PAT 204/504 (Fall 2025)

Creative Coding

Lecture 10: Motion & Physics

Instructor: Hao-Wen Dong

UNIVERSITY OF MICHIGAN

Acceleration

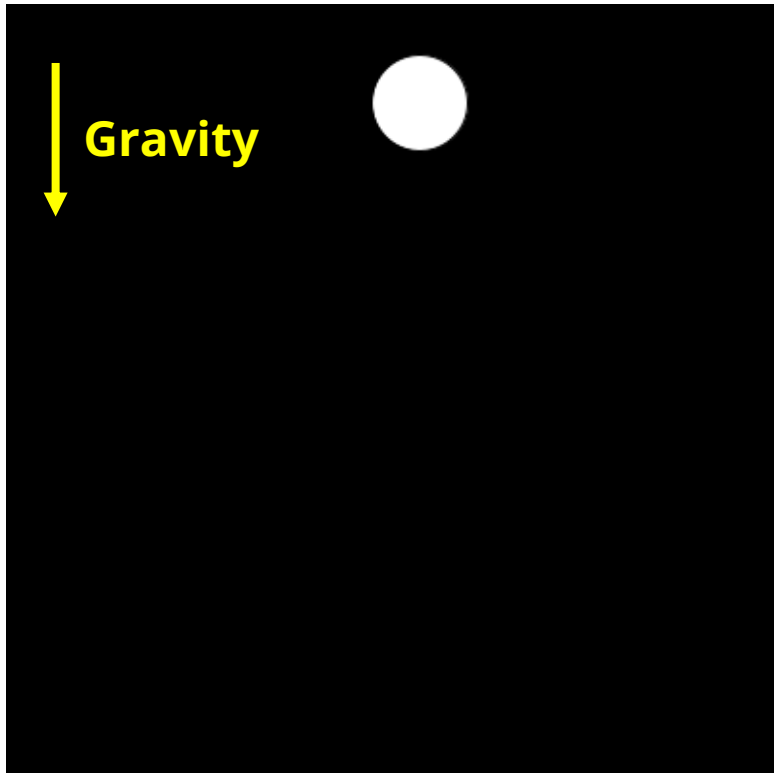
Example: Gravity

```
// Apply gravity to the ball
void applyGravity() {
  speedY += gravity;
}
```

Apply gravity as y-acceleration

```
// Check if the ball hit the walls
void checkWalls() {
  ...

  // Check if the ball hit the top and bottom walls
  if (y > height - radius) {
    speedY = -abs(speedY) * decay;
    y = height - radius;
  } else if (y < radius) {
    speedY = abs(speedY);
    y = radius;
  }
}
```

Reduce the speed a little bit when it hits the bottom wall



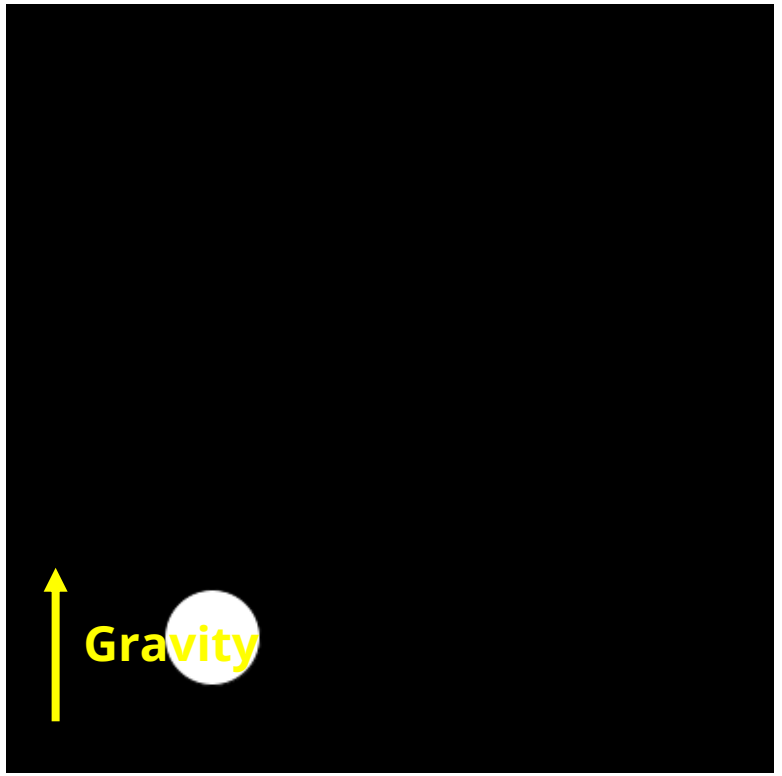
Example: Upside-down Gravity

```
// Apply gravity to the ball
void applyGravity() {
  speedY -= gravity;
}
```

Apply gravity as y-acceleration

```
// Check if the ball hit the walls
void checkWalls() {
  ...

  // Check if the ball hit the top and bottom walls
  if (y > height - radius) {
    speedY = -abs(speedY);
    y = height - radius;
  } else if (y < radius) {
    speedY = abs(speedY) * decay;
    y = radius;
  }
}
```

Reduce the speed a little bit when it hits the top wall

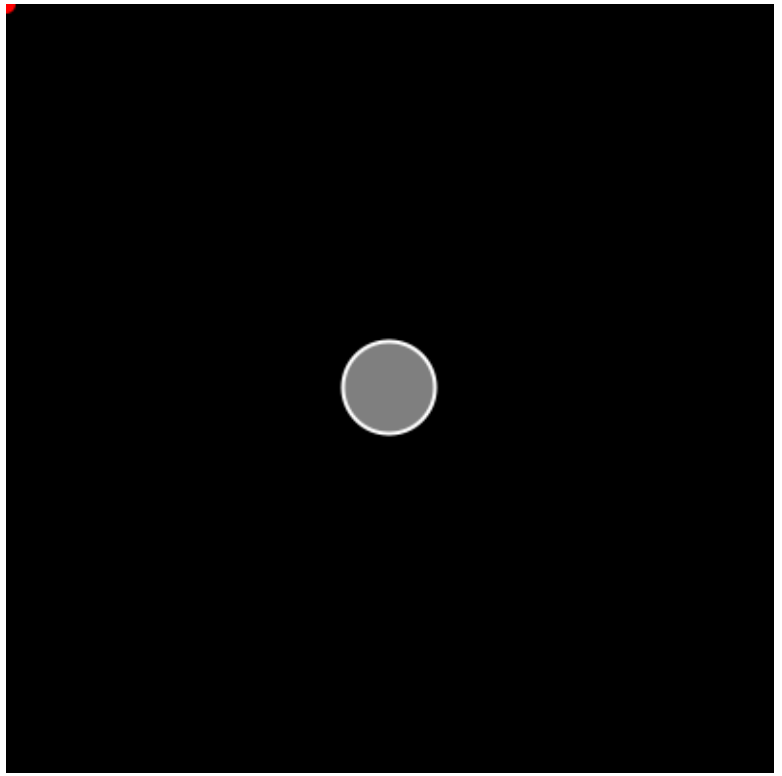


Example: Acceleration

```
class Mover {
  PVector location;
  PVector velocity;
  PVector acceleration;
  float topspeed = 5;

  Mover() {
    location = new PVector(width/2, height/2);
    velocity = new PVector(0, 0);
  }

  void display() {
    stroke(255);
    strokeWeight(2);
    fill(127);
    ellipse(location.x,location.y,48,48);
  }

  ...
}
```

Declare the location, velocity and acceleration vectors

Initialize the ball to be at the center with zero initial speed



processing.org/examples/accelerationwithvectors.html

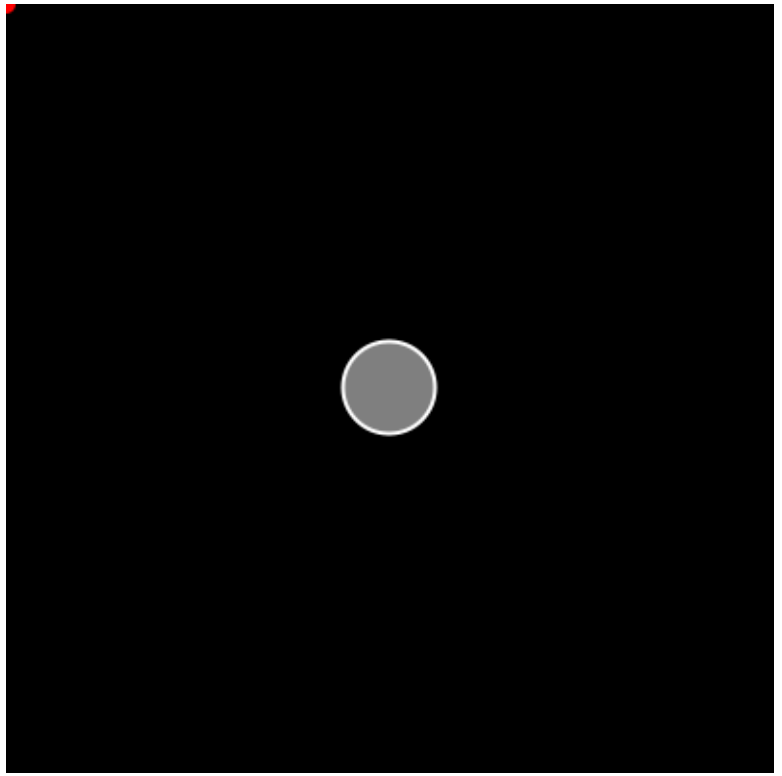
Example: Acceleration

```
class Mover {
  PVector location;
  PVector velocity;
  PVector acceleration;
  float topspeed = 5;

  ...

  void update() {
    PVector mouse = new PVector(mouseX, mouseY);
    PVector acceleration = PVector.sub(mouse, location);
    acceleration.setMag(0.2);

    velocity.add(acceleration);
    velocity.limit(topspeed);

    location.add(velocity);
  }
}
```

Calculate acceleration

Apply the acceleration

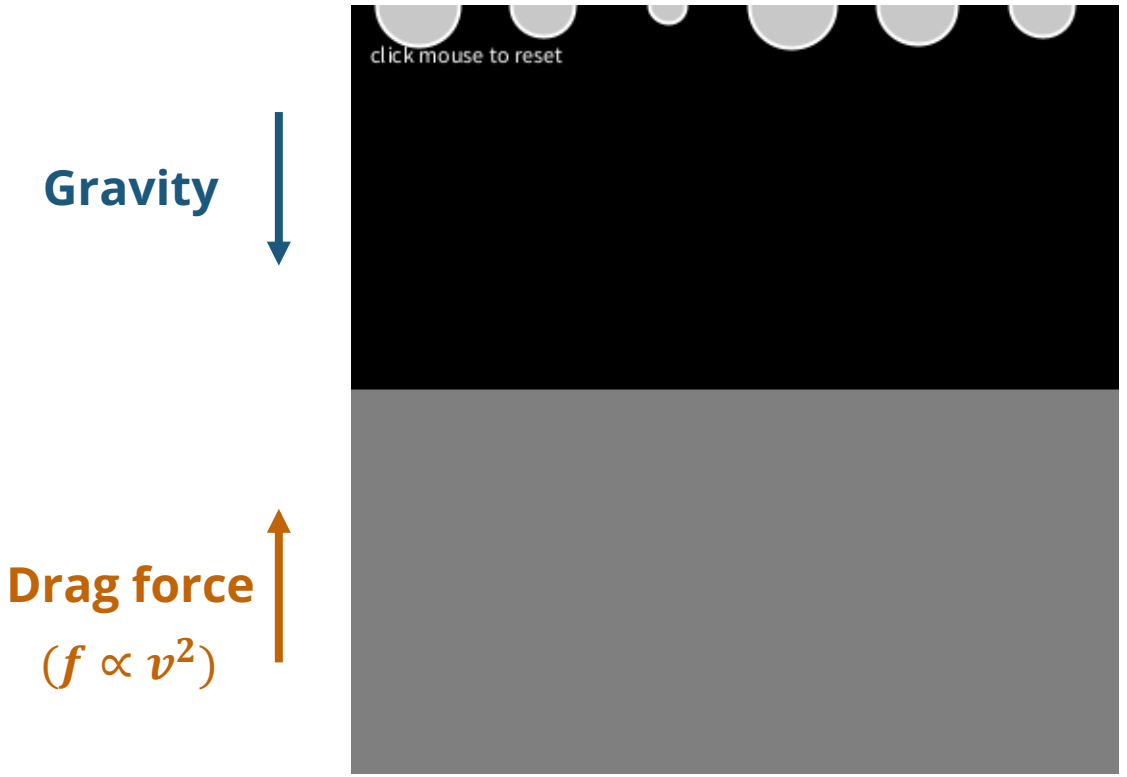
Move the ball



processing.org/examples/accelerationwithvectors.html

Example: Gravity & Fluid Resistance

- Simulating multiple forces
  - Gravity
  - Fluid resistance (drag force)



processing.org/examples/forceswithvectors.html

Example: Purple Rain

- Maintaining an array of **`Drop`** objects
- Each **`Drop`** object
  - Falls with gravity
  - Random initial velocity
  - Random stroke weight (illusion of distance)

Collision

Example: Bouncing Balls

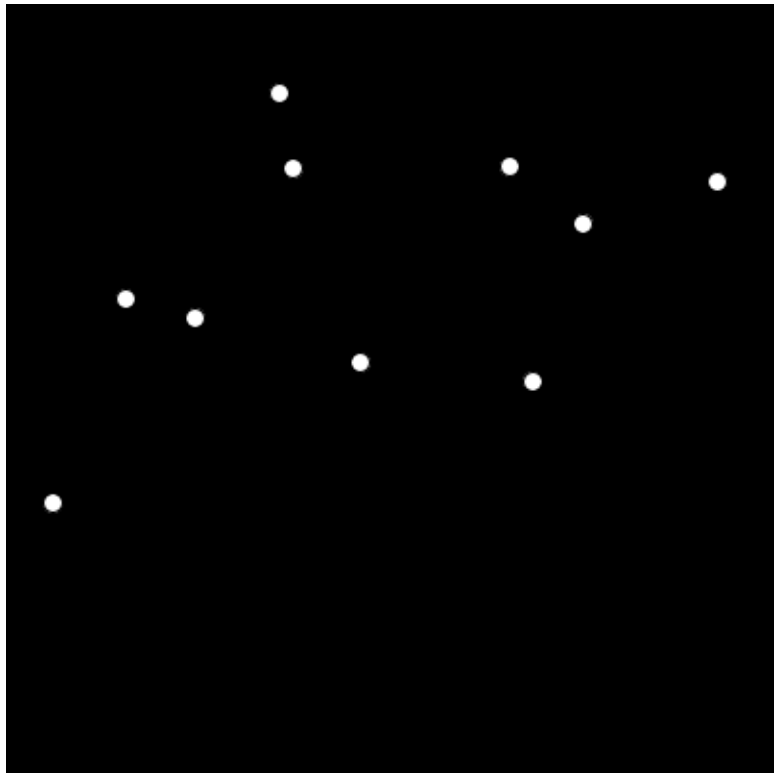
```
Ball[] balls = new Ball[20];
```

An array of objects

```
void setup() {
  size(400, 400);

  for (int i = 0; i < balls.length; i++) {
    balls[i] = new Ball();
  }
}

void draw() {
  background(0);

  for (int i = 0; i < balls.length; i++) {
    balls[i].move();
    balls[i].checkWalls();
    balls[i].show();
  }
}
```

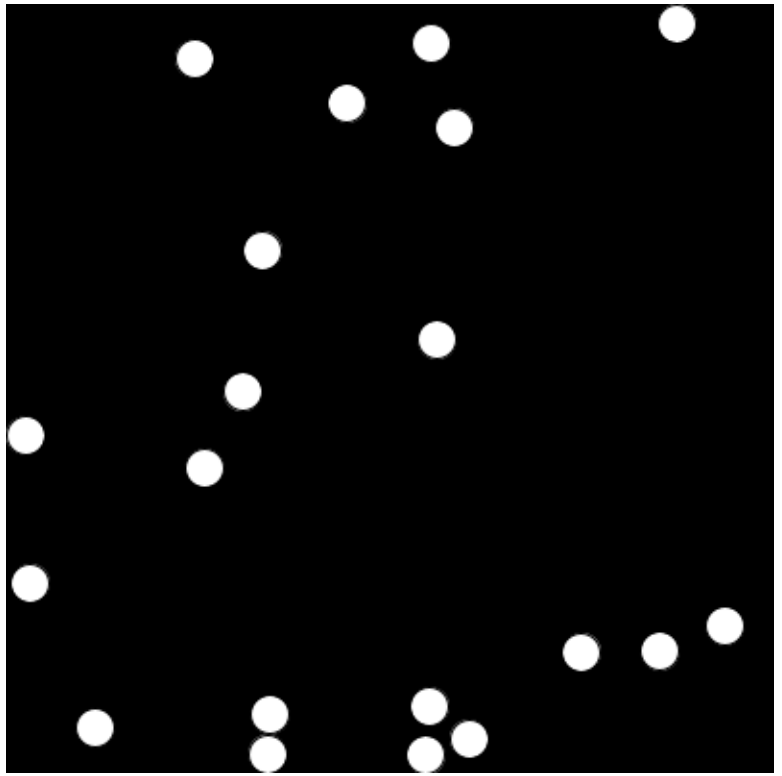
Initialization

Call the methods!



Example: Bouncing Balls with Collision Detection

- What else do we need?
- How to check if the ball collides with another?
- What kind of function do we need?



```
Ball[] others;
```

An array to store other balls

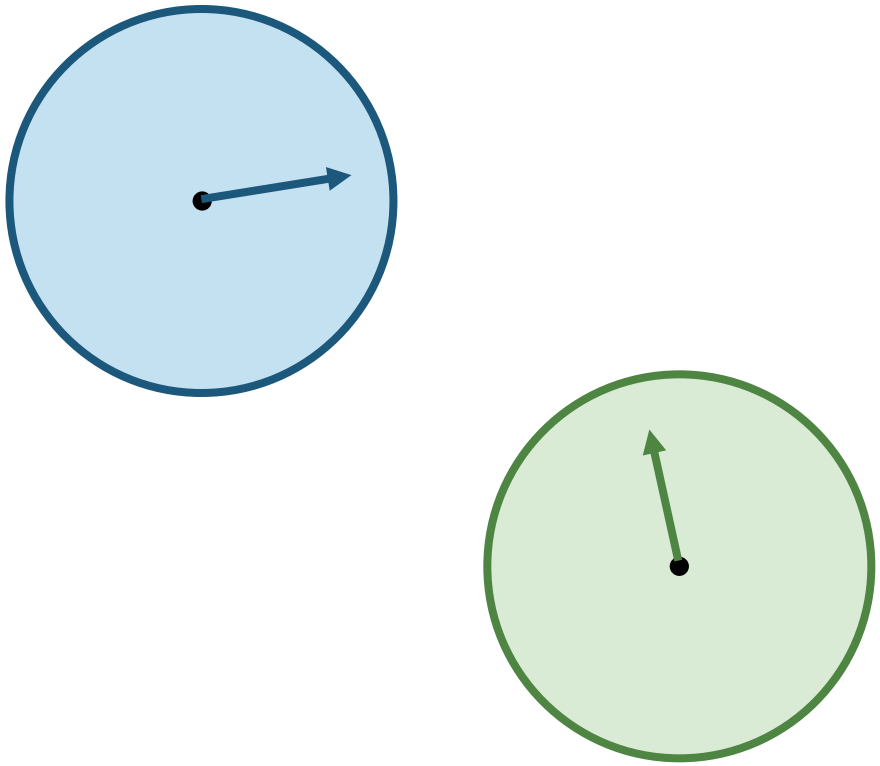
```
void checkCollisions() {
  for (Ball other: others) {
    collide(other);
  }
}
```

Iterate over other balls to check if there's a collision

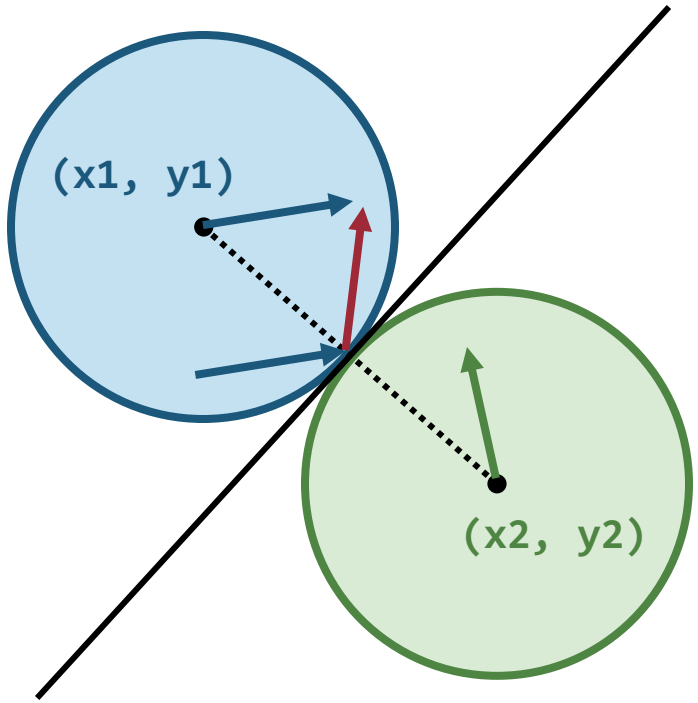
```
void collide(Ball other) {
  ???
}
```

Check if there's a collision

Handling Collisions



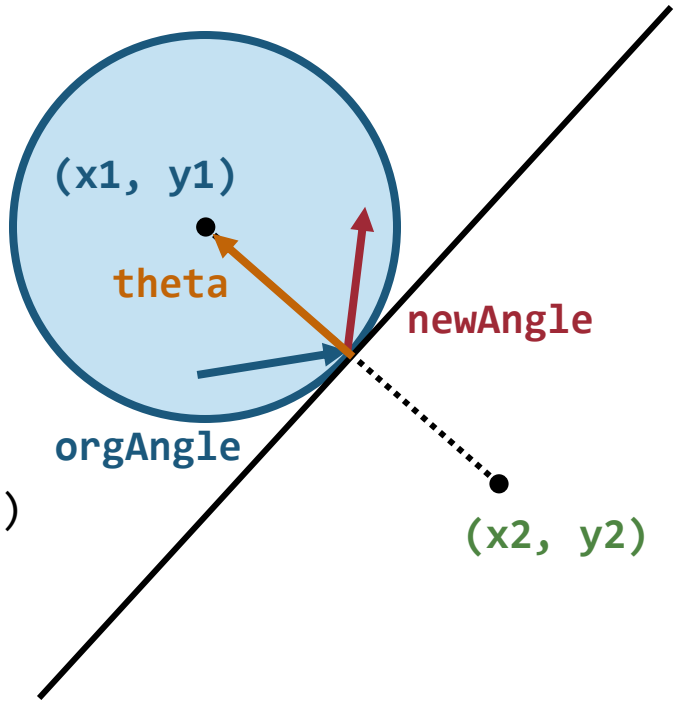
Handling Collisions

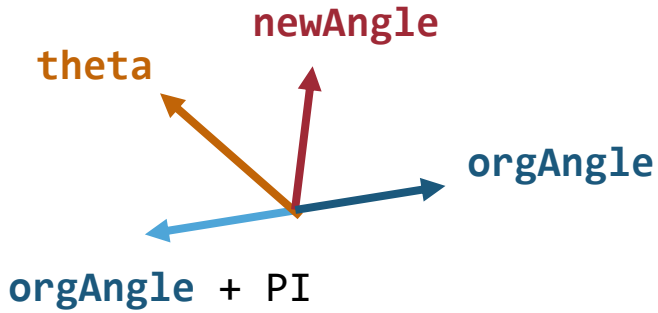


Handling Collisions

```
theta = atan2(y2 - y1, x2 - x1)
```

```
orgAngle = atan2(speedY, speedX)
```






```
newAngle = (orgAngle + PI) – 2 (orgAngle + PI – theta)
         = 2 theta - PI - orgAngle
```

Example: Bouncing Balls with Collision Detection

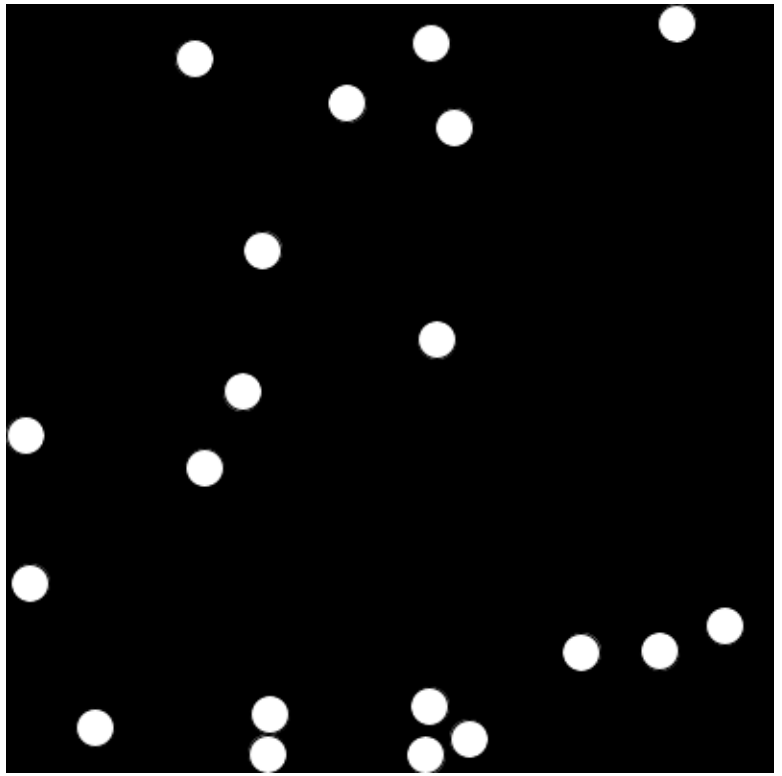
```
void collide(Ball other) {
  if (other == this) return;
  float dist = dist(x, y, other.x, other.y);
  if (dist >= size) return;
  x -= speedX;
  y -= speedY;
  float theta = atan2(other.y - y, other.x - x);
  float orgAngle = atan2(speedY, speedX);
  float newAngle = (theta - PI + theta - orgAngle);
  speedX = speed * cos(newAngle);
  speedY = speed * sin(newAngle);
}
```

Do nothing if it's the same ball

Do nothing if they do not collide

Revert the ball back to where it was before the collision

Find the velocity after the collision



Elastic Collisions

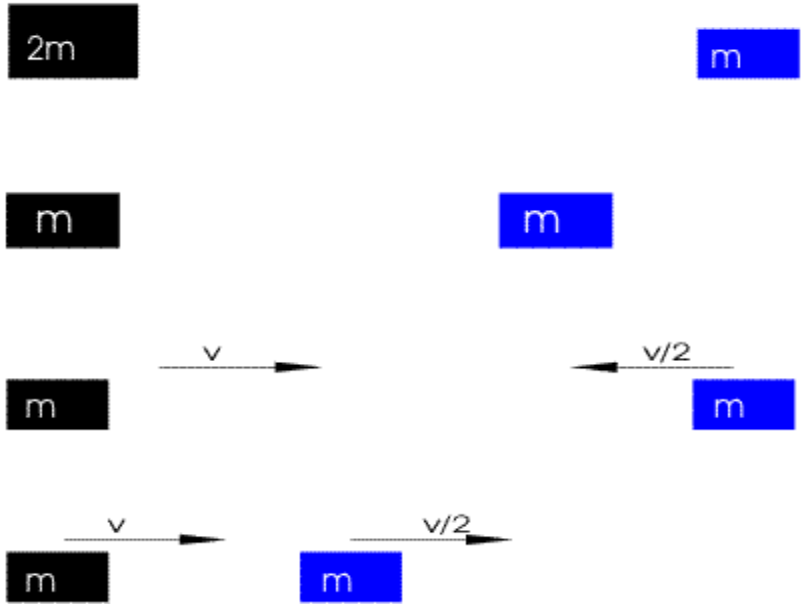
Conservation of momentum

$$m_1 v_1 + m_2 v_2 = m_1 v_1' + m_2 v_2'$$

Conservation of kinetic energy

$$m_1 v_1 + m_2 v_2 = \frac{1}{2} m_1 {v_1'}^2 + \frac{1}{2} m_2 {v_2'}^2$$

(Does not hold for inelastic collision)



(Source: Raul Roque, via Wikimedia Commons)

2D Elastic Collisions

Conservation of momentum

$$m_1\vec{v}_1 + m_2\vec{v}_2 = m_1\vec{v}_1' + m_2\vec{v}_2'$$

Conservation of kinetic energy

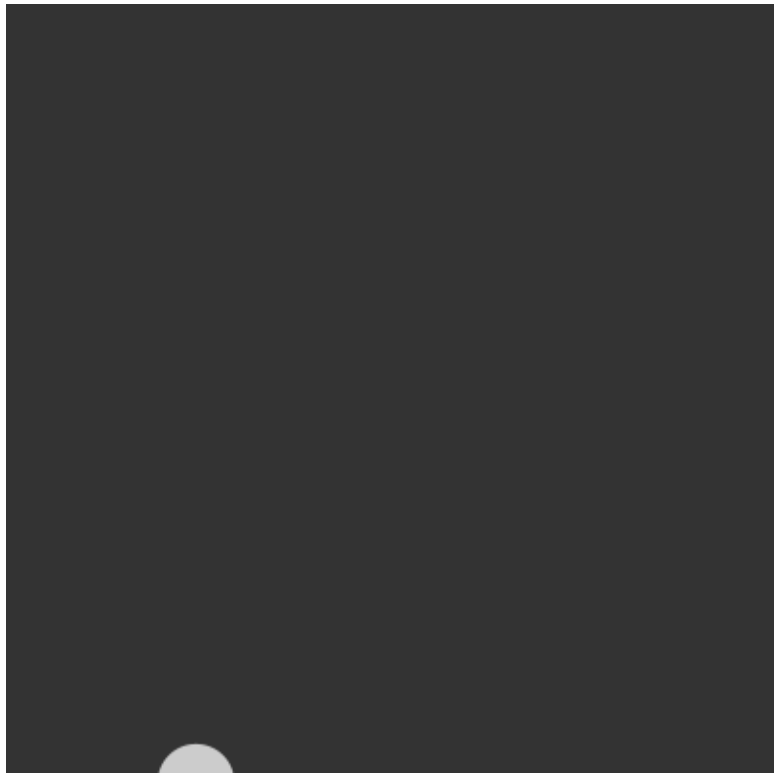
$$m_1\vec{v}_1 + m_2\vec{v}_2 = \frac{1}{2}m_1\vec{v}_1'^{\,2} + \frac{1}{2}m_2\vec{v}_2'^{\,2}$$

Conservation of angular momentum

(Source: Simon Steinmann, via Wikimedia Commons)

Example: Collision with Weights

- Different **weights** and different **speeds**
- Solve the equations to find the new velocities
  - Conservation of momentum
  - Conservation of kinetic energy
  - Conservation of angular momentum



processing.org/examples/circlecollision.html

Example: Reflection

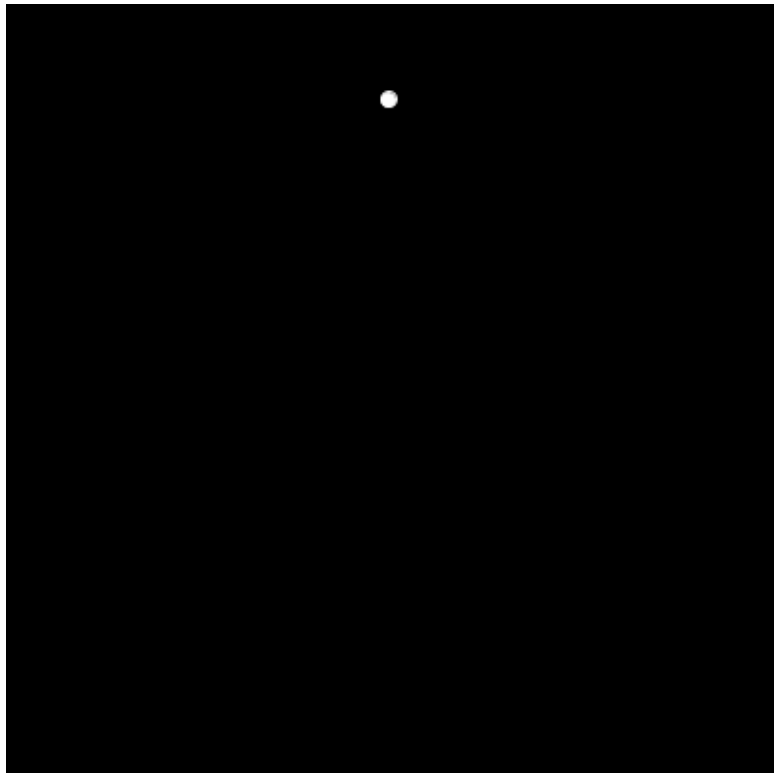
- Several `Ground` objects
  - Initialized with random widths and slopes
  - Together compose the ground surface
- An `Orb` object
  - Initialized with a small x-velocity
  - Influenced by gravity
  - May collide with the walls and the grounds

processing.org/examples/reflection2.html

Particle Systems

Example: Simple Particle System

- A `Particle` object
  - Falls with gravity
  - Fades out over time (die after a predefined lifespan)
- A `ParticleSystem` object is an ArrayList of `Particle` objects (i.e., `ArrayList<Particle>`)
  - Allows storing a variable number of particles
- Show all the `Particle` objects at each call of the `draw()` function



Example: Fireworks

- A `Firework` object
  - Starts as one single `Particle` object
    - Initialized with random force up
    - Flies up with gravity slowing it down
    - Explodes when speed reaches zero
  - Becomes many `Particle` objects after explosion
    - Initialized with random forces towards random directions
    - Fall with gravity
    - Die after invisible on the canvas

Some More Things Worth Knowing

PShape: Object-oriented Interface for Custom Shapes

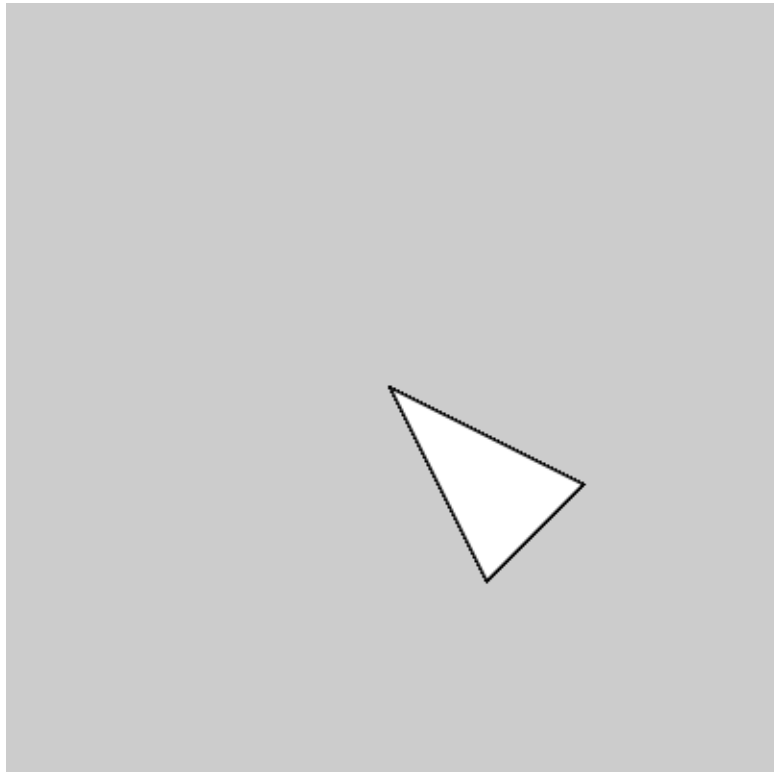
```
PShape s;
```
Declare the shape

```
void setup() {
  size(400, 400);

  s = createShape();
  s.beginShape();
  s.vertex(0, 0);
  s.vertex(50, 100);
  s.vertex(100, 50);
  s.endShape(CLOSE);
}

void draw() {
  shape(s, 200, 200);
}
```

Create the shape

Define the shape

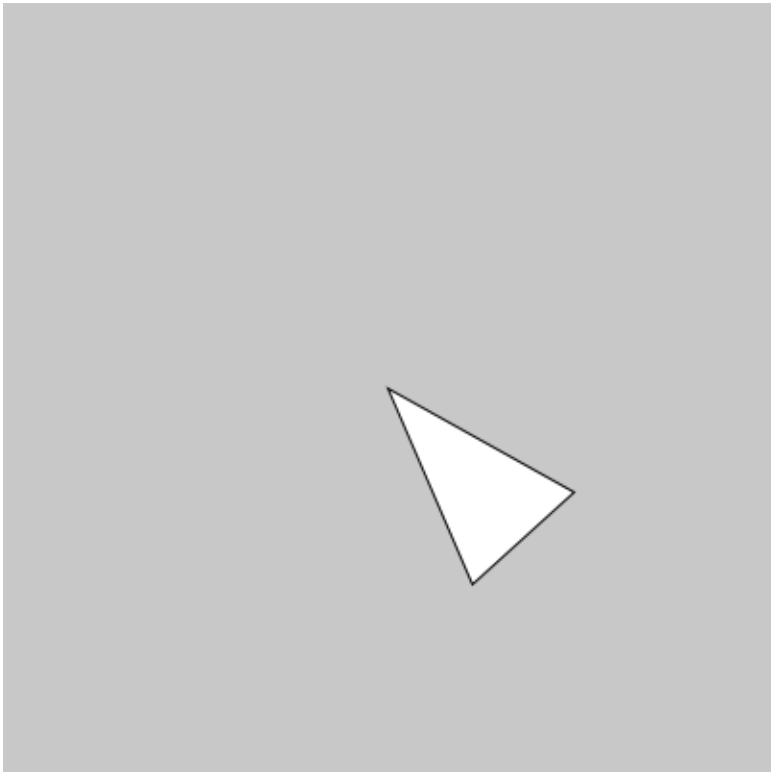
Show the shape



PShape: Object-oriented Interface for Custom Shapes

```
PShape s;

void setup() {
  size(400, 400);

  s = createShape();
  s.beginShape();
  s.vertex(0, 0);
  s.vertex(50, 100);
  s.vertex(100, 50);
  s.endShape(CLOSE);
}

void draw() {
  background(200);
  s.rotate(0.05);
  shape(s, 200, 200);
}
```

Rotate the shape



PShape: Object-oriented Interface for Custom Shapes

- **`loadShape()`** Load a shape from an SVG or OBJ file

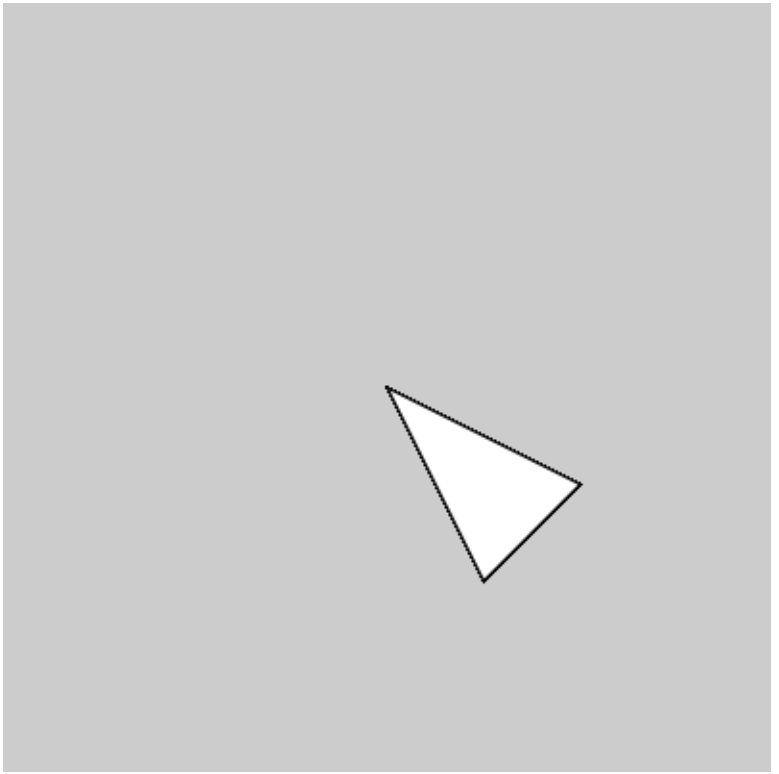
- **`translate()`** Translate the shape
- **`rotate()`** Rotate the shape
- **`scale()`** Scale the shape
- **`resetMatrix()`** Reset the transformation matrix of the shape

PShape: Object-oriented Interface for Custom Shapes

```
PShape s;

void setup() {
  size(400, 400);
  noLoop();

  s = createShape();
  s.beginShape();
  s.vertex(0, 0);
  s.vertex(50, 100);
  s.vertex(100, 50);
  s.endShape(CLOSE);
}

void draw() {
  stroke(#FF0000);
  shape(s, 200, 200);
}
```

Does not apply to PShape

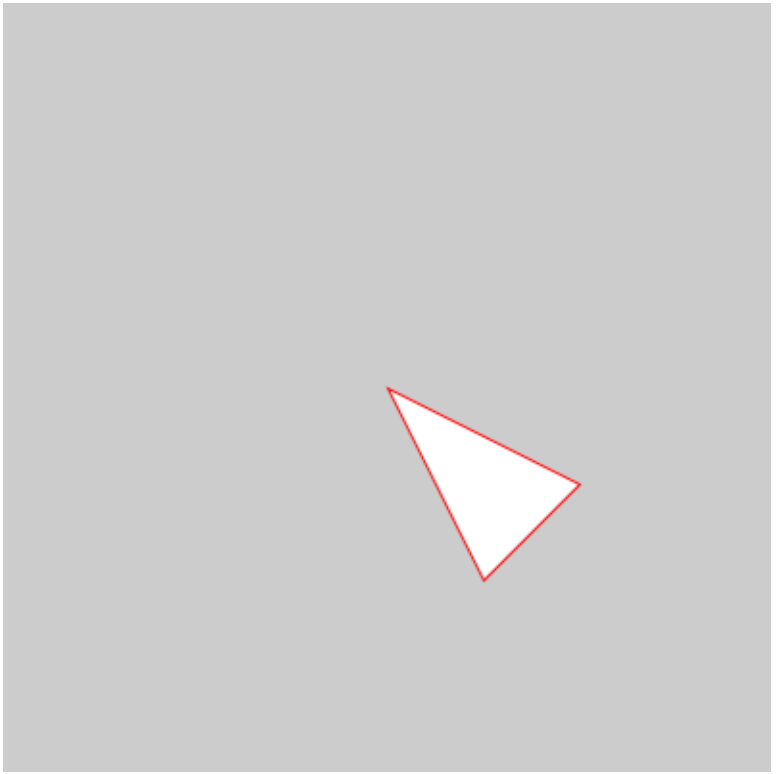
~~Set the stroke color~~



PShape: Object-oriented Interface for Custom Shapes

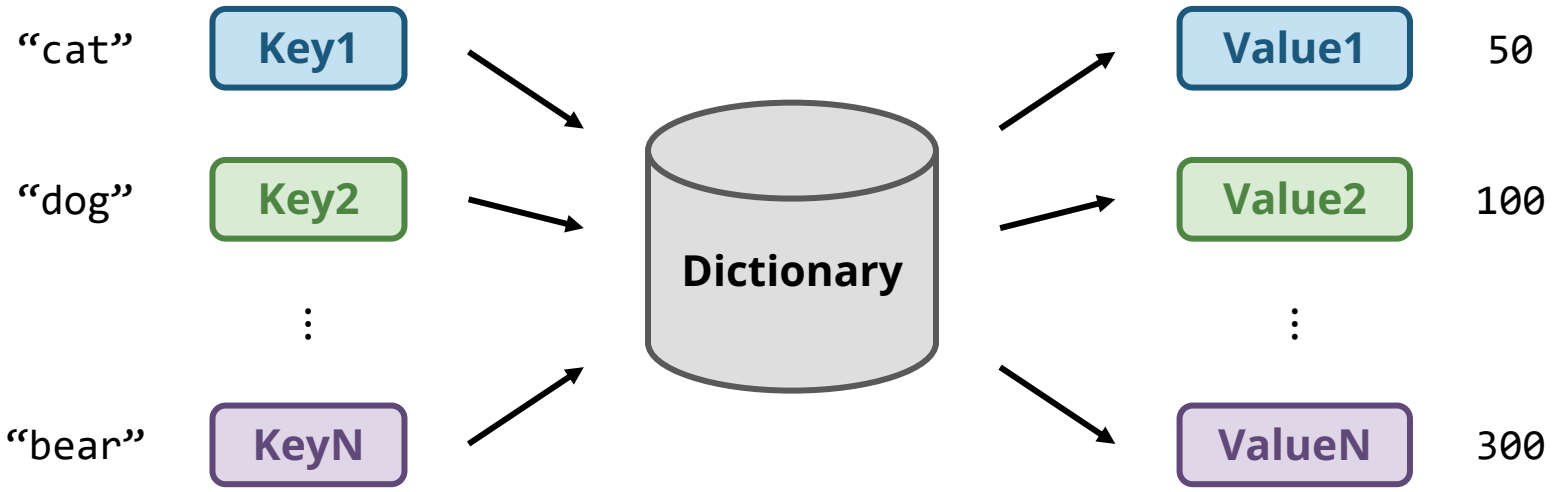
```
PShape s;

void setup() {
  size(400, 400);
  noLoop();

  s = createShape();
  s.beginShape();
  s.vertex(0, 0);
  s.vertex(50, 100);
  s.vertex(100, 50);
  s.endShape(CLOSE);
}

void draw() {
  s.setStroke(#FF0000);
  shape(s, 200, 200);
}
```

Object-oriented!



Dictionary

- A dictionary supports **fast lookup** of the corresponding "**value**" for a "**key**"
- **`IntDict`** String-to-integer mapping
- **`FloatDict`** String-to-float mapping
- **`StringDict`** String-to-string mapping



IntDict: String-to-Integer Mapping

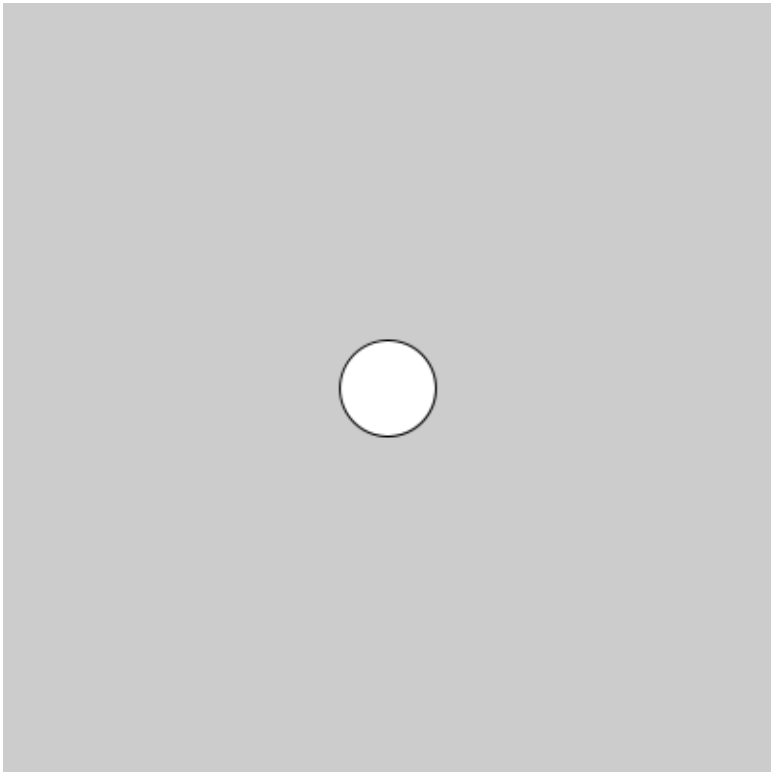
```
IntDict weightMap = new IntDict();
```

Declare the shape

```
void setup() {
  size(400, 400);
  noLoop();

  weightMap.set("cat", 50);
  weightMap.set("dog", 100);
  weightMap.set("bear", 300);
}

void draw() {
  int weight = weightMap.get("cat");
  circle(200, 200, weight);
}
```

Set up the dictionary

Query the dictionary



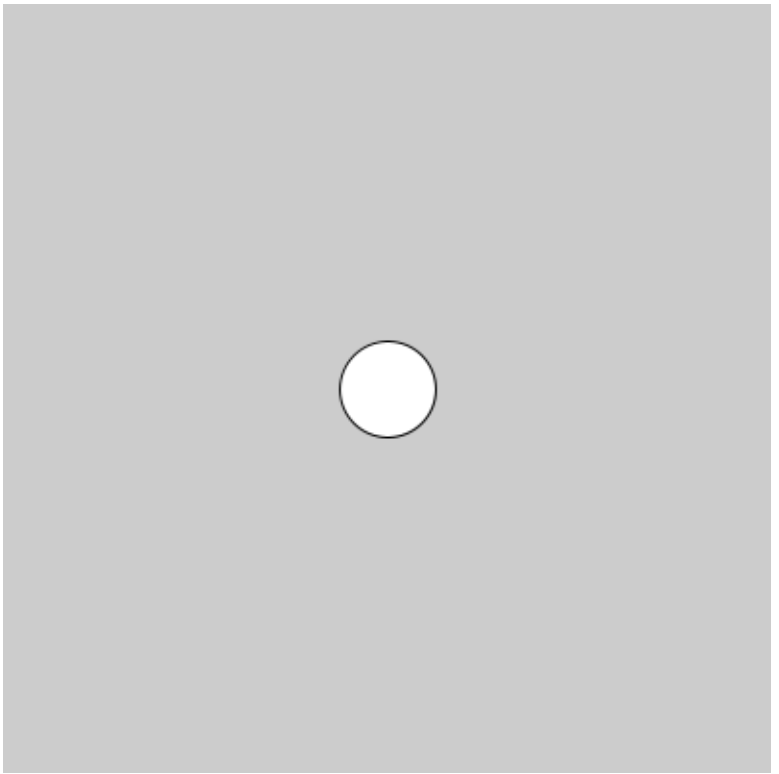
IntDict vs "string[] & int[]"

- Can we use **a string array** and **an integer array** to do something similar?

```
String[] animals = {"cat", "dog", "bear"};
int[] weights = {50, 100, 300};

void setup() {
  size(400, 400);
  noLoop();
}

void draw() {
  for (int i = 0; i < animals.length; i++) {
    if (animals[i] == "cat") {
      circle(200, 200, weights[i]);
    }
  }
}
```

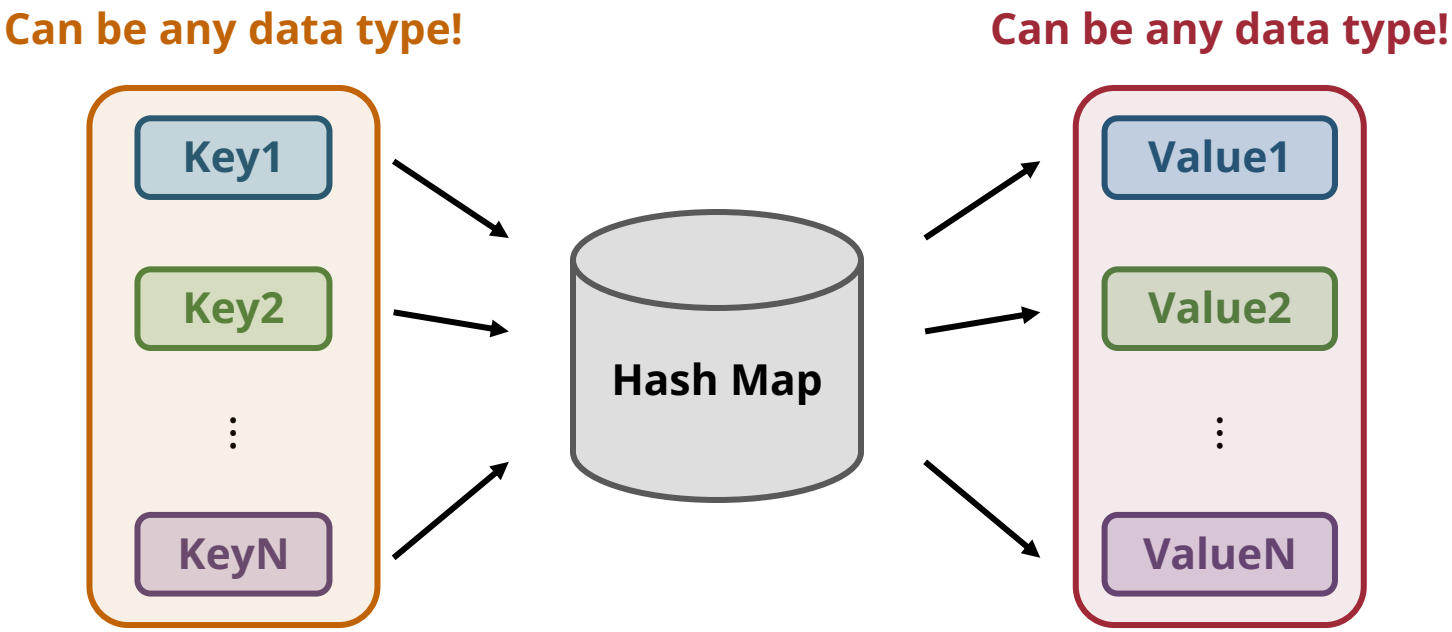
This is slow because we may need to go through the whole array to find the match!

Use a for loop to look up the "key"



Hash Map

- A **hash map** is the magic behind dictionaries
- **`HashMap`** Key-to-value mapping with desired key and value data types



A String-to-Array Hash Map

```
HashMap<String, int[]> m;
```

Declare a hash map that maps strings to integer arrays

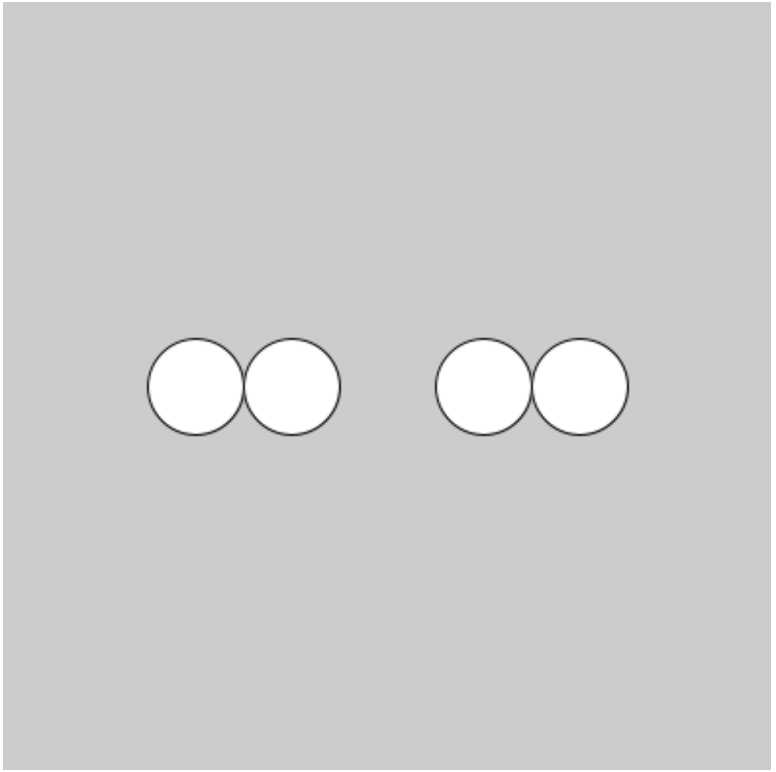
```
void setup() {
  size(400, 400);
  noLoop();

  m = new HashMap<String, int[]>();
  m.put("cat", new int[]{100, 150, 200});
  m.put("dog", new int[]{100, 300});
  m.put("bear", new int[]{100, 150, 250, 300});
}
```

Set up the dictionary

```
void draw() {
  int[] val = m.get("bear");
  for (int x: val) {
    circle(x, 200, 50);
  }
  println(m);
}
```

Query the hash map

```
{cat=[I@76cc6ab0, bear=[I@1d88e68, dog=[I@4e0d571f}
```



Recap

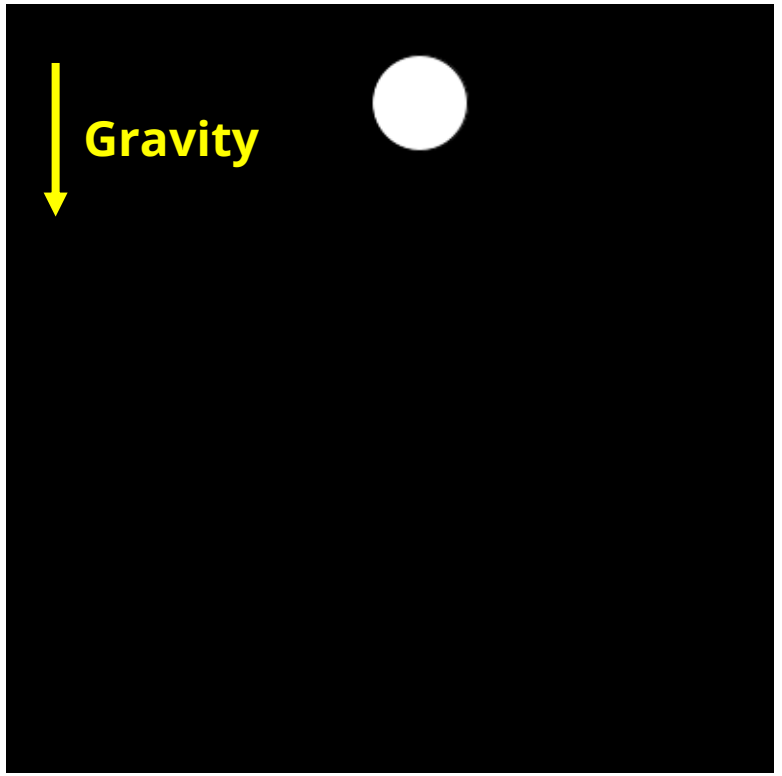
Example: Gravity

```
// Apply gravity to the ball
void applyGravity() {
  speedY += gravity;
}
```

Apply gravity as y-acceleration

```
// Check if the ball hit the walls
void checkWalls() {
  ...

  // Check if the ball hit the top and bottom walls
  if (y > height - radius) {
    speedY = -abs(speedY) * decay;
    y = height - radius;
  } else if (y < radius) {
    speedY = abs(speedY);
    y = radius;
  }
}
```

Reduce the speed a little bit when it hits the bottom wall



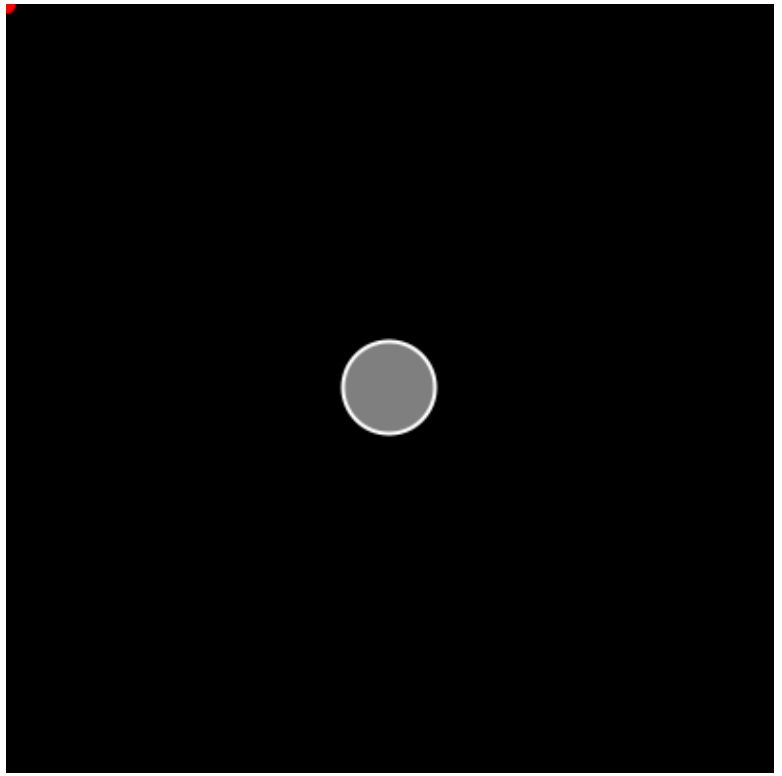
Example: Acceleration

```
class Mover {
  PVector location;
  PVector velocity;
  PVector acceleration;
  float topspeed = 5;

  ...

  void update() {
    PVector mouse = new PVector(mouseX, mouseY);
    PVector acceleration = PVector.sub(mouse, location);
    acceleration.setMag(0.2);

    velocity.add(acceleration);
    velocity.limit(topspeed);

    location.add(velocity);
  }
}
```

Calculate acceleration

Apply the acceleration

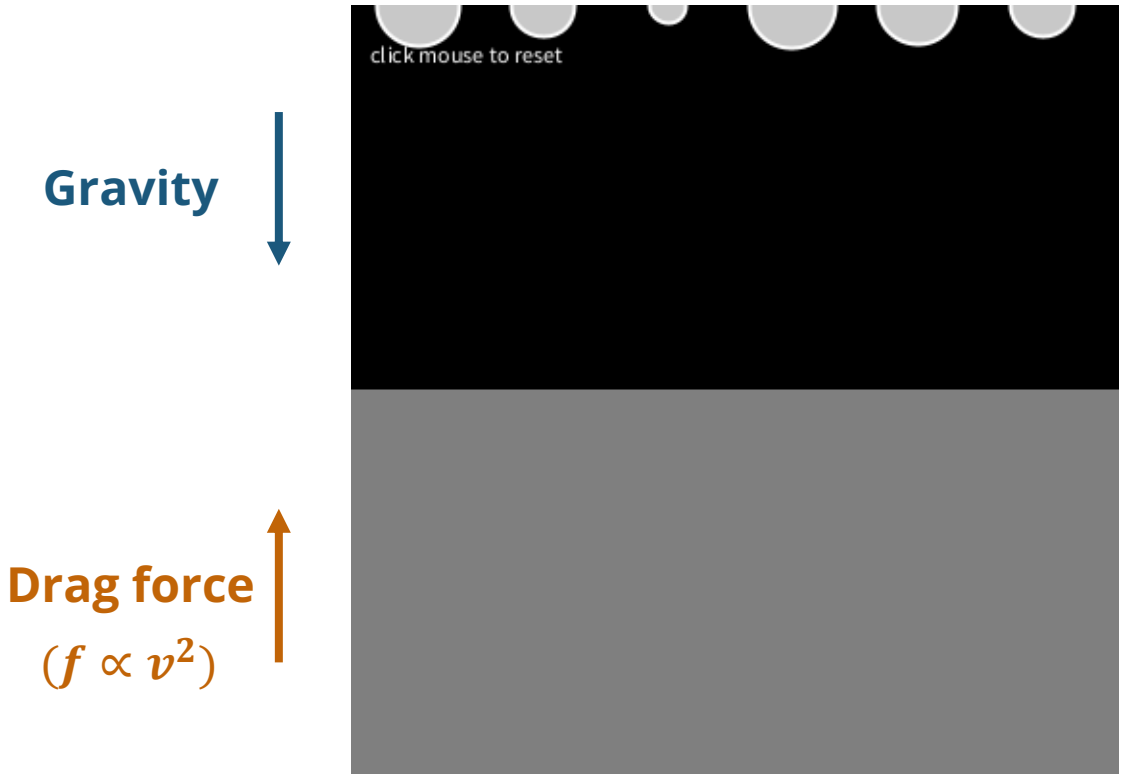
Move the ball



processing.org/examples/accelerationwithvectors.html

Example: Gravity & Fluid Resistance

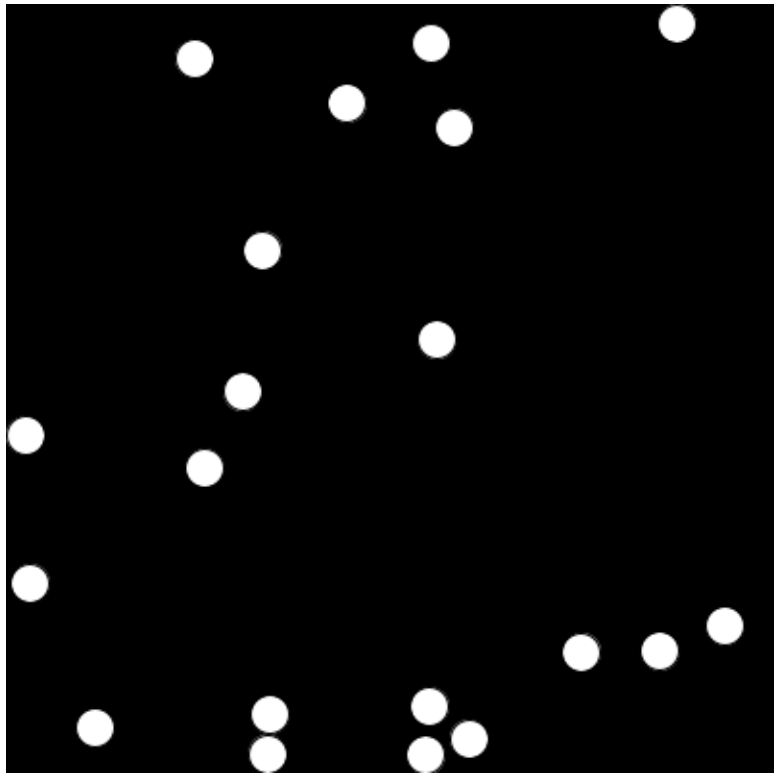
- Simulating multiple forces
  - Gravity
  - Fluid resistance (drag force)



processing.org/examples/forceswithvectors.html

Example: Bouncing Balls with Collision Detection

- What else do we need?
- How to check if the ball collides with another?
- What kind of function do we need?



```
Ball[] others;
```

An array to store other balls

```
void checkCollisions() {
  for (Ball other: others) {
    collide(other);
  }
}
```

Iterate over other balls to check if there's a collision

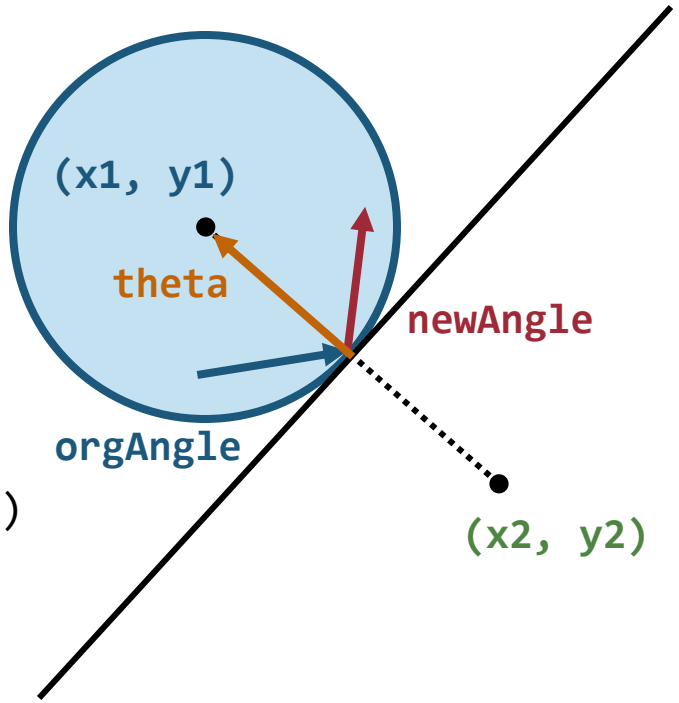
```
void collide(Ball other) {
  ???
}
```

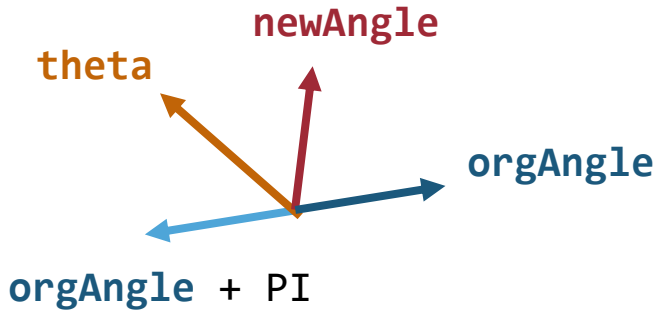
Check if there's a collision

Handling Collisions

```
theta = atan2(y2 - y1, x2 - x1)
```

```
orgAngle = atan2(speedY, speedX)
```






```
newAngle = (orgAngle + PI) – 2 (orgAngle + PI – theta)
         = 2 theta - PI - orgAngle
```

Example: Bouncing Balls with Collision Detection

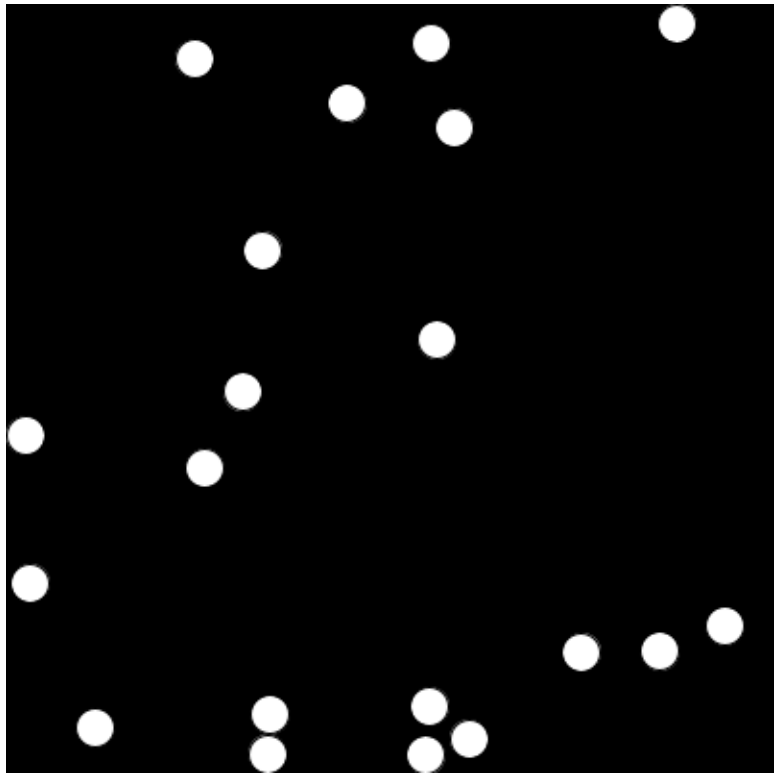
```
void collide(Ball other) {
  if (other == this) return;
  float dist = dist(x, y, other.x, other.y);
  if (dist >= size) return;
  x -= speedX;
  y -= speedY;
  float theta = atan2(other.y - y, other.x - x);
  float orgAngle = atan2(speedY, speedX);
  float newAngle = (theta - PI + theta - orgAngle);
  speedX = speed * cos(newAngle);
  speedY = speed * sin(newAngle);
}
```

Do nothing if it's the same ball

Do nothing if they do not collide

Revert the ball back to where it was before the collision

Find the velocity after the collision



2D Elastic Collisions

Conservation of momentum

$$m_1\vec{v}_1 + m_2\vec{v}_2 = m_1\vec{v}_1' + m_2\vec{v}_2'$$

Conservation of kinetic energy

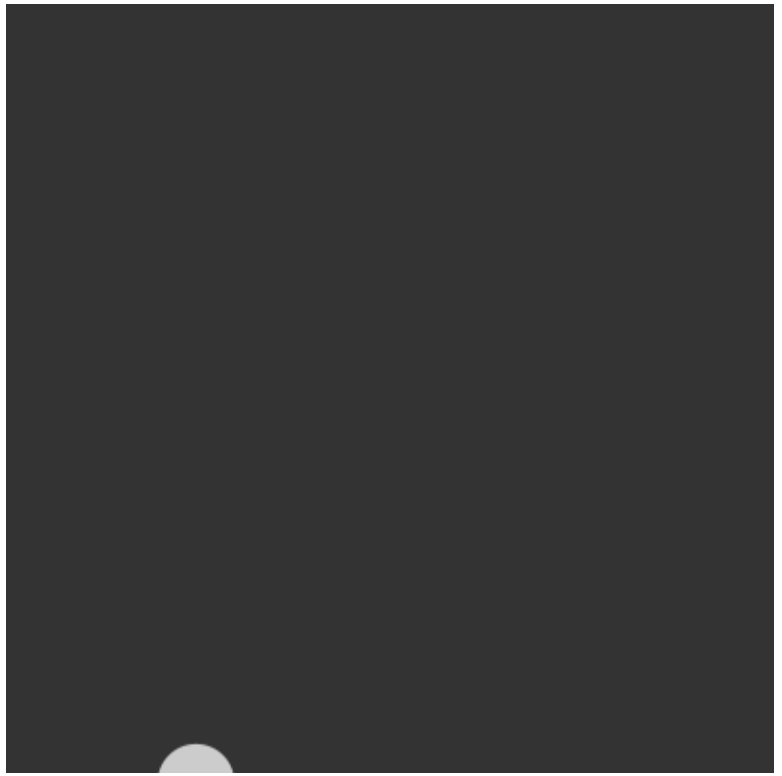
$$m_1\vec{v}_1 + m_2\vec{v}_2 = \frac{1}{2}m_1\vec{v}_1'^{\,2} + \frac{1}{2}m_2\vec{v}_2'^{\,2}$$

Conservation of angular momentum

(Source: Simon Steinmann, via Wikimedia Commons)

Example: Collision with Weights

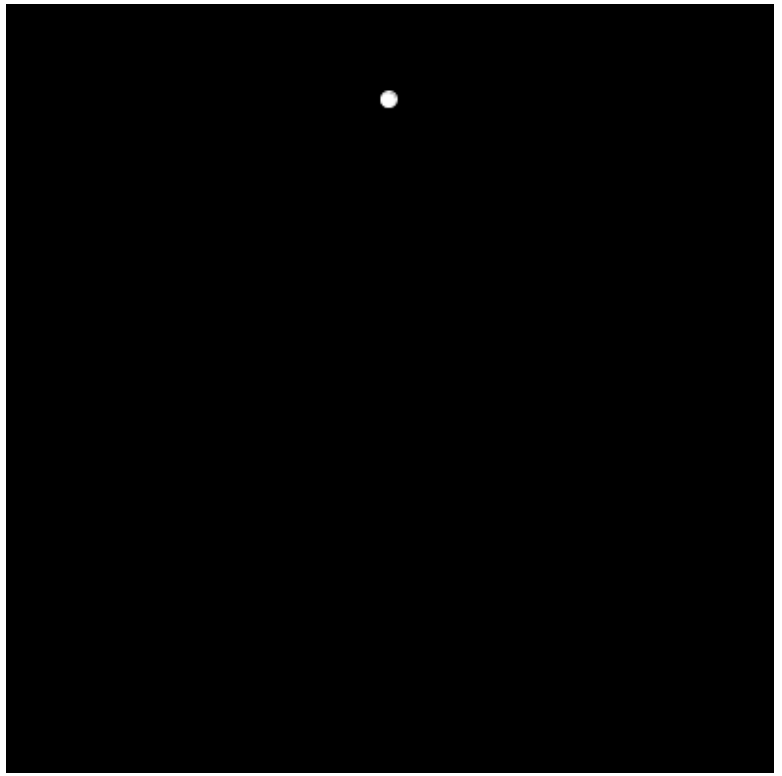
- Different **weights** and different **speeds**
- Solve the equations to find the new velocities
  - Conservation of momentum
  - Conservation of kinetic energy
  - Conservation of angular momentum



processing.org/examples/circlecollision.html

Example: Simple Particle System

- A `Particle` object
  - Falls with gravity
  - Fades out over time (die after a predefined lifespan)
- A `ParticleSystem` object is an ArrayList of `Particle` objects (i.e., `ArrayList<Particle>`)
  - Allows storing a variable number of particles
- Show all the `Particle` objects at each call of the `draw()` function



Example: Fireworks

- A `Firework` object
  - Starts as one single `Particle` object
    - Initialized with random force up
    - Flies up with gravity slowing it down
    - Explodes when speed reaches zero
  - Becomes many `Particle` objects after explosion
    - Initialized with random forces towards random directions
    - Fall with gravity
    - Die after invisible on the canvas

Next Lecture

Midterm Assignment Showcase