MiDAS Unemployment System: Algorithm Alchemy Created Lead, Not Gold

“34,000 plus individuals wrongfully accused of unemployment fraud… by Michigan Integrated Data Automated System (MiDAS). A review found that MiDAS adjudicated—by algorithm alone—40,195 cases of fraud, with 85 percent resulting in incorrect fraud determinations.

“As algorithms take on even more decisions in the criminal justice system, in corporate and government hiring, in approving credit and the like, it is imperative that those affected can understand and challenge how these decisions are being made.”

Administrivia

- Assignments:
  - Portfolio Update 1 due tonight (1/31)
  - Creativity Assignment (2/2)

- Midterm in class on Monday, 2/5
  - 1 sheet of notes (2-sided, letter, handwritten)
  - Fill-in-the-blank(s), short answer questions, maybe simple drawing

- Living Computers Museum “field trip” upcoming
Outline

- For-Loop Review
- Nested Loops
- Arrays
  - Arrays and Loops
For-Loop Review

- Loops control a sequence of *repetitions*
  - Do the same thing (or similar things) over and over again

**Examples:** What is changing?

- Common: diagonal lines
  - Change: y-position
- Common: diagonal lines (different thickness per group)
  - Change: x-position
- Common: concentric circles
  - Change: radius/size
- Common: vertical lines
  - Change: transparency/color x-position
Example: Line Gradient

```java
size(400, 400);
background(255);
strokeWeight(5);

for(int i = 0; i < 400; i = i + 8){
  stroke(i);
  line(i, 0, i, 400);
}
```
Exercise: Circle Loop

- Consecutive concentric circles differ in diameter by 10:

```cpp
size(400, 400);
noFill();
also works: int d = 10;
for (int d = 450; d <= 450; d = d + 10) {
    ellipse( width/2, height/2, d, d );
}
```
Example: Looping with User Interaction?

- Draw lines from left side of screen to the horizontal position of the mouse: \( \text{draw lines from } x=0 \text{ to mouseX} \)
Example: Draw Lines to mouseX

```java
void setup() {
    size(400, 400);
    strokeWeight(4);
}

void draw() {
    background(204);

    for(int i = 10; i < mouseX; i = i + 8){
        line(i, 10, i, 390);
    }
}
```
Outline

- For-Loop Review
- **Nested Loops**
- Arrays
  - Arrays and Loops
Nested Loops

- Generally a for-loop has a single loop variable that changes with each iteration

- What if you need/want more things to change?
  - Can nest loops – *i.e.* put a loop inside of another loop
Example: Rectangle Grid

```java
size(400, 400);

for(int y = 20; y < height-20; y = y + 20) {
    for(int x = 20; x < width-20; x = x + 20) {
        rect(x, y, 20, 20);
    }
}
```
Outline

- For-Loop Review
- Nested Loops
- Arrays
  - Arrays and Loops
Arrays

- “Structures” that store many values *of the same* datatype
  - Can think of as a group of related variables

- Arrays store large amounts of data that you can access using a single name
  - Accessing arrays with loops is very convenient
Arrays Terminology

- “Structures” that store many values *of the same datatype*
  - **Element**: a single value in the array
  - **Index**: a number that specifies the location of a particular element of the array
    - Start from 0, so numbered 0 to length - 1
  - **Length**: total number of elements in the array

**Example:**

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>12</td>
<td>49</td>
<td>-2</td>
<td>5</td>
<td>17</td>
</tr>
</tbody>
</table>
Arrays in Processing

- **Declaration:** `type[]` name
  - e.g. `int[]` is array of integers, `color[]` is array of colors

- **Creation:** `new type[num]`
  - e.g. `int[]` `intArr = new int[5];`
  - Default value for all elements is “zero-equivalent” (0, 0.0, `false`, black)
  - Remember that actual indices are from 0 to `num-1`

- **Initialization:** `{elem0, elem1, ..., elemN};`
  - e.g. `int[]` `intArr = {12, 49, -2, 5, 17};`
Arrays in Processing

- **Use element:** `name[index]`
  - In *expression*, uses value of that index of the array (READ)
  - In *assignment*, modifies value of that index of the array (WRITE)

- **Get length:** `name.length`

**Example:**

```java
int[] intArr = {12, 49, -2, 5, 17};
p
```
Arrays Worksheet

- Attach buckets to velcro when array is created
  - Creation of array automatically initializes it

- Access array using notation `ar_name[index]`
  - Indices are numbered starting from 0
  - Use just like you would a variable
  - When WRITING, replace ball in bucket
  - When READING, take ball of same color (leave current one in bucket)
Example: TMNT

```java
// array order: {don, raf, mic, leo}
int[] tmnt_x = {100, 200, 300, 400};
color[] tmnt_c = {color(88, 44, 1410), color(255, 0, 0), color(255, 171, 3), color(...

// draw TMNT using arrays
void draw() {
    background(255, 245, 220);  // paint over drawing canvas
    for (int i=0; i<tmnt_x.length; i=i+1) {
        tmnt(tmnt_x[i], tmnt_c[i]);
    }
```
Example: Index of Smallest Number

**Algorithm:**
- Keep track of the *index* of the smallest number seen so far
  - Start with index 0
- Check each *element* 1-by-1; if number is smaller, then update the smallest index

```c
// returns the index of the smallest number in an array
int find_smallest(float[] list) {
  int smallest = 0;
  for(int i = 0; i < list.length; i = i + 1) {
    if(list[i] < list[smallest]) {
      smallest = i;
    }
  }
  return smallest;
}
```