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.
Defining Your Own Object with Classes

- Classes are blueprints or prototypes for new objects
- Classes define all field and method declarations
  ... which are repeated for each new object created
- Classes DO NOT set the data values stored in fields
  ... but they likely determine how
- Using a class to create a new object is called instantiating an object
  ... creating a new object instance of the class
- Classes often model real-world items
Class vs. Object

Object Oriented Programming

- Objects are **software bundles** that wrap up all semantically related variables and functions.
  - Object variables are called **fields**
  - Object functions are called **methods**

- Objects can be **created**, **named** and **referenced** with variables
  - Very similar to standard data types

- An object's individual fields and methods are accessed using syntax called **dot-notation**
Class/Object

- Keyword class
- 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();
```

```java
Point p1 = new Point();
Point p2 = new Point();
```
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

// 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.
}

// Defining Your Own Objects with Classes

// 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.
}
// A Ball Class

class Ball {
  // Fields
  int w; int h; // width and height of ball
  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;
    spdY += gravity;
    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);
  }
}
Defining Your Own Object with Classes

- Classes are blueprints or prototypes for new objects.
- Classes define all field and method declarations...
  ... which are repeated for each new object created.
- Classes DO NOT set the data values stored in fields...
  ... but they likely determine how.
- Using a class to create a new object is called *instantiating* an object...
  ... creating a new object *instance* of the class.
- Classes often model real-world items.
Constructor overloading

- 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;
Examples