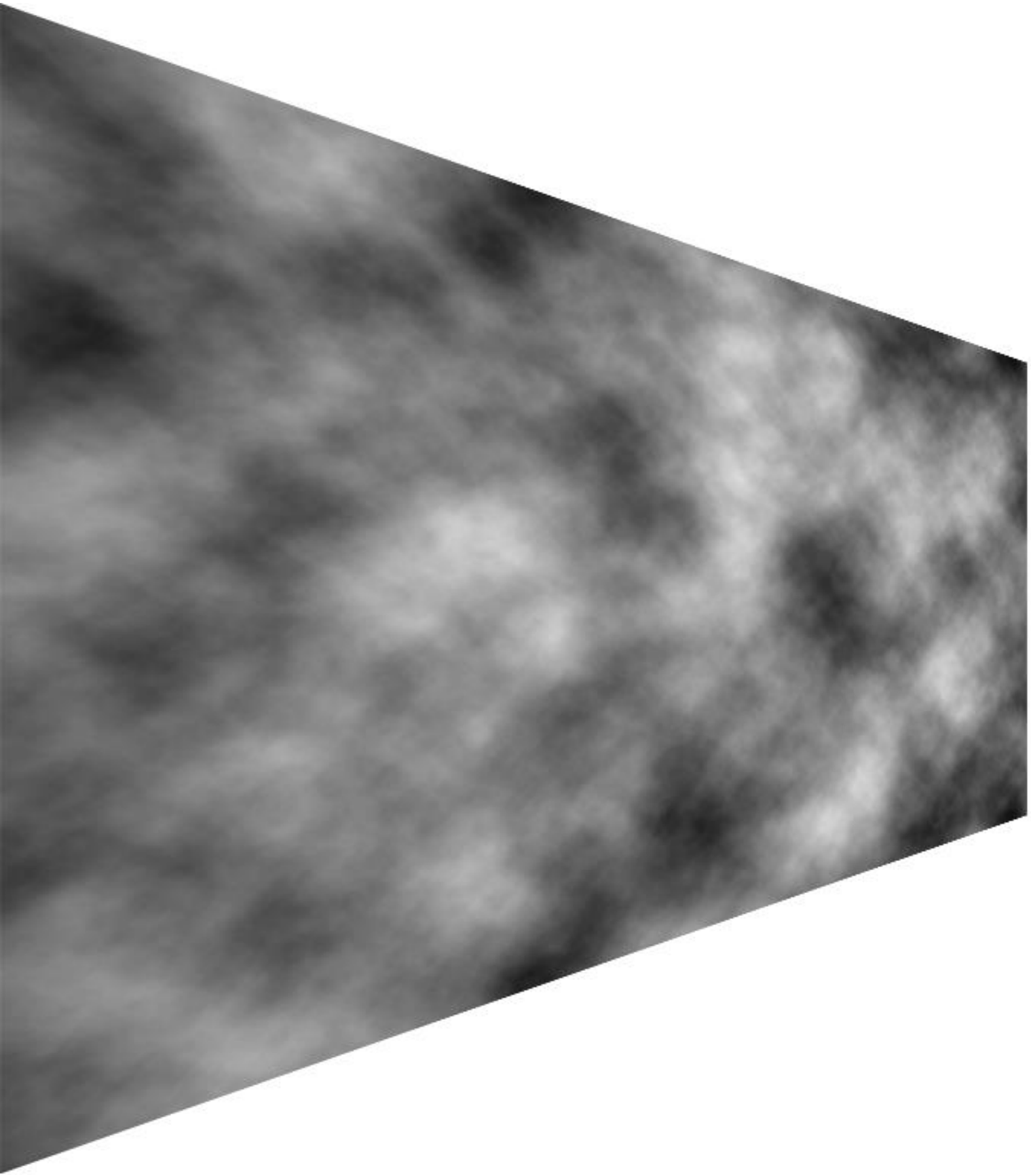


Module 08

Noise

CS 106 Winter 2020

noise()

- Perlin noise is a random sequence generator producing a more natural ordered, harmonic succession of numbers compared to the standard **random()** function.

- It was invented by Ken Perlin in the 1980s and been used since in graphical applications to produce procedural textures, natural motion, shapes, terrains etc.

1D noise()

- Always returns a number between 0-1
- For any given run of your program the same argument always returns the same result.
  - noise(6);
    - Returns a number between 0-1
  - Another call noise(6);
    - Returns the same number

Remember random()

- random(1) returns a number between 0 and 1
- Calling random(1) again returns a different number between 0-1
- random(6) returns a number between 0-6

noise(x) always returns the same number

```
let v;
function setup() {
  let start = 100;
  v = noise(start);// v is between 0 and 1
  print(v);
  v = noise(start);// v is same number as the v above
  print(v);
  v = noise(start);// v is same number as both v above
  print(v);
}
```

Varying the noise() argument
noise() can return similar or dissimilar numbers

```
let v1;
let v2;
let v3;
function setup() {
  let start = 10;

  v1 = noise(start); // returns a number between 0-1

  v2 = noise(start + 0.001);//returns a num close to v1
                            // num is between 0-1 always

  v3 = noise(start + 1);//returns a dissimilar num
                        // number is between 0-1 always

  print(v1, v2, v3);
}
```

Create a smooth line with noise()

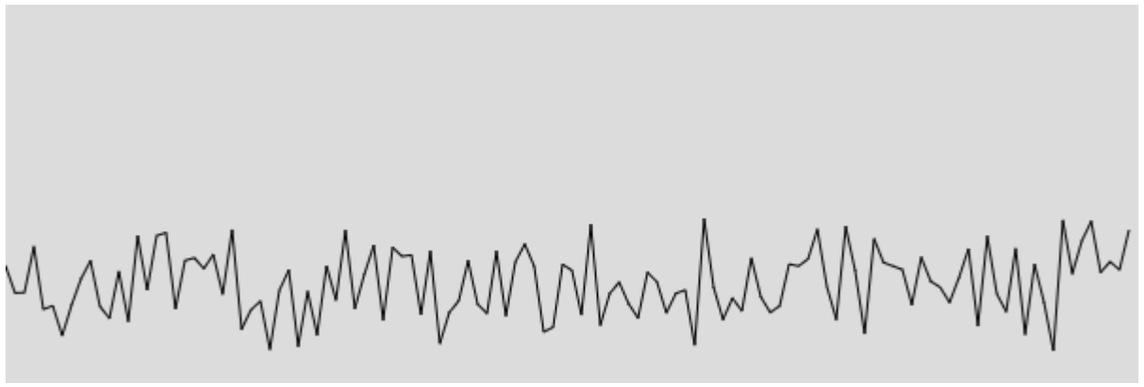
```
// Let's draw a smooth line
function setup() {
  createCanvas(600, 200);
  background(220);
  noFill();
  let v = 10;
  let vInc = 0.05;
  let space = 5;
  let numPoints = width / space;

  beginShape();
  for (let i = 0; i < numPoints; i++) {
    vertex(i * space, height/2 + (noise(v) * 100));
    v = v + vInc;
  }
  endShape();
}
```

Modify the above code: vInc = 0.001;

- The line is not straight. But it doesn’t vary much. It is very smooth.

Modify the above code: vInc = 1.0;

- The line varies a lot. It is not a smooth line.

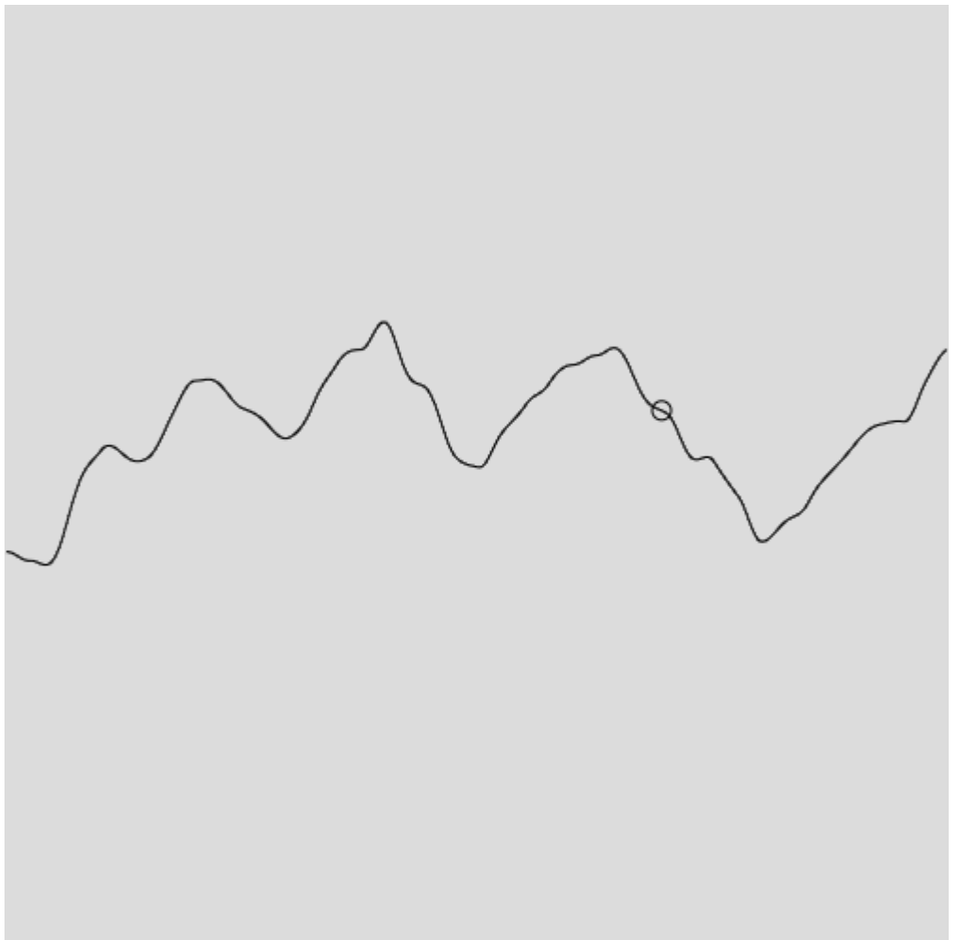


Moving a ball along a noisey line

- Demo code:
  - “BallOnNoiseyLine”

BallOnNoiseyLine (1 of 2)

```
let dx;


let count = 1;
let v;
let vInc = 0.01;
let ballX;
let ballY;


function setup() {
  createCanvas(500, 500);
  noFill();
}
```



BallOnNoiseyLine (2 of 2)

```
function draw() {
  background(220);
  v = 1;
  beginShape();
  for (let i = 1; i < width; i++) {
    let x = i;
    let y = map(noise(v), 0, 1, 100, 400);
    vertex(x, y);
    v = v + vInc;
    if (i === count) {
      ballX = x;
      ballY = y;
    }
  }
  endShape();
  ellipse(ballX, ballY, 10, 10);
  count = (count + 1) % width;
}
```

Demo Code

- Demo code:
  - “Noise1DDirectManip”

Direct Manipulation

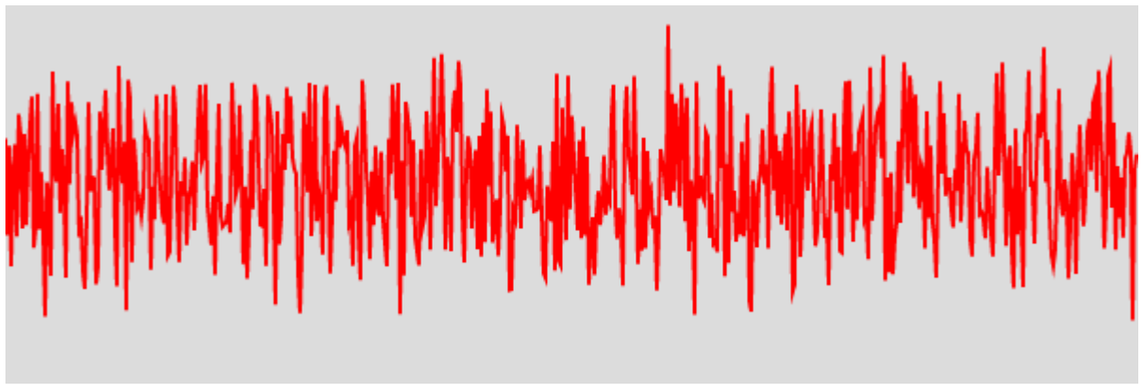
- Use mouseDragged() function
- Calculate movement of the mouse (left-right or right-left)
- Use mouse movement as Direct Manipulation

noise1DDirectManip (1 of 2)

```
let dx;

function setup() {
  createCanvas(600, 200);
  dx = 0;
}
```

noise1DDirectManip (2 of 2)

```
function draw() {
  background(220);
  strokeWeight(2);
  stroke(255, 0, 0);
  noFill();

  beginShape();
  for (let x = 0; x < 600; x++) {
    let v = noise(x - dx);
    let y= map(v, 0, 1, 0, height);
    vertex(x, y);
  }
  endShape();
}

function mouseDragged() {
  dx += mouseX - pmouseX;
}
```



2D Noise

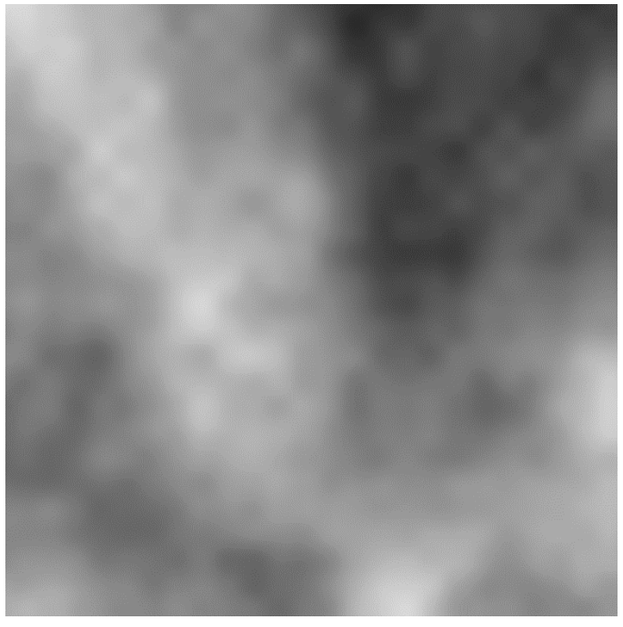
- Go through demo code:
  - “Noise2DDirectManip”

“Noise2DDirectManip” (1 of 2)

```
let tx;
let ty;

// Scaling factor for the noise() function.  Try
// changing this number!
let sc = 100.0;

function setup(){
  createCanvas(300, 300);
}
```

“Noise2DDirectManip” (2 of 2)

```
function draw(){
  background(220);
  for ( let y = 0; y < width; ++y ) {
    for ( let x = 0; x < height; ++x ) {
      let v = noise( (x-tx) / sc, (y-ty) / sc );
      set( x, y, color( v * 256.0 ) );
    }
  }
}

function mouseDragged(){
  tx += mouseX - pmouseX;
  ty += mouseY - pmouseY;
}
```



Goals

- Be able to write short sketches that use the noise() function.

- Understand how noise() works in 1D and 2D, especially 1D.

- Understand the difference between random() and noise().

Which of these expressions is NOT guaranteed to return a number between 0 and 1?

Assume we have the following two lines of code:

```
let a = noise(99.0);
let b = noise(99.01);
```

Assume we have the following line of code:

let a = noise(99.0);

The following 3 clicker questions are about this code:

The following 3 clicker questions are about this code:

The following 3 clicker questions are about this code:

```
createCanvas(400, 100);
let v = noise(10);
let x1 = 100 + (v * 100);
let x2 = x1 + 100;
line(x1, 50, x2, 50);
```

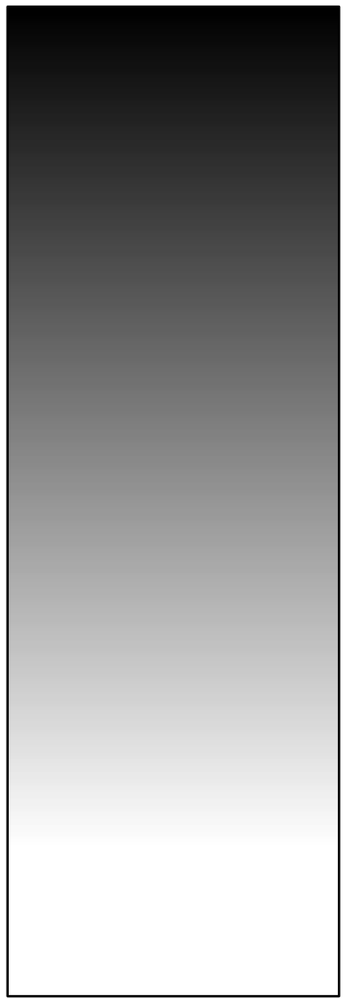
What might the value of “x2” be?

(A) `A number exactly 100 larger than x1`

(B) `A number between 100-200`

(C) `A number between 200-300`

Remember this ex from CS105
“Similar” code is needed in Lab08
Let’s Review the code (next slide)



Draw Gradient:
From CS105 Lecture Slides

```
let shade = 0;

function setup() {
  createCanvas(100, 255);
  background(220);

  for (let y = 0; y <= height; y++) {
    stroke(shade);
    line(0, y, width, y);
    shade += 1;
  }
}
```