Review

- Visualization
  - Use arrays of data
  - Or sample a mathematical function
    - directly
    - or by making an array of discrete values
  - Plot data
    - Plot axes and supporting information.
    - Rects (for bar plot)
    - points, lines, and/or vertices for Line plots

Variable Grouping

- Concept 1 (sequence)
  - Array
    - a fixed size
    - one type of value
    - declare an array
      - float[] temps;
      - String[] names;
    - instantiate an array
      - intervals = new int[10];
      - temps = {1.0,32.0,212.0};
      - names = new String[5];
    - assign values to elements of an array
      - intervals[0] = 10;
      - names[3] = "Beyonce";
      - temps[2] = -300.0;

- Concept 2 (set of values with cohesive meaning)
  - Class: Object
    - PVector
      - class PVector{
        float x;
        float y;
        float z;
      }
    - Declare a PVector
      - PVector position;
    - Instantiate PVector
      - position = new PVector(10,10);
    - Assign values to elements of PVector
      - position.x = 100;
      - position.y = 10;

Beyond Grouping: Methods

- Methods provide the ability to access and change the values of an Object
  - PVector
    - set – set the values using floats, a PVector, or a float[]
    - add(),sub() – adds/subtracts values using floats or a PVector
    - mult(),div() – multiplies/divides vector by a scalar
    - dist() – returns the Euclidean distance between 2 points
    - angleBetween() – returns the angle between 2 points
    - ...

What is an Object?

- An object is an instance of a class.
- What is an instance?
  - An instance is a distinct example of the class that
    - is in memory
    - has specific assignments for the variables declared by the class it represents.
    - has functionality based on the class.
- What is a class?
  - A complex data type.
  - The design for objects of its type.
Object Oriented Programming

- Objects group related variables and functions.
  - Object variables are called fields.
  - Object functions are called methods.
- An object has to be designed first and it has a custom type.
- Objects can be created, named and referenced with variables.
  - Very similar to standard data types.

Creating New Objects with Classes

- To create a new instance of an object, use the `new` keyword and call the object Constructor.

```java
MyObjectName ob = new MyObjectName();
Point p1 = new Point();
Point p2 = new Point();
```
- To access fields and methods, use the dot notation.

```java
p1.display();
pintle(p2.x);
```

Class/Object

- **Keyword class**
  - declares a new type.
- **Data fields (class variables)**
- **Constructor**
- **Methods (class functions)**
  - update
  - move
  - display/draw

```java
class Point {
    // Fields
    int x;
    int y;
    Color c;
    // Constructor
    Point() {
        x = 0;
        y = 0;
        c = Color(255, 255, 255);
    }
    // Methods
    void update() {
    }
    void display() {
        noStroke();
        fill(c);
        ellipse(x, y, 10, 10);
    }
}
```

Creating New Objects with Classes

- To create a new instance of an object, use the `new` keyword and call the object Constructor.

```java
MyObjectName ob = new MyObjectName();
Point p1 = new Point();
Point p2 = new Point();
```
- To access fields and methods, use the dot notation.

```java
p1.display();
pintle(p2.x);
```

Creating New Objects with Classes

- To create a new instance of an object, use the `new` keyword and call the object Constructor.

```java
MyObjectName ob = new MyObjectName();
Point p1 = new Point();
Point p2 = new Point();
```
- To access fields and methods, use the dot notation.

```java
p1.display();
pintle(p2.x);
```

Class vs. Object

- **Class**
  - A special function that always carries the same name as the class itself.
  - Called automatically at the creation/instantiation of an object.
  - Used to initialize all of the objects variables.

**Example**

- eyes

The Constructor

- A special function that always carries the same name as the class itself.
- Called automatically at the creation/instantiation of an object.
- Used to initialize all of the objects variables.
Defining Your Own Objects with Classes

```java
// Defining a new class of object
class MyObjectName {
    // All field variable declarations go here;
    // Define a special function-like statement called
    // the class's Constructor.
    // It's name is same as object class name,
    // with no return value.
    MyObjectName( optional arguments ) { 
        // Perform all initialization here
    }
    // Declare all method functions here.
}
```

Example

- Ball Object

```
// A Ball Class
class Ball {
    // Fields
    int w; // width
    int h; // height
    float x; // x position
    float y; // y position
    float spdX; // x velocity
    float spdY; // y velocity
    float gravity = .03;

    // Constructor
    Ball() {
        w = h = 20;
        x = random(0, width/2); y = random(10, 20);
        spdX = random(0.5, 1.3); spdY = 0;
    }

    // Methods
    void update() {
        x += spdX;
        y += spdY;
        // Bounce off walls and floor
        if (x + w/2 > width || x – w/2 < 0)
            spdX = -spdX;
        if (y + h/2 > height || y – h/2 < 0)
            spdY = -spdY;
    }
    void display() {
        ellipse( x, y, w, h);
    }
}
```

Constructors can take arguments.

- More than one constructor can be written for a class.
- As long as they are differentiable in the number/type of parameters they take.
- There is a default constructor even if you don’t write one
  - it doesn’t do much though.
  - all basic data types are initialized to their default value
    (usually 0 or false), color is a basic data type in Processing
  - all Reference data types are initialized to null;

```
// Ball object
Ball b1 = new Ball(0, 0);
Ball b2 = new Ball(20, 20);
```
Examples

- ballDropObj2
- ballDropObjArray
- ballDropObjArray2