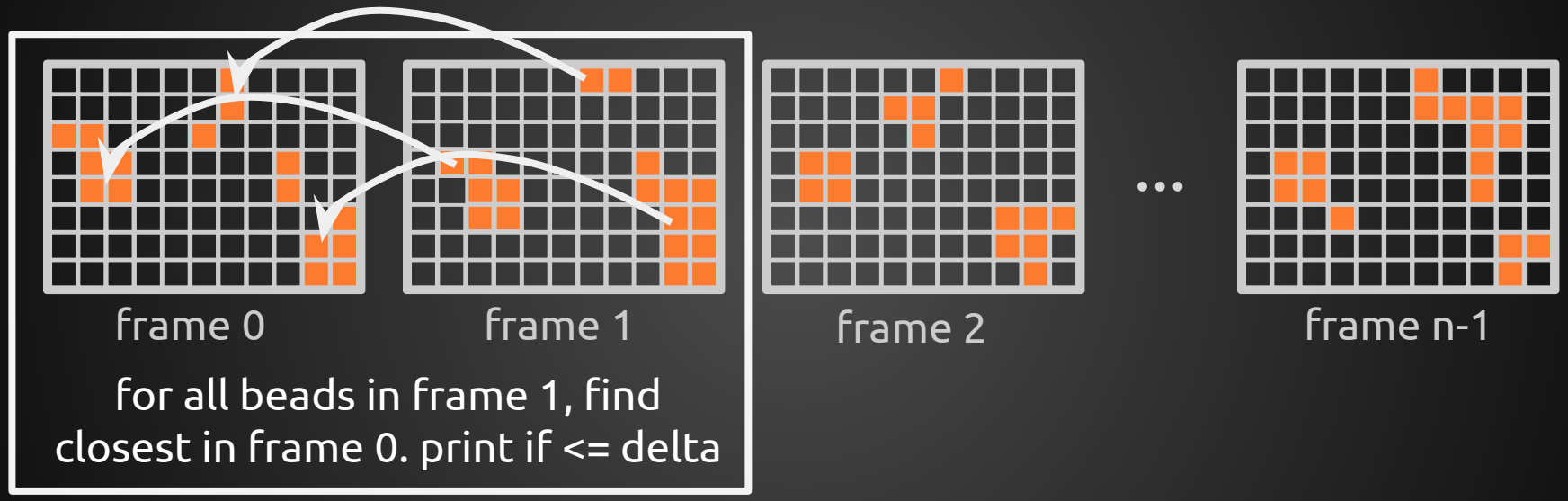
BeadTracker



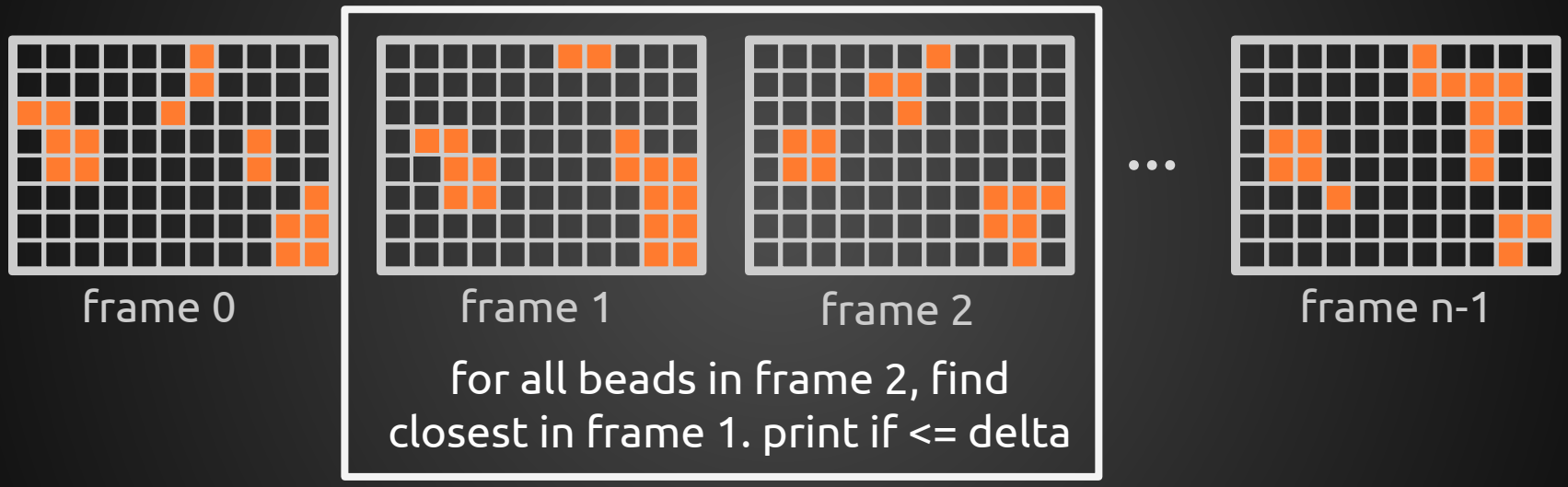
BeadTracker



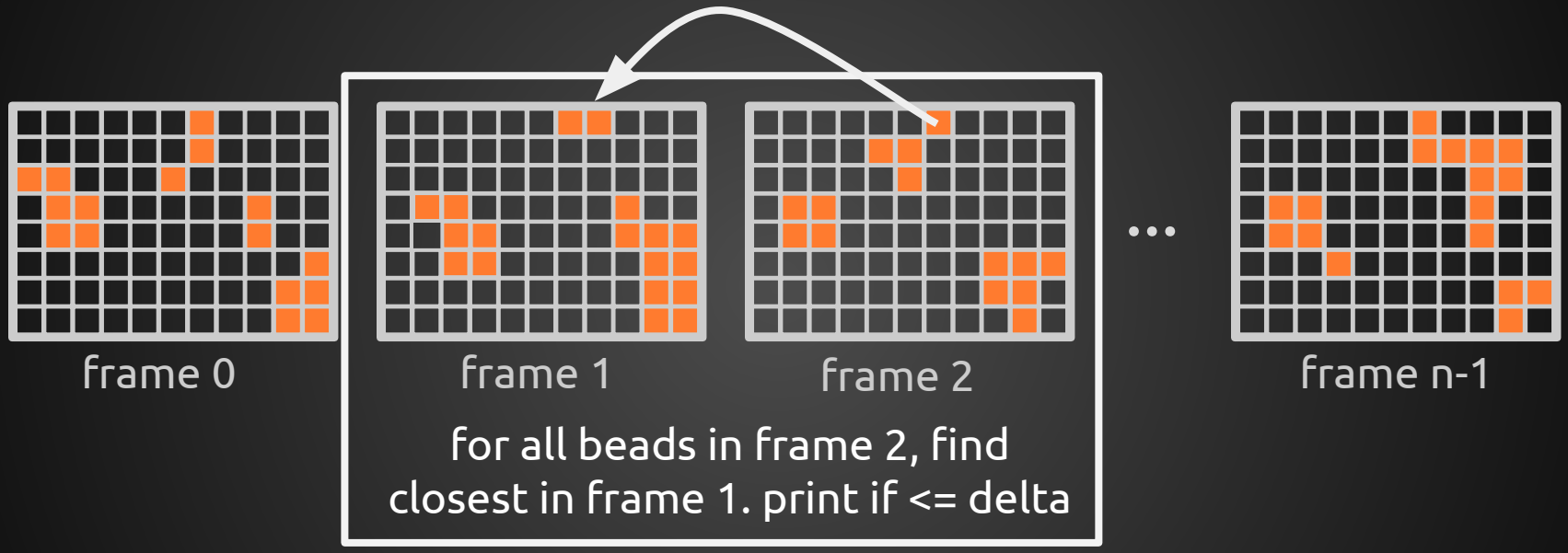
BeadTracker



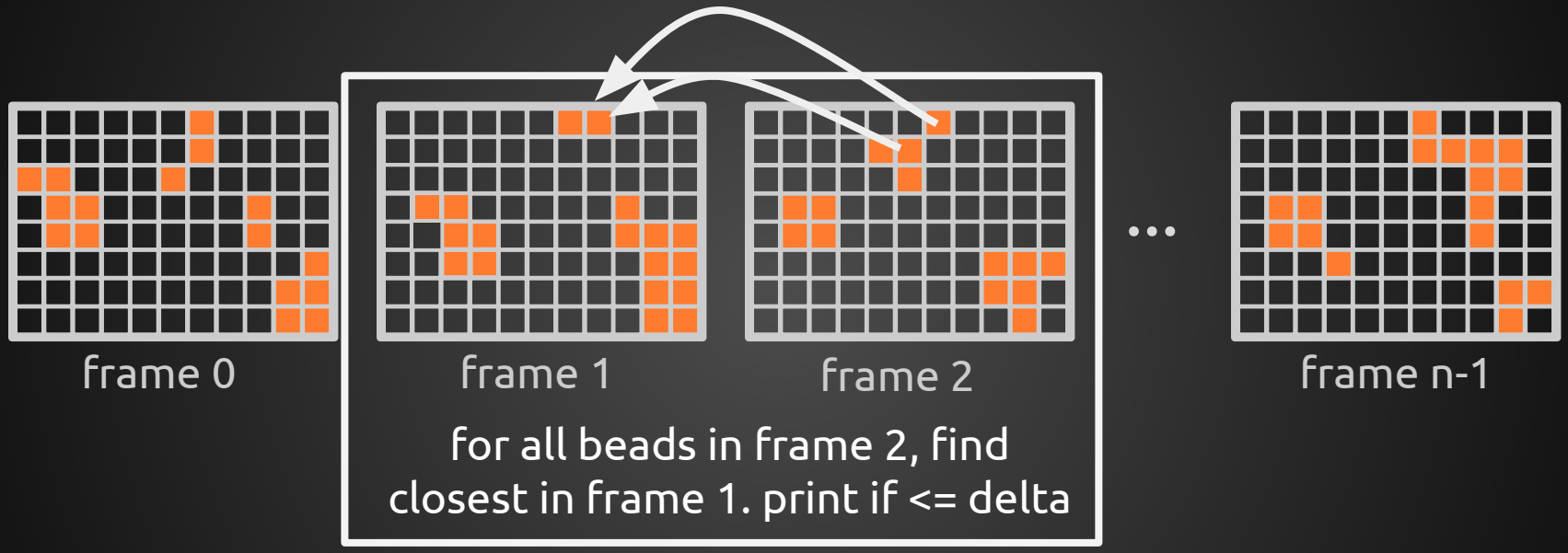
BeadTracker



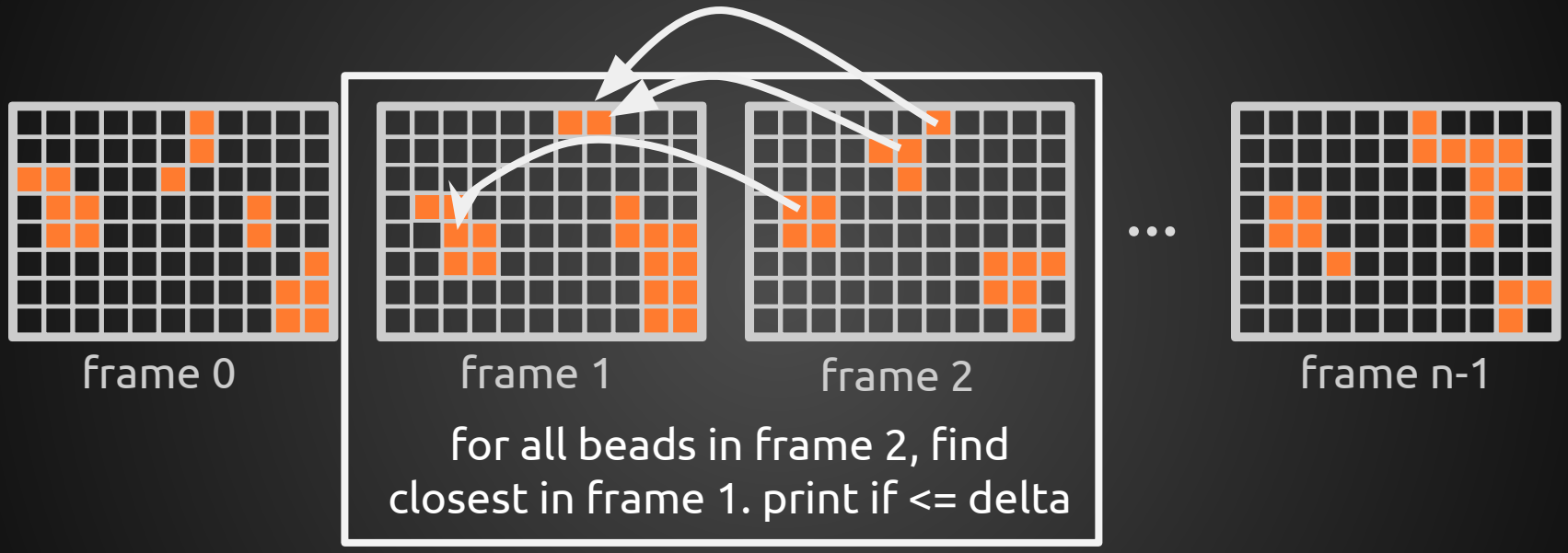
BeadTracker



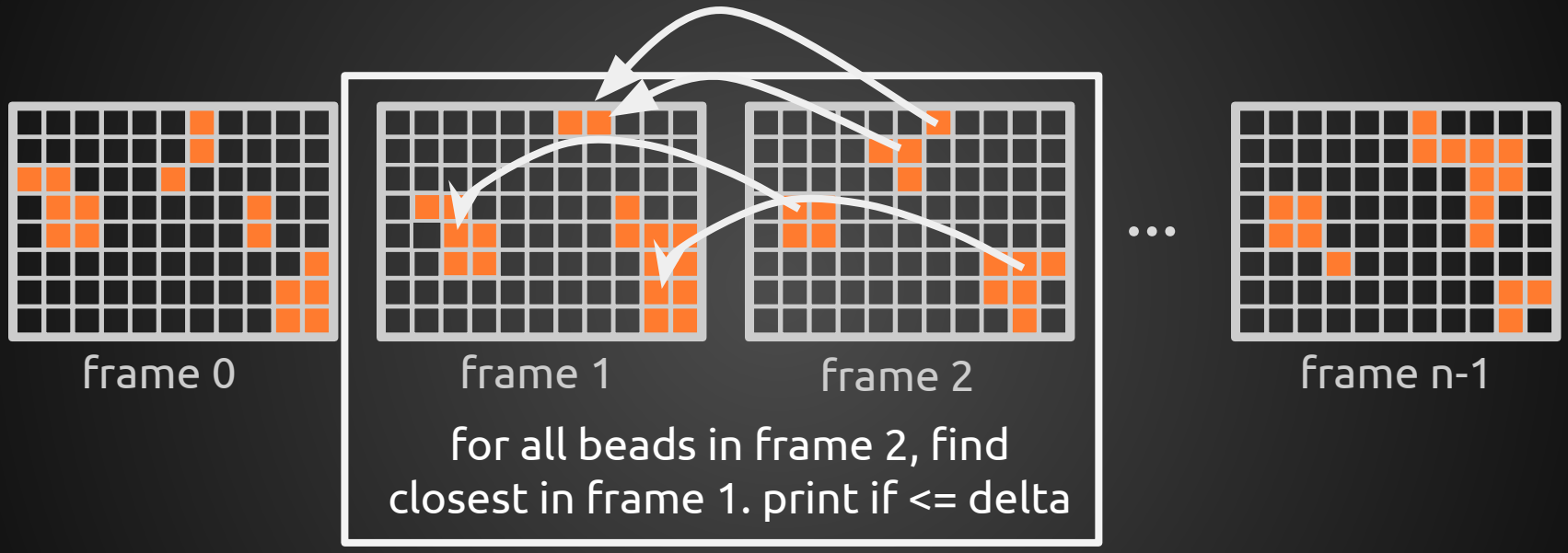
BeadTracker



BeadTracker



BeadTracker



Avogadro.java

Use displacements to calculate Avogadro's #.

Avogadro.java

Use displacements to calculate Avogadro's #.

- Lots of formulas, pretty straightforward.

Avogadro.java

Use displacements to calculate Avogadro's #.

- Lots of formulas, pretty straightforward.
- Be careful with your units!

Avogadro.java

Use displacements to calculate Avogadro's #.

- Lots of formulas, pretty straightforward.
- Be careful with your units!
  - Converting from pixels to meters can be tricky.

Avogadro.java

Use displacements to calculate Avogadro's #.

- Lots of formulas, pretty straightforward.
- Be careful with your units!
  - Converting from pixels to meters can be tricky.
- Can test without the other parts working.

Avogadro.java

Use displacements to calculate Avogadro's #.

- Lots of formulas, pretty straightforward.
- Be careful with your units!
  - Converting from pixels to meters can be tricky.
- Can test without the other parts working.
  - We provide sample input files.

Avogadro.java

Use displacements to calculate Avogadro's #.

- Lots of formulas, pretty straightforward.
- Be careful with your units!
  - Converting from pixels to meters can be tricky.
- Can test without the other parts working.
  - We provide sample input files.
  - Can work on it while waiting for help.

General Tips

- When working on BeadTracker, you may find a bug in BeadFinder. Don't forget to recompile!

General Tips

- When working on BeadTracker, you may find a bug in BeadFinder. Don't forget to recompile!
- Make sure all your corner cases are right.

General Tips

- When working on BeadTracker, you may find a bug in BeadFinder. Don't forget to recompile!
- Make sure all your corner cases are right.
- StackOverflowError = DFS base case issues.

General Tips

- When working on BeadTracker, you may find a bug in BeadFinder. Don't forget to recompile!
- Make sure all your corner cases are right.
- StackOverflowError = DFS base case issues.
- Don't forget the timing analysis in the readme.

General Tips

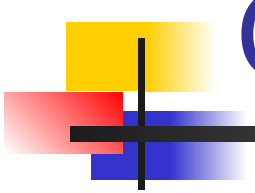
- When working on BeadTracker, you may find a bug in BeadFinder. Don't forget to recompile!
- Make sure all your corner cases are right.
- StackOverflowError = DFS base case issues.
- Don't forget the timing analysis in the readme.
  - BeadTracker is time sink, so analyze that.

General Tips

- When working on BeadTracker, you may find a bug in BeadFinder. Don't forget to recompile!
- Make sure all your corner cases are right.
- StackOverflowError = DFS base case issues.
- Don't forget the timing analysis in the readme.
  - BeadTracker is time sink, so analyze that.
  - How can you run 100 frames?

Amedeo Avogadro

Conclusion: Final Tips



- Avoiding subtle bugs in BeadFinder
  - Double check what happens at corner cases (e.g. at boundary pixels, or when luminance == tau, or mass == cutoff)
- Common errors in BeadFinder
  - `NullPointerException`
  - `StackOverflowError` (e.g., if no base case)
  - No output (need to add print statements)
- Look at *Possible Progress Steps*
  - Click ▸ to expand!

Conclusion: Final Tips

- Avoid *magic numbers*
  - Define named constants
- No Checkstyle or other errors/warnings
- Testing with a `main()`
- There is a limit of **twenty (20) times** that you may click the Check Submitted Files to receive feedback from the TigerFile auto-grader
  - So, test locally! I.e., on your laptop before using TigerFile to run test cases

Conclusion: Final Tips

- Timing analysis - *doubling method!*
  - Wild cards
    - The frames use the following naming convention:
      - `frame00000.jpg, frame00001.jpg …`
        `frame00198.jpg, frame00199.jpg`
    - On command line:
      - 10 frames? `run_1/frame0000*.jpg`
      - 20 frames? `run_1/frame000[0-1]*.jpg`
      - 40 frames? `run_1/frame000[0-3]*.jpg`
      - 100 frames? `run_1/frame000*.jpg`
      - 200 frames? `run_1/frame*.jpg`
      - 400 frames? `run_1/frame*.jpg run_1/frame*.jpg`

Thank you

References and Credits

Primary Color slides prepared by Prof. Ibrahim Albluwi

Black and Orange motif slides prepared by Dr. Dan Leyzberg

https://upload.wikimedia.org/wikipedia/commons/thumb/3/32/Robert_Brown_(botanist).jpg/220px-Robert_Brown_(botanist).jpg

https://cdn.miniphysics.com/wp-content/uploads/2011/01/brownianmotion.gif

https://upload.wikimedia.org/wikipedia/commons/d/d3/Albert_Einstein_Head.jpg

https://en.wikipedia.org/wiki/Jean_Baptiste_Perrin#/media/File:Jean_Perrin_1926.jpg

Amedeo Avogadro