DRAWING WITH FOR LOOPS

LESSON PLAN

1. Review of previous lessons
   - ➔ The Caterpillar class has two segments: a body and a head
   - ➔ The draw function draws the caterpillar
2. Review of for loops as used in previous classes
3. Student Activity/Peer Instruction
   - ➔ Students explain the functions of three for loops, two being introductory without graphics, and the third loop drawing 20 body functions at the same coordinates
4. Instruction
   - ➔ Version 0 of the Caterpillar class' draw function is introduced, which does not include a for loop but places segments on a linear slope
5. Student Activity/Peer Instruction
   - ➔ Students are asked to write version 1 of the draw function using a for loop to replace repetitive code, version 1 must draw a caterpillar that looks just like version 0 and some are chosen to write theirs on the board
   - ➔ The class determines whether the version 1 draw functions on the board would draw the caterpillar correctly
     - ◆ Students are shown other versions of the draw function and are asked on an individual basis what the caterpillars drawn by these functions would look like
6. Instruction
   - ➔ Students are shown in real time how using different draw functions affects the look of the caterpillar
7. Wrap up/summary

LEARNER PROFILE

Students are 11th-12th graders, this is their second CS class. This class is focused on using the previous year's knowledge and applying it to graphics and more hands on projects. This is an honors class. It is only a few weeks into the semester, and we are reviewing topics from last year as we build new projects using JavaScript

GOALS

1. Students will be able to use for loops to design drawings with repeated elements on a 2D coordinate plane

2. Students will be able to explain why for loops are used and how they are efficient

RESEARCH

Primary Question - What is the most effective way of teaching young high schoolers how to use for loops to design drawings with repeated elements on a 2D coordinate plane?

Manipulation - After reviewing for loops, we taught students two new concepts: how a for loop can be used to draw body segments at the same coordinate position and how adjusting coordinate arguments to body segment functions draw a caterpillar at a linear slope.

Hypothesis - Students will be able to leverage their previous knowledge of for loops to write a draw function that applies both new concepts.

Result - Students wrote their own draw functions and we selected volunteers to write theirs on the board. Students responded well to our methods of instruction because we made sure they were active learners.

Design Challenges
- 15 minutes to teach the lesson, making it difficult to accomplish very much, had to keep it simple
- We overcomplicated the lesson at first, and then oversimplified
- We had some difficulty at first attempting to share code with the class and have them follow along, it didn't work

SUPPLEMENTARY MATERIALS

Established Code Review

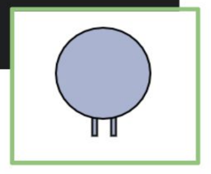
1. We created the caterpillar class which allows us to set the length and radius

```
class Caterpillar {
  constructor(length, r) {
    this.length = length;
    this.r = r;
  }
```

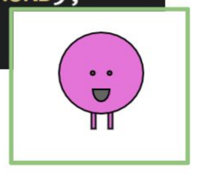
Established Code Review

2. We wrote functions to create a head and a body segment

```
  body(x, y) {
    // fill body with a bright color
    fill(random(100, 255),
random(100, 255), random(100, 255));
    rect(x - this.r * .1, y + this.r
* .3, this.r * .05, this.r * .75);
    rect(x + this.r * .1, y + this.r
* .3, this.r * .05, this.r * .75);
    ellipse(x, y, this.r);
  }
```

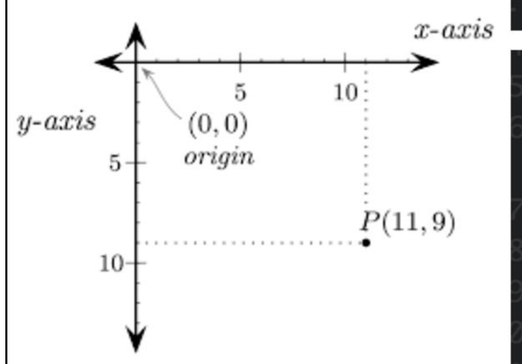


```
  head(x, y) {
    this.body(x, y);
    // fill the eyes with grey
    fill(100);
    ellipse(x - this.r * .1, y,
this.r * .05, this.r * .05);
    ellipse(x + this.r * .1, y,
this.r * .05, this.r * .05);
    arc(x, y + this.r * .2, this.r *
.2, this.r * .3, 0, 180, CHORD);
  }
```



Established Code Review

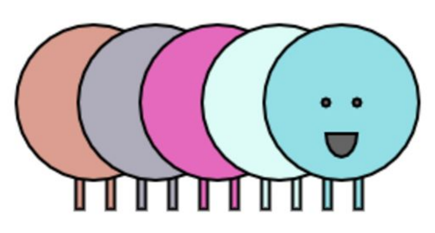
3. We created an instance of the Caterpillar class, set up the canvas, and drew the new caterpillar



```
var larva0 = new Caterpillar(20, 50);
```

```
function setup() {
  createCanvas(windowWidth,
windowHeight);
  ellipseMode(CENTER);
  rectMode(CENTER);
  angleMode(DEGREES);
  larva0.drawV0(100, 150);
}
```

Established Code Review



```
  drawV0(x, y) {
    // draw the a body at the x and y
coordinates given to the function
    this.body(x, y);
    // increment x values by 20 for
the following body segments
    this.body(x + 20, y);
    this.body(x + 40, y);
    this.body(x + 60, y);
    // draw the head
    this.head(x + 80, y);
  }
```

For Loop Examples

```
for (let i = 0; i < 10; i++) {
  print('hi');
}
```

```
for (let i = 10; i > 0; i--) {
  print(i);
}
```

```
for (let i = 0; i < 20; i++) {
  this.body(10, 10);
}
```

```
class Caterpillar {
  constructor(length, r) {
    this.length = length;
    this.r = r;
    this.draw = [
      (x, y) => {
        // draw a body at the x and y coordinates
        // given to the function
        this.body(x, y);
        // draw additional body segments at
        // different locations by changing
        // the x and y coordinates
        this.body(x + 20, y + 5);
        this.body(x + 40, y + 10);
        this.body(x + 60, y + 15);
        this.body(x + 80, y + 20);
        // draw the head
        this.head(x + 100, y + 25);
  },
      (x, y) => {
        for (let i = 0; i < this.length; i++) {
          this.body(x, y);
          x += 20;
          y += 5;
        }
        this.head(x, y);

        // How could we draw caterpillars that are
        // half the length?
  },
      (x, y) => {
        for (let i = 0; i < this.length / 2; i++) {
          this.body(x, y);
          x += 20;
          y += 5;
        }
        this.head(x, y);

        // How could we draw caterpillars that are
        // twice as long?
  },
      (x, y) => {
        for (let i = 0; i < this.length * 2; i++) {
          this.body(x, y);
          x += 20;
          y += 5;
        }
        this.head(x, y);

        // How would the caterpillars look if we
        // decrement 5 from y on each iteration
        // of the for loop instead of increment 5?
  },
      (x, y) => {
        for (let i = 0; i < this.length; i++) {
          this.body(x, y);
          x += 20;
          y -= 5;
        }
        this.head(x, y);

        // How would the caterpillars look if we
        // decrement 20 from x on each iteration
        // of the for loop instead of increment 20?
  },
      (x, y) => {
        for (let i = 0; i < this.length; i++) {
          this.body(x, y);
          x -= 20;
          y += 5;
        }
        this.head(x, y);

        // What value could we change so that we
        // could draw caterpillars with body
        // segments that are really close together?
  },
      (x, y) => {
        for (let i = 0; i < this.length; i++) {
          this.body(x, y);
          x += 1;
          y += 5;
        }
        this.head(x, y);

        // What value could we change so that we
        // could draw caterpillars with body
        // segments that are far together?
  },
      (x, y) => {
        for (let i = 0; i < this.length; i++) {
          this.body(x, y);
          x += 40;
          y += 5;
        }
        this.head(x, y);
  },
      (x, y) => {
        for (let i = 0; i < this.length; i++) {
          this.body(x, y);
          x += this.r * .5;
          y += this.r * .1;
        }
        this.head(x, y);
  },
      (x, y) => {
        for (let i = 0; i < this.length; i++) {
          this.body(x, y);
          x += this.r * .5;
          y += this.r * .1;
        }
        this.head(x, y);
  },
      (x, y) => {
        for (let i = 0; i < this.length; i++) {
          this.body(x, y);
          x += random(this.r * .1, this.r * .7);
          y += random(this.r * .1, this.r * .2);
        }
        this.head(x, y);
  },
      (x, y) => {
        let originalY = y - this.r;
        for (let i = 0, angle = 0; i < this.length; i++, angle += 30) {
          this.body(x, y);
          x += random(this.r * .5, this.r * .75);
          y = cos(angle) * this.r + originalY;
        }
        this.head(x, y);
  }];
  }

  body(x, y) {
    // fill body with a bright color
    fill(random(100, 255), random(100, 255), random(100, 255));
    rect(x - this.r * .1, y + this.r * .3, this.r * .05, this.r * .75);
    rect(x + this.r * .1, y + this.r * .3, this.r * .05, this.r * .75);
    ellipse(x, y, this.r);
  }

  head(x, y) {
    this.body(x, y);
    fill(100); // fill the eyes with grey
    ellipse(x - this.r * .1, y, this.r * .05, this.r * .05);
    ellipse(x + this.r * .1, y, this.r * .05, this.r * .05);
    arc(x, y + this.r * .2, this.r * .2, this.r * .3, 0, 180, CHORD);
  }
}

var larva0 = new Caterpillar(20, 50);
var larva1 = new Caterpillar(20, 25);
var drawVersion = -1;

function setup() {
  createCanvas(windowWidth, windowHeight);
  ellipseMode(CENTER);
  rectMode(CENTER);
  angleMode(DEGREES);
  response(' ');
}

function keyTyped() {
  response(key);
}

function response(key) {
  switch (key) {
    case ' ':
      drawVersion++;
      caterpillar(true);
      break;
    case 'n':
      drawVersion++;
      caterpillar(false);
      break;
    case 'b':
      drawVersion--;
      caterpillar(false);
      break;


    default:
      break;
  }
}

function caterpillar(log) {
  if (drawVersion >= 11) {
    drawVersion = 0;
  }
  if (drawVersion < 0) {
    drawVersion = 10;
  }
  if (log) {
    console.log('version ' + drawVersion + ' of the draw function!');
    console.log(larva0.draw[drawVersion].toString());
  }
  background(255);
  larva0.draw[drawVersion](50, 200);
  larva1.draw[drawVersion](50, 400);
}
```