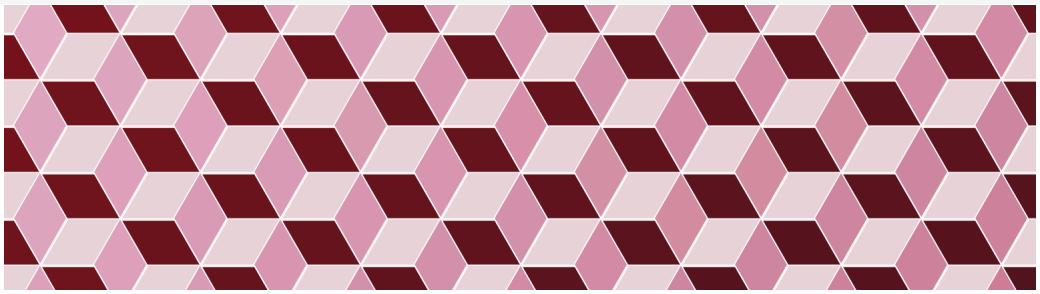


15-104 Introduction to Computing for Creative Practice

Fall 2020

04 Constraints

Instructor: Tom Cortina, tcortina@cs.cmu.edu, GHC 4117, 412-268-3514

1

Print

- Print statements help you display the contents of variables for debugging.
- Calls to print display on the browser console.
- Print statements always print strings. e.g. `print("Hello, world!");`
  - To print number values, use the toString() function on the number.
    e.g. `print(mouseX.toString());`
  - To print out messages that contain strings and numbers, use the + operator for concatenation.
    e.g.

```
print ("mouseX = " + mouseX.toString());
```

2

Bounds

- `min(num1, num2)`
  Returns the minimum of the numbers `num1` and `num2`
  e.g. `min(15, 104)` returns (or evaluates to) `15`
- `max(num1, num2)`
  Returns the maximum of the numbers `num1` and `num2`
  e.g. `max(15, 104)` returns (or evaluates to) `104`
- `constrain(num, low, high)`
  Returns: `num`, if `num` is between `low` and `high` (inclusive)
  `low`, if `num` is less than `low`
  `high`, if `num` is greater than `high`
  e.g. `constrain(238, 15, 104)` returns `104`

3

Constraining the ellipse

```
function setup() {
    createCanvas(200, 200);
    frameRate(10);     // set the frame (refresh) rate to 10 frames per second
}
function draw() {
    background(128);
    fill(255); rect(50,50,100,100);   // box for reference only
    fill(0); // black
    var x = min(mouseX, 150);         // upper limit of x is 150
    ellipse(x, mouseY, 30, 30);
    print(mouseX.toString() + " " + mouseY.toString());
}
```

4

Exercises

- Modify the code so that the *entire* ball stays inside the box right edge.

```
var x = min(mouseX, 135);
```

- Modify the code so that the entire ball stays inside the box bottom edge.

```
var y = min(mouseY, 135);
ellipse(mouseX, y, 30, 30);
```

- Modify the code using `max` to keep the entire ball inside the left edge of the box.

```
var x = max(mouseX, 65);
ellipse(x, mouseY, 30, 30);
```

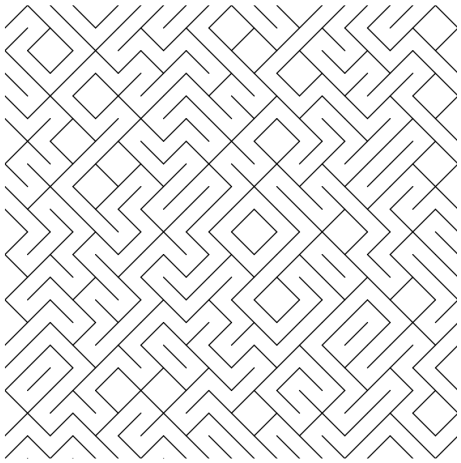
- Modify the code so that the *entire* ball is constrained between the left and right edges of the box.

```
var x = constrain(mouseX, 65, 135);
```

5

Random Lines

```
var x = 0;
var y = 0;
function setup() {
    createCanvas(400, 400);
    background(255);
}
```



6

Random Lines (cont'd)

```
function draw() {
    if (random(1) >= 0.5) {
        line(x, y, x+20, y+20);
    } else {
        line(x, y+20, x+20, y);
    }
    x += 20;
    if (x > width) {
        x = 0;
        y += 20;
    }
    // cont'd in next column
```

```
    if (y > height) {
        background(255);
        x = 0;
        y = 0;
    }
}
```

random(x) is the same as random(0,x)

7

Random Lines (cont'd)

- Each time draw repeats, what basic element does it draw?
- How many choices does it have when it draws this basic element?
- How does the program draw a row of these elements?
- How does the program advance from row to row?
- How does the program "know" when to redraw a new screen?
- How does it make a "blank canvas" to get a clean start?
- How do you make this stop after 5 complete drawings?

8

Fadeout

```
var bright = 0;
function setup() {
    createCanvas(640,360);
}
function draw() {
    if (mouseIsPressed) {
        bright = 0;
    }
    background(bright);
    bright = bright + 1;
}
```

- What variable controls the color of the canvas?
- If you do not click for a long time, what color or grayscale value is used?
- Is there something odd about this?

9

Random Walk

```
var x;
var y;
var walk = 5;

function setup() {
    createCanvas(400, 400);
    x = width / 2;
    y = height / 2;
}
```

```
// cont'd
function draw() {
    background(0);
    ellipse(x, y, 50, 50);
    x += random(-walk, walk);
    y += random(-walk, walk);
}
```

10

Why doesn't this work?

```
var x = width / 2;
var y = height / 2;
var walk = 5;

function setup() {
    createCanvas(400, 400);
}

// cont'd
function draw() {
    background(0);
    ellipse(x, y, 50, 50);
    x += random(-walk, walk);
    y += random(-walk, walk);
}
```

11

Try this:

- For the random motion, if the circle tries to move off of the canvas, have it become blocked and stay where it is instead.
- For the random motion, if the circle tries to move off of the canvas, have it bounce back in the opposite direction the same random walk amount.

12