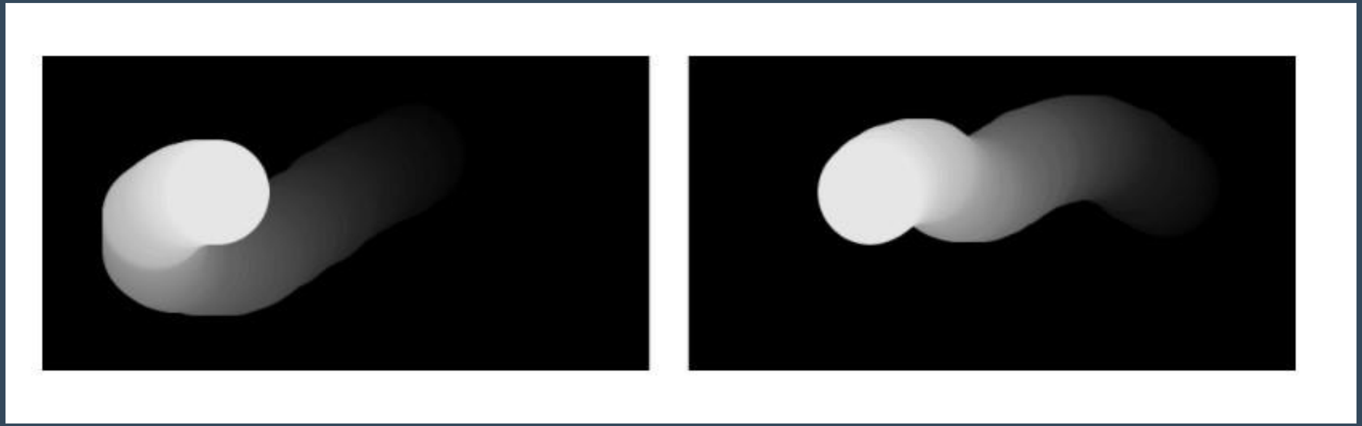
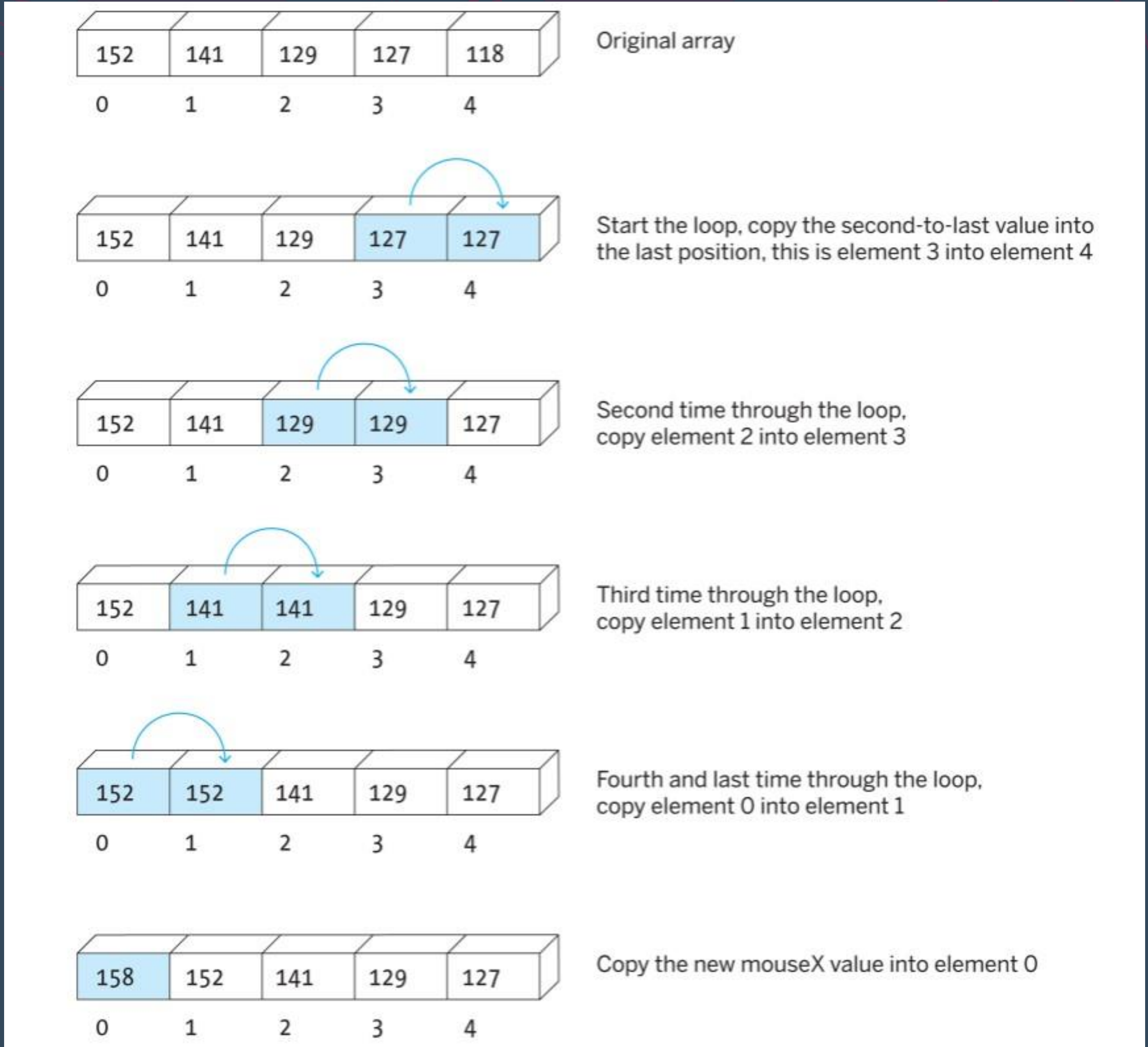
Example 11-9

Storing a shifting buffer of numbers in an array

So cool, right?



How does it happen?



Original array

Start the loop, copy the second-to-last value into the last position, this is element 3 into element 4

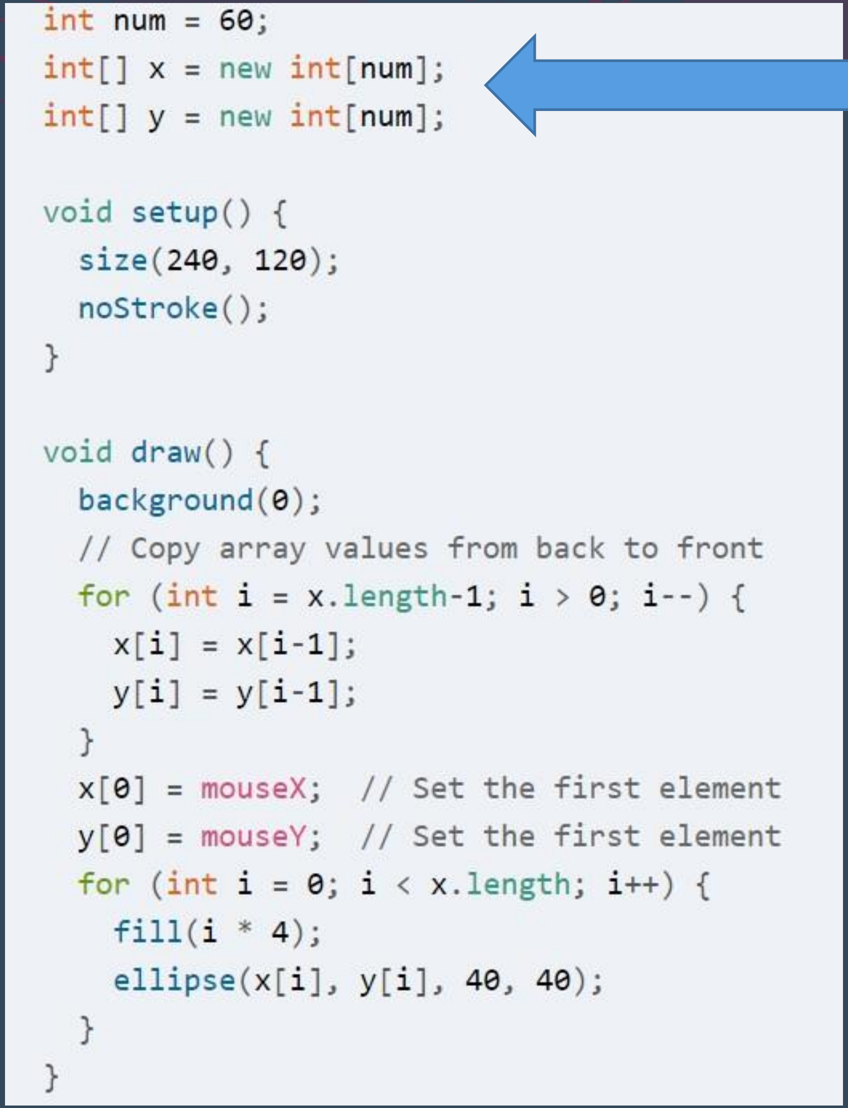
Second time through the loop, copy element 2 into element 3

Third time through the loop, copy element 1 into element 2

Fourth and last time through the loop, copy element 0 into element 1

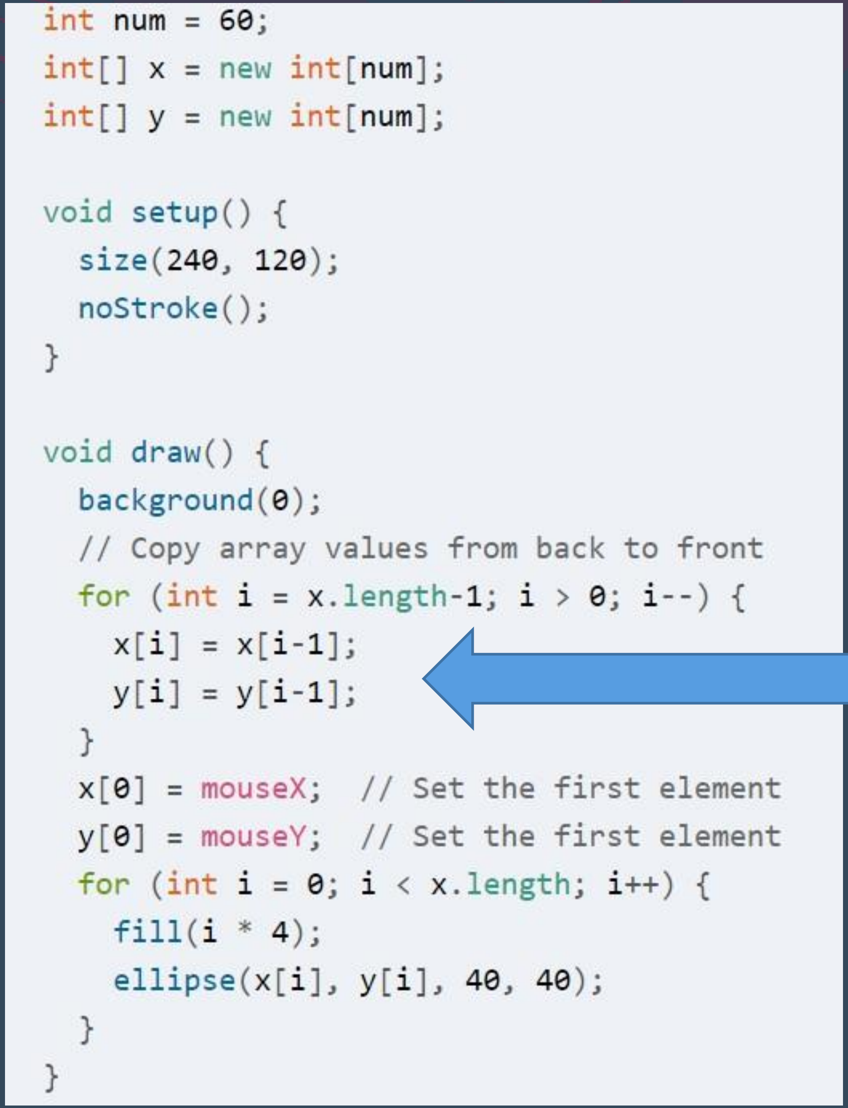
Copy the new mouseX value into element 0

This is a conceptual, step-by-step figure showing how this algorithm works using a simplified array with just 5 elements.



```
int num = 60;
int[] x = new int[num];
int[] y = new int[num];

void setup() {
  size(240, 120);
  noStroke();
}

void draw() {
  background(0);
  // Copy array values from back to front
  for (int i = x.length-1; i > 0; i--) {
    x[i] = x[i-1];
    y[i] = y[i-1];
  }
  x[0] = mouseX;  // Set the first element
  y[0] = mouseY;  // Set the first element
  for (int i = 0; i < x.length; i++) {
    fill(i * 4);
    ellipse(x[i], y[i], 40, 40);
  }
}
```

Here, two integer arrays (x and y) are being declared and instantiated. The integer variable num has already been initialized to 60, so there will be 60 elements in each array.



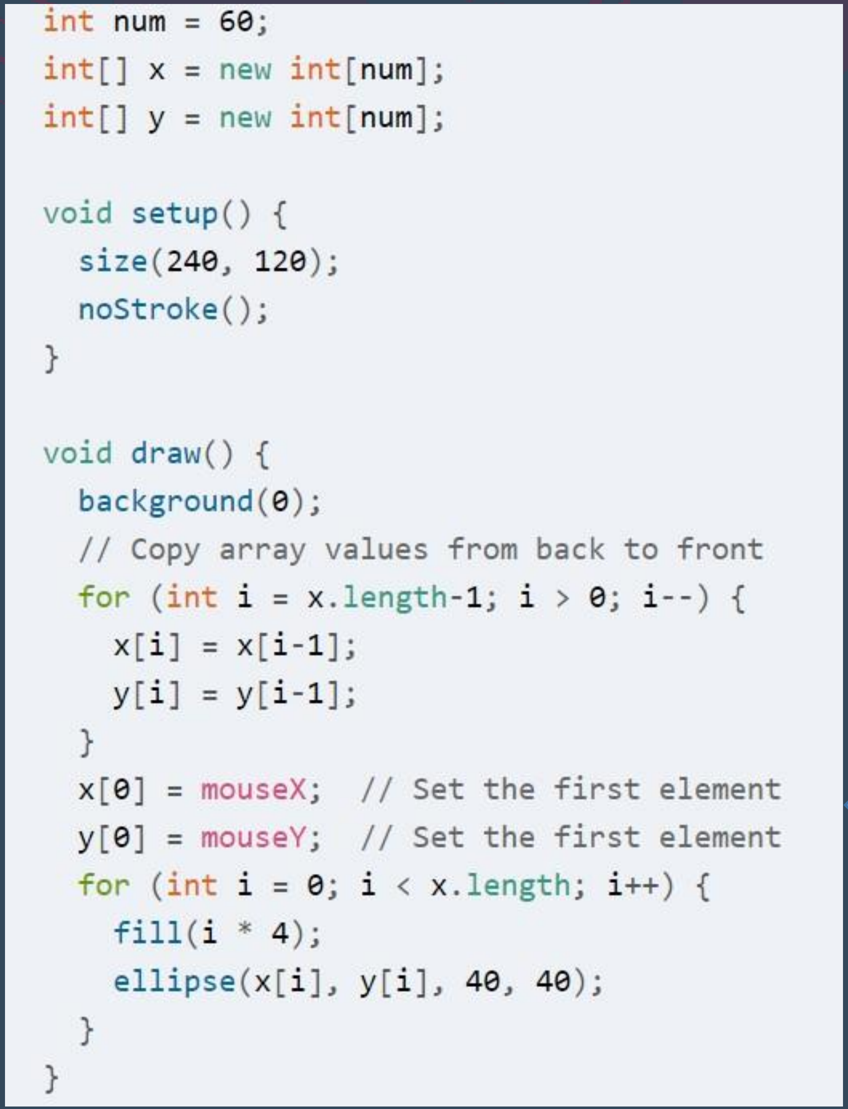
```
int num = 60;
int[] x = new int[num];
int[] y = new int[num];

void setup() {
  size(240, 120);
  noStroke();
}

void draw() {
  background(0);
  // Copy array values from back to front
  for (int i = x.length-1; i > 0; i--) {
    x[i] = x[i-1];
    y[i] = y[i-1];
  }
  x[0] = mouseX;   // Set the first element
  y[0] = mouseY;   // Set the first element
  for (int i = 0; i < x.length; i++) {
    fill(i * 4);
    ellipse(x[i], y[i], 40, 40);
  }
}
```

This loop takes care of *most* of the moving of elements.

For integers from `x.length-1` (so, 59) up to but **not including** 0 (so, 1), subtracting 1 from i each time the loop runs:

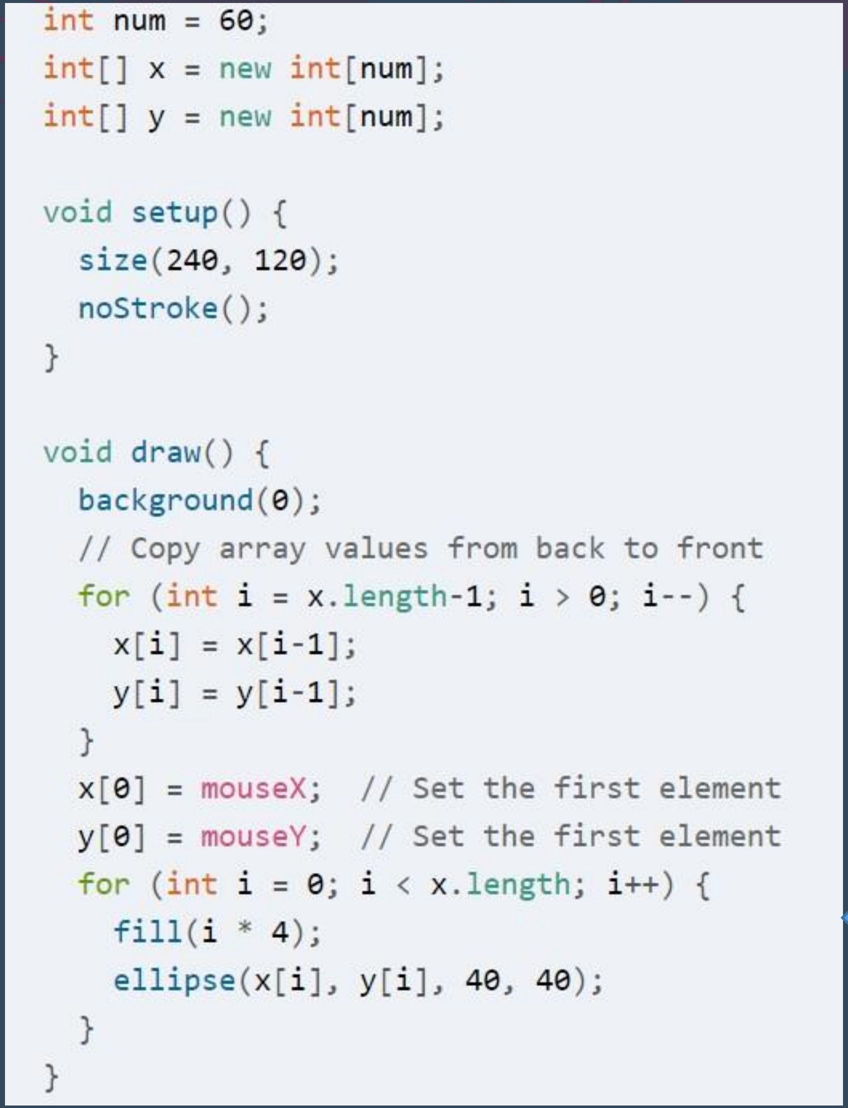
`x[i]` will be replaced with the element from `x[i-1]`, and `y[i]` will be replaced with the element from `y[i-1]`.

*Note that this loop will NOT replace the `x[0]` or `y[0]` element! That's coming up…



```
int num = 60;
int[] x = new int[num];
int[] y = new int[num];

void setup() {
  size(240, 120);
  noStroke();
}

void draw() {
  background(0);
  // Copy array values from back to front
  for (int i = x.length-1; i > 0; i--) {
    x[i] = x[i-1];
    y[i] = y[i-1];
  }
  x[0] = mouseX;  // Set the first element
  y[0] = mouseY;  // Set the first element
  for (int i = 0; i < x.length; i++) {
    fill(i * 4);
    ellipse(x[i], y[i], 40, 40);
  }
}
```

Here, the elements for `x[0]` and `y[0]` are set to be the values of `mouseX` and `mouseY`, respectively.



```
int num = 60;
int[] x = new int[num];
int[] y = new int[num];

void setup() {
  size(240, 120);
  noStroke();
}

void draw() {
  background(0);
  // Copy array values from back to front
  for (int i = x.length-1; i > 0; i--) {
    x[i] = x[i-1];
    y[i] = y[i-1];
  }
  x[0] = mouseX;  // Set the first element
  y[0] = mouseY;  // Set the first element
  for (int i = 0; i < x.length; i++) {
    fill(i * 4);
    ellipse(x[i], y[i], 40, 40);
  }
}
```

This `for` loop takes care of drawing the circles. It will draw 60 circles as it goes from i=0 to i=59.

Why are the circles near end of the "trail" so much closer to white, and why does the trail seem to fade into the mouse position?

Check out that fill! `x[0]` and `y[0]`, which store the mouse position, will result in a fill of 0 *4 = 0 (black), whereas the last position stored, `x[59]` and `y[59]`, will result in a fill of 59*4 = 236, which is approaching 255 (white)!