objects and classes

DAY 8
classes are templates.
objects are specific instances of a class.

These are Objects Of **furniture** class
1. Chair
2. Table
3. Bed
4. Etc.
classes - factories with instructions on how to construct an object when invoked by program.

build me a green Audi Car

Car Factory

instantiate
objects - a specific member of the class; made of variables, functions (methods), constructor.
objects are specific instances of a class. methods are actions an object can perform.
Plan

• 1. specify the class - methods - variables,
• 2. call the class to instantiate objects.

• But first, group exercise!
Exercise

What are some examples of classes, objects, and methods in everyday (or not so everyday) life?

Methods (functions)?

StartEngine();
StopEngine();
Open(int door);
Close(int door);
Turn(int direct);
etc...
Bouncing Ball: object oriented version

```java
// Name of class
class BouncingBall {

  // Variables

  // The constructor

  // Functions

// Main BouncingBall Program
void setup() {
  size (600, 600);
  smooth();
}

void draw () {
  background (0);
}
```
myBall, an instance of class BouncingBall

```java
class BouncingBall {
    // VARIABLES (Note: these variables are global to the class, you are just declaring them here)
    float x = 0;
    float y = 0;

    // THE CONSTRUCTOR (Note: the constructor runs only once when called it is not a looping function)
    BouncingBall() {
    }

    // FUNCTIONS
    void display () {
        ellipse (200,200,20,20);
    }

    // Initializes
    void setup () {
        size (660,600);
        smooth();
        myBall = new BouncingBall();  // The keyword "new" is saying build a new instance of the class.
    }

    // Functionality
    void draw () {
        background (0);
        myBall.display();  // This is a function of the class, the dot syntax opens up the class and its functions. In this case we have one function, display()
    }
}
```
```java
public class BouncingBall {
    // GLOBAL VARIABLES (Specific only to the BouncingBall class)
    float x = 0;
    float y = 0;

    // THE CONSTRUCTOR (float _x, float _y)
    BouncingBall(float _x, float _y) { // here, we are adding the variables x and y inside the constructor.
        // x and y are only visible to the constructor. We need x and y as placeholders so we can manipulate our objects in the main program.
        x = _x; // we also want our objects to access the functionality below. To do this
        y = _y; // we set our x and y placeholders to equal our class's global variables x and y.
    }

    // FUNCTIONS
    void display() { // We replace the first two values in the ellipse with x and y so we can have
        ellipse(x, y, 20, 20); // choices for its functionality,
    }

    void move() { // move the ball around the screen
        x = x + speedX;
        y = y + speedY;
    }

    void bounce() { // bounce the ball when you get to the edges.
        if ((x > width) || (x < 0)) {
            speedX = speedX * -1;
        }
        if ((y > height) || (y < 0)) {
            speedY = speedY * -1;
        }
    }

    void gravity() { // mimic gravity by changing speed on y axis.
        speedY = speedY + 0.2;
    }

    void display() { // display the ball
        ellipse(x, y, 20, 20);
    }
}
```

//Main BouncingBall Program

//Declared
BouncingBall myBall;

//Initializes
void setup() {
  size (600,600);
  smooth();
  myBall = new BouncingBall (400,400);
}

//Functionality
void draw () {
  background (0);
  myBall.display();
  myBall.move();
  myBall.bounce();
  myBall.gravity();
}

class BouncingBall {

  //GLOBAL VARIABLES
  float x = 0;
  float y = 0;
  float speedX = 4;
  float speedY = .5;

  //THE CONSTRUCTOR
  BouncingBall(float _x, float _y) {
    x = _x;
    y = _y;
  }

  //FUNCTIONS

  void run() { // New comprehensive function called run that lumps them all together in one move.
    move();
    bounce();
    gravity();
    display();
  }

  void move() {
    x = x + speedX;
    y = y + speedY;
  }

  void bounce() {
    if ((_x > width) || (_x < 0)) {
      speedX = speedX * -1;
    }
    if ((_y > height) || (_y < 0)) {
      speedY = speedY * -1;
    }
  }

  void gravity() {
    speedY = speedY + 0.2;
  }

  void display () { //display the ball
    ellipse (x,y,20,20);
  }
}
//Main BouncingBall Program

//Declared
BouncingBall myBall;

//Initializes
void setup() {
  size (600,600);
  smooth();
  myBall = new BouncingBall (400,400);
}

//Functionality
void draw () {
  background (0);
  myBall.run(); //Just one function needed here now, run()
Exercises (for fun)

• Make an array of BouncingBalls and put them at regular spacing around the screen.

• Show the balls’ movements (trails) as they are influenced by the physics.
objects and classes

DAY 8