Basic operations

I’m sorry,

You’re basic.
Banana Nut Bread

Directions
1. Preheat the oven to 350°F (175°C).
2. Mix butter into the mashed bananas in a large mixing bowl.
3. Mix in the sugar, egg, and vanilla.
4. Sprinkle the baking soda and salt over the mixture and mix in.
5. Add the flour and nuts last, mix.
6. Pour mixture into a buttered 4x8 inch loaf pan.
Banana Nut Bread

Java Directions
1. `oven.preheat(350);`
2. `bowl.mix(butter, bananas);`
3. `bowl.mix(sugar, egg, vanilla);`
4. `bowl.sprinkle(baking soda, salt);`
5. `bowl.mix(flour, nuts);`
6. `bowl.pour(pan);`
7. `pan.bake(60);`
8. `pan.cool();`
In Java, methods (verbs) must always have parenthesis

Even if there are no arguments:

sing()

This is the way Java differentiates between methods and objects
A method you will use a lot is: `System.out.println()`

You can give this method a single argument in quotes and it will show this on the monitor:

```
System.out.println(“Hi!”);
```

... will show: Hi! on the screen

(See: HelloWorld.java)
NetBeans

Create a new project every time you want to make a new piece of code

Combined editor and compiler

It is important that you name the file the same as the class inside it.
The main() method is special:

```java
public static void main(String[] args) {
```

(Ignore the blue words before main for now)

main() is where the program will always start, so it always has step 1 of the instructions

You cannot have multiple main() methods in the same project
There are a couple different ways to display things to the screen:

```
System.out.println()
System.out.print()
System.out.printf()
```

You can take your pick (printf() is more advanced)
Output

You can force the computer to go down a line either by using:

1. `System.out.println()` with no arguments
2. `\n`

This means these two of code are identical:

```
System.out.println("Hello world!");
System.out.print("Hello world!\n");
```

Note: anything with a `\` in quotes will not be output verbatim
Magic 8 ball
Magic 8 ball

What a rip off!
package magic8ball;

public class magic8ball {
    public static void main(String[] args) {
        System.out.println("Maybe.");
    }
}
package quadratic;

class quadratic {

    public static void main(String[] args) {
        double a, b, c;
        double ans1, ans2;

        a = 2;
        b = 5;
        c = 2;

        ans1 = (-b+Math.sqrt(b*b-4*a*c)) / (2*a);
        ans2 = (-b-Math.sqrt(b*b-4*a*c)) / (2*a);

        System.out.println("Answer 1 = "+ans1);
        System.out.println("Answer 2 = "+ans2);
    }
}
Comments

Comments are ignored pieces of code

// denotes a single line that is commented
// (everything before hitting enter)

/* denotes the beginning of a comment
and the end of a comment is denoted by */
“White space” is ignored
(this is why we need a semi-colon)

Braces denote a block of code  
(belonging to a method, class, etc.)

Talk about semicolons later... for now put a
semicolon after every “step” in the directions

Ctrl+Space is your friend in NetBeans
Types of errors

Syntax error - code will not compile
  e.g.  `System.out.println("hi");`

Runtime error - code crashes after starting
  e.g.  `(char input in RuntimeError.java)`

Logic error - code runs but doesn't return
  the correct answer
  (See: LogicError.java)