Variables & Data Types

Brandon Krakowsky

Variables & Data Types

Intro to Variables

- A variable is a *symbolic* name for (or reference to) a value
  - You can use variables to store all kinds of data!

Intro to Variables

- A variable is a *symbolic* name for (or reference to) a value
  - You can use variables to store all kinds of data!
- Variable names are case sensitive
  - You typically name variables using “camelCase”, starting with a lowercase letter

Intro to Variables

- A variable is a *symbolic* name for (or reference to) a value
  - You can use variables to store all kinds of data!
- Variable names are case sensitive
  - You typically name variables using “camelCase”, starting with a lowercase letter
- Every variable in Java has a pre-defined *type*
  - You declare the *type* in front of the variable

```
int myInt = 0; //myInt can only store an int
```

Intro to Variables

- A variable is a *symbolic* name for (or reference to) a value
  - You can use variables to store all kinds of data!
- Variable names are case sensitive
  - You typically name variables using “camelCase”, starting with a lowercase letter
- Every variable in Java has a pre-defined *type*
  - You declare the *type* in front of the variable
    ```
    int myInt = 0; //myInt can only store an int
    ```
- You MUST store that kind of data in the variable
  - For example, you can’t do this:
    ```
    int myInt = “hello”; //”hello” is a String, not an int!
    ```
  - Eclipse won’t even let you compile your code!

Intro to Variables

- A variable is a *symbolic* name for (or reference to) a value
  - You can use variables to store all kinds of data!
- Variable names are case sensitive
  - You typically name variables using “camelCase”, starting with a lowercase letter
- Every variable in Java has a pre-defined *type*
  - You declare the *type* in front of the variable

```
int myInt = 0; //myInt can only store an int
```

- You MUST store that kind of data in the variable
  - For example, you can’t do this:

```
int myInt = “hello”; //”hello” is a String, not an int!
```

  - Eclipse won’t even let you compile your code!
- The *type* of a variable CANNOT be changed
  - Java is *statically* typed
  - For comparison, in a language like Python, you can change variable types on the fly, because it’s *dynamically* typed

Data Types

- Some primitive (simple) data types
  - int: Integer
  - float: Floating point (decimal)
  - boolean: true/false

Data Types

- Some primitive (simple) data types
  - int: Integer
  - float: Floating point (decimal)
  - boolean: true/false
- Some other primitive types
  - char: Single character
  - double: Large and precise floating point
  - byte, short, or long: Various integer sizes (8, 16, 64 bits)

Data Types

- Some primitive (simple) data types
  - int: Integer
  - float: Floating point (decimal)
  - boolean: true/false
- Some other primitive types
  - char: Single character
  - double: Large and precise floating point
  - byte, short, or long: Various integer sizes (8, 16, 64 bits)
- Another type is String, which is an Object (not a primitive)
  - It’s used to store a *character string*

Data Types

- Some primitive (simple) data types
  - int: Integer
  - float: Floating point (decimal)
  - boolean: true/false
- Some other primitive types
  - char: Single character
  - double: Large and precise floating point
  - byte, short, or long: Various integer sizes (8, 16, 64 bits)
- Another type is String, which is an Object (not a primitive)
  - It's used to store a *character string*
- You might also come across Integer, Boolean, Double, etc.
  - Don't worry about these for now!

Declaring Variables

- You can declare variables WITH initial values

```
//Declares a variable to store an int
int count = 0;

//Declares a variable to store a String
String firstName = “Brandon”; //Use double quotes to define a String
```

Declaring Variables

- Or declare variables WITHOUT initial values

```
//Declares a double without actually creating a double
double distance;

//Declares a String without actually creating a String
String color;
```

Declaring Variables

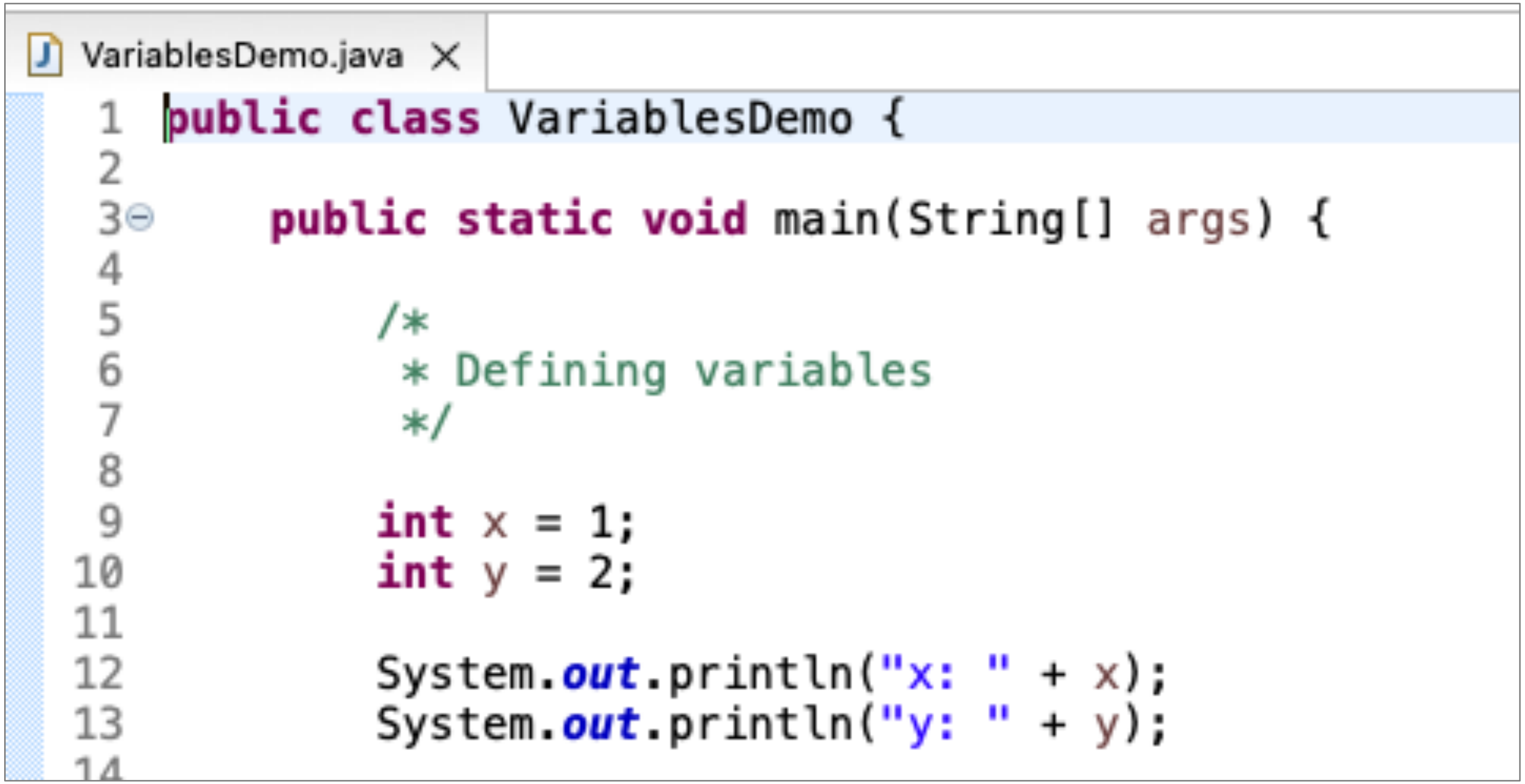
- Or declare variables WITHOUT initial values

```
//Declares a double without actually creating a double
double distance;

//Declares a String without actually creating a String
String color;
```

- And obviously set the variables later

```
//Puts a double in the distance variable
distance = 2.3;

//Puts a String in the color variable
color = “red”;
```

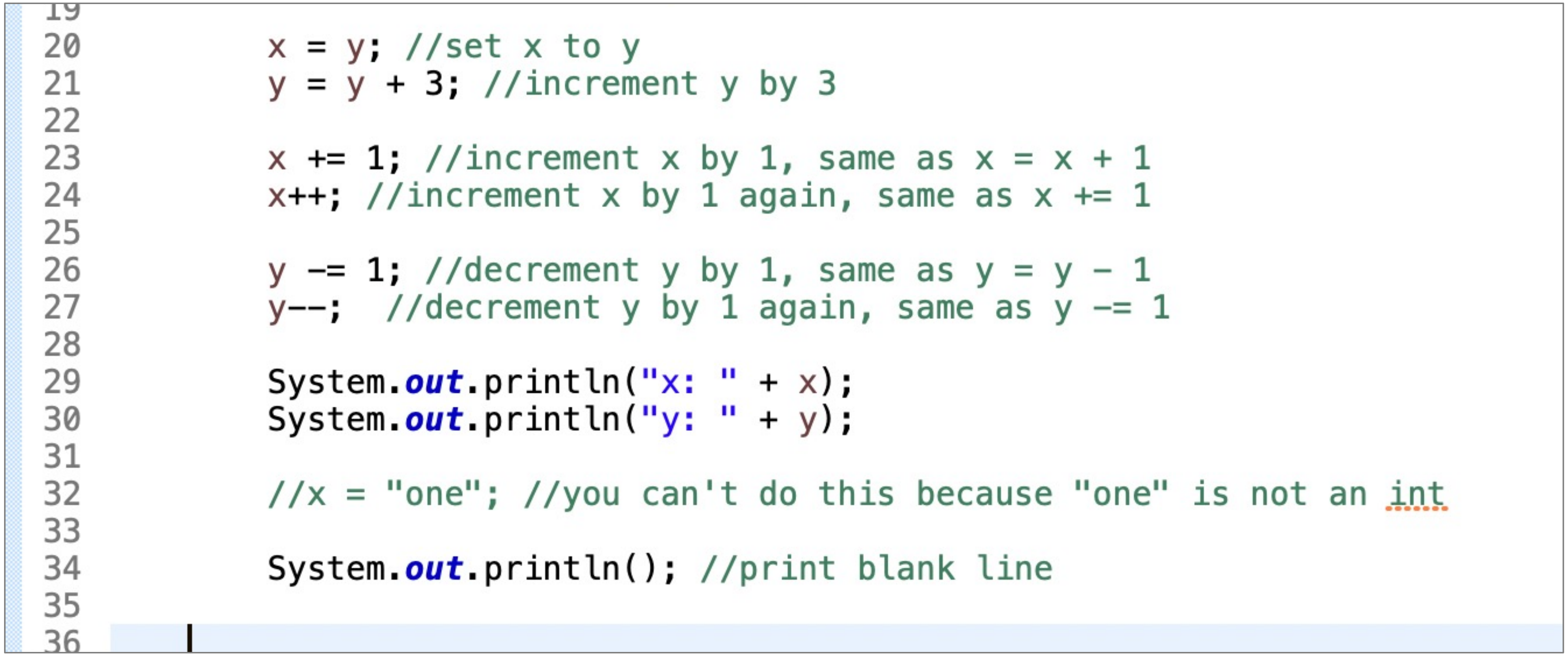
Variables



VariablesDemo.java

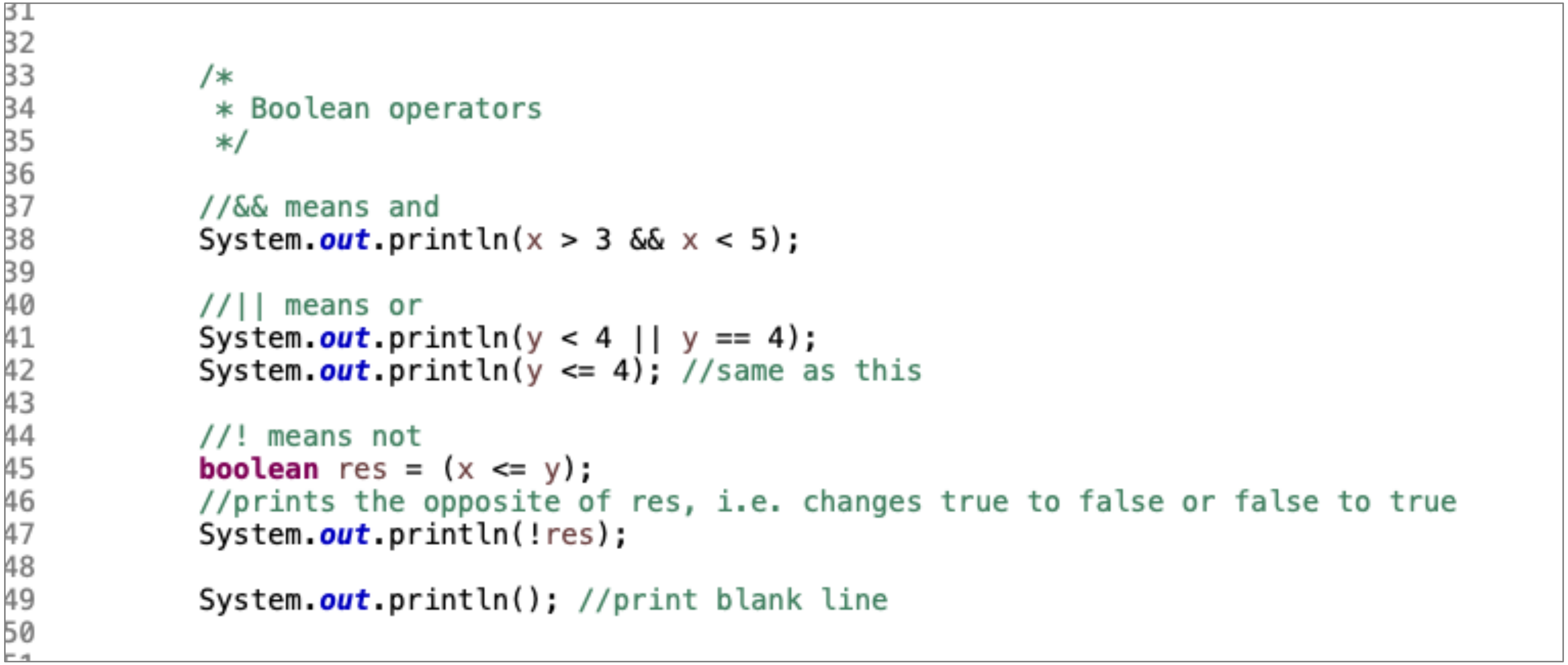
```
public class VariablesDemo {

    public static void main(String[] args) {

        /*
         * Defining variables
         */

        int x = 1;
        int y = 2;

        System.out.println("x: " + x);
        System.out.println("y: " + y);
```

Variables



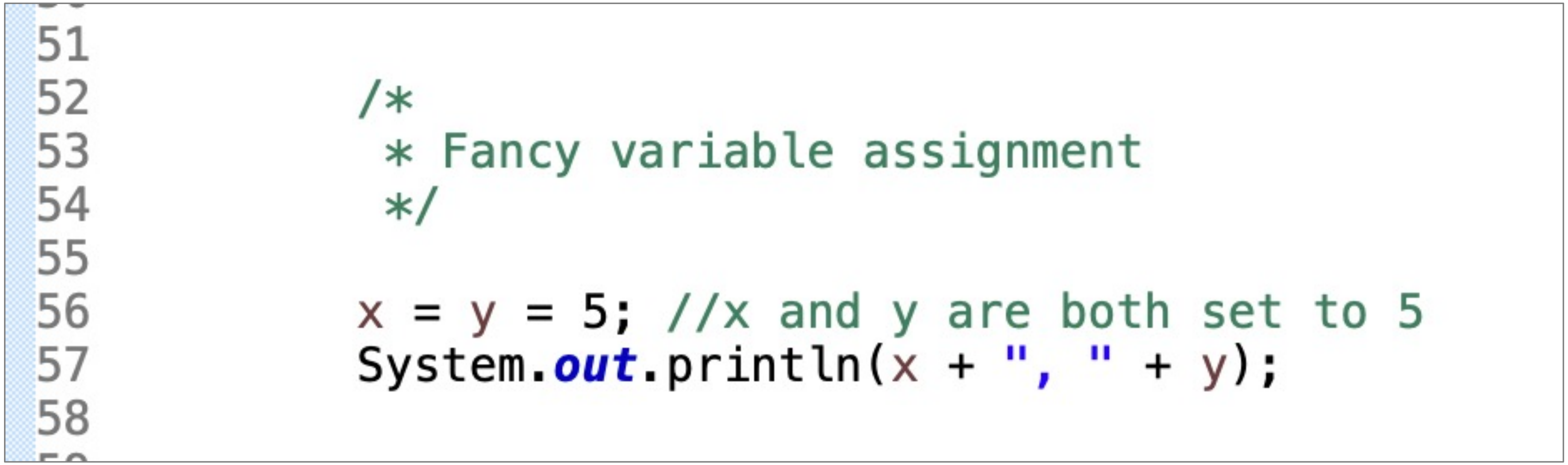
```
        x = y; //set x to y
        y = y + 3; //increment y by 3

        x += 1; //increment x by 1, same as x = x + 1
        x++; //increment x by 1 again, same as x += 1

        y -= 1; //decrement y by 1, same as y = y - 1
        y--;  //decrement y by 1 again, same as y -= 1

        System.out.println("x: " + x);
        System.out.println("y: " + y);

        //x = "one"; //you can't do this because "one" is not an int

        System.out.println(); //print blank line

```

Variables – Boolean Operators



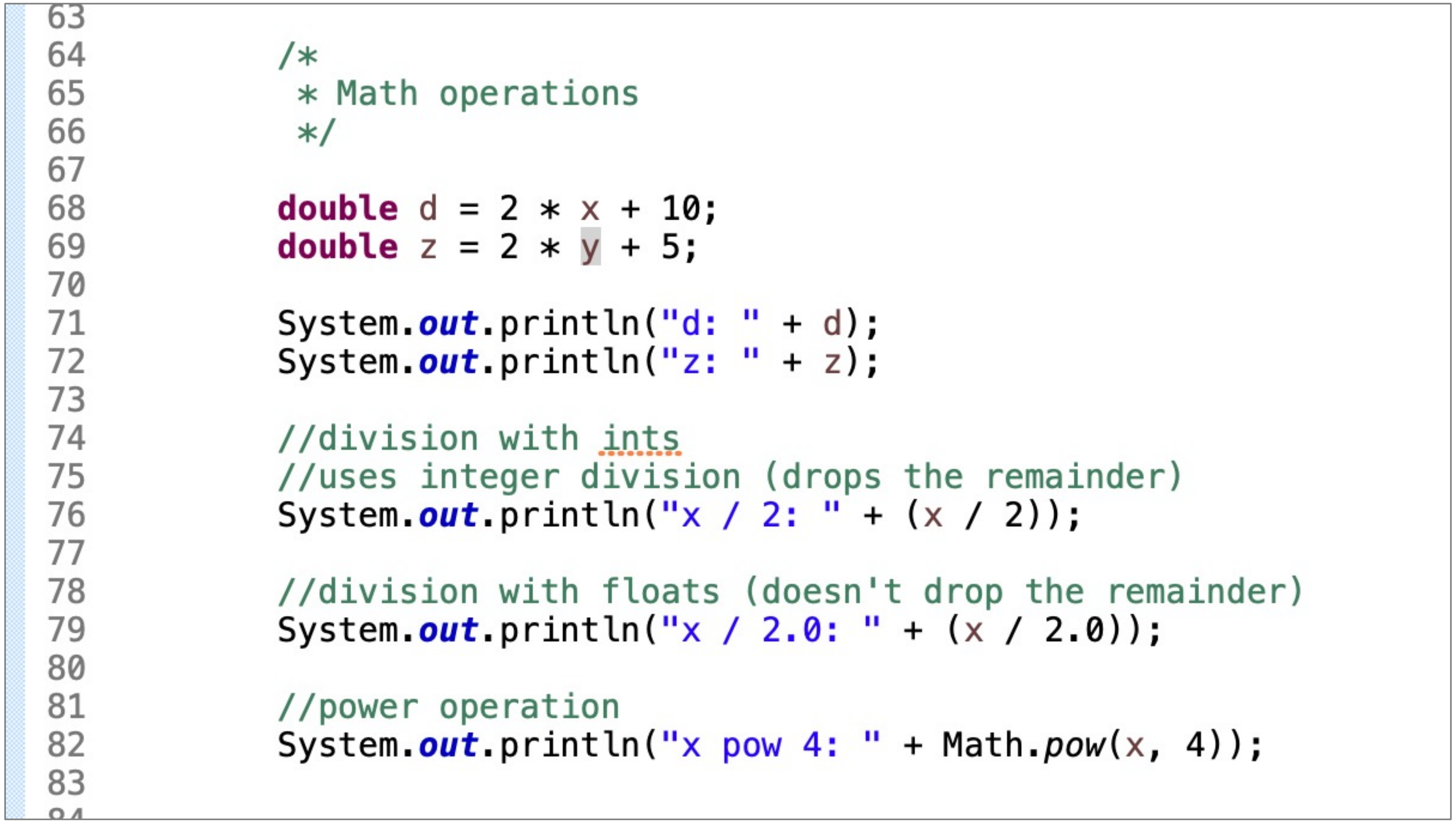
```
        /*
         * Boolean operators
         */

        //&& means and
        System.out.println(x > 3 && x < 5);

        //|| means or
        System.out.println(y < 4 || y == 4);
        System.out.println(y <= 4); //same as this

        //! means not
        boolean res = (x <= y);
        //prints the opposite of res, i.e. changes true to false or false to true
        System.out.println(!res);

        System.out.println(); //print blank line
```

Variables – Fancy Variable Assignment



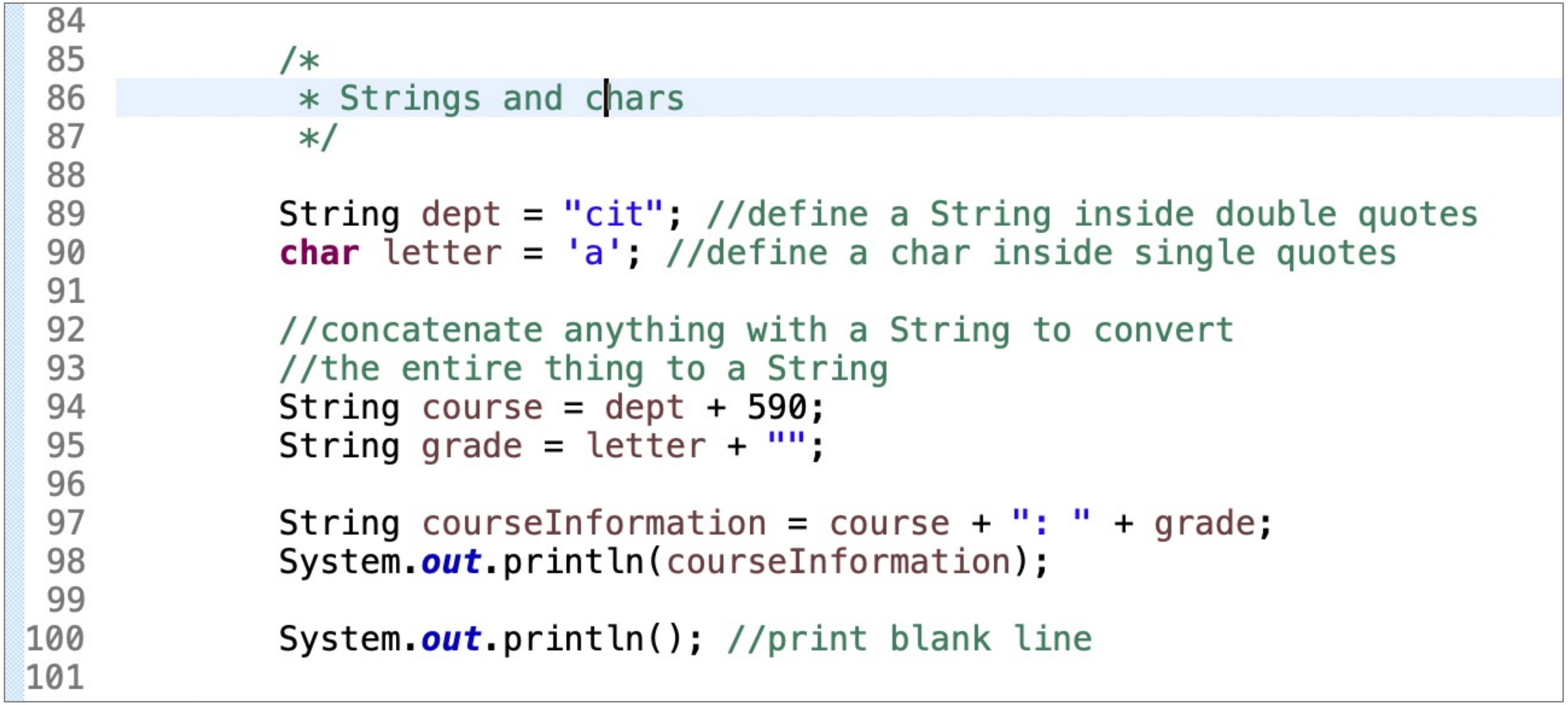
```
        /*
         * Fancy variable assignment
         */

        x = y = 5; //x and y are both set to 5
        System.out.println(x + ", " + y);

```

Variables – Math Operations



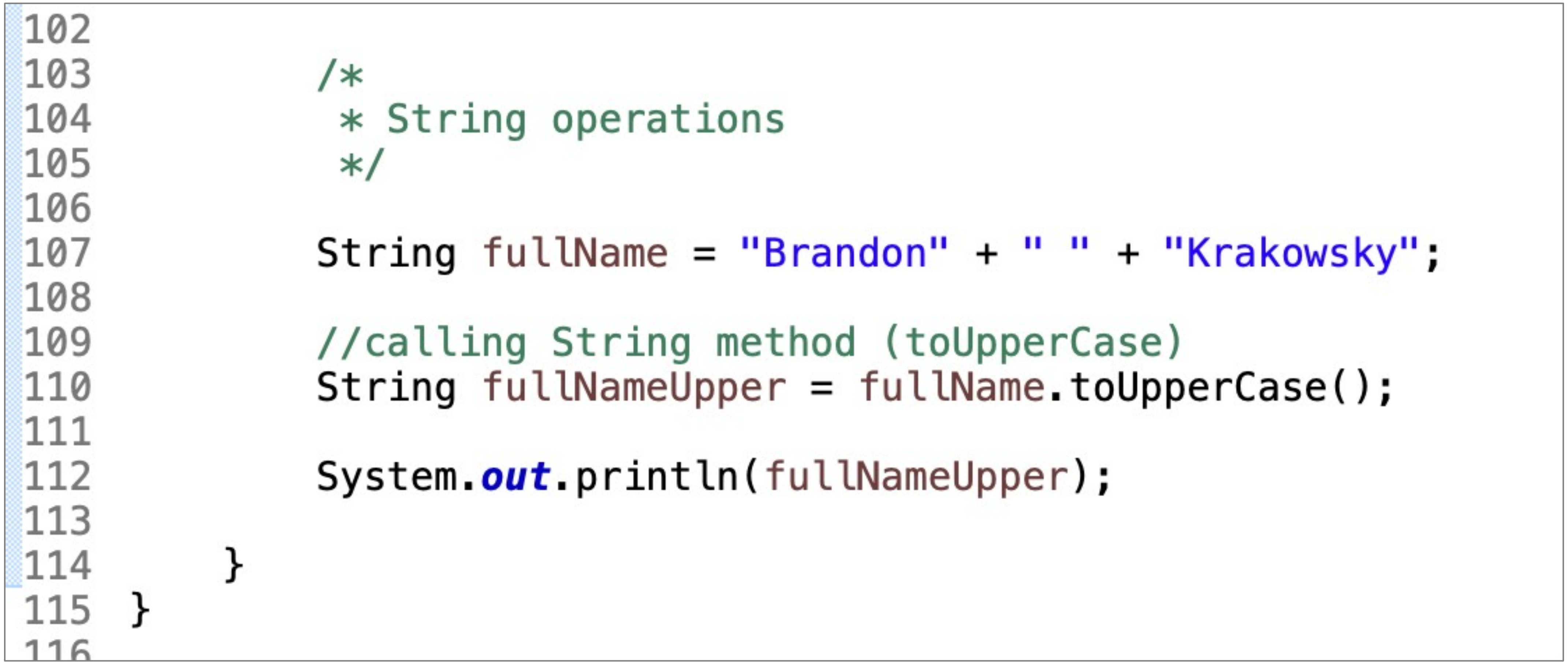
```java
/*
 * Math operations
 */

double d = 2 * x + 10;
double z = 2 * y + 5;

System.out.println("d: " + d);
System.out.println("z: " + z);

//division with ints
//uses integer division (drops the remainder)
System.out.println("x / 2: " + (x / 2));

//division with floats (doesn't drop the remainder)
System.out.println("x / 2.0: " + (x / 2.0));

//power operation
System.out.println("x pow 4: " + Math.pow(x, 4));

```

Variables – Strings & chars



```
        /*
         * Strings and chars
         */

        String dept = "cit"; //define a String inside double quotes
        char letter = 'a'; //define a char inside single quotes

        //concatenate anything with a String to convert
        //the entire thing to a String
        String course = dept + 590;
        String grade = letter + "";

        String courseInformation = course + ": " + grade;
        System.out.println(courseInformation);

        System.out.println(); //print blank line
```

Variables – String Operations



```java

        /*
         * String operations
         */

        String fullName = "Brandon" + " " + "Krakowsky";

        //calling String method (toUpperCase)
        String fullNameUpper = fullName.toUpperCase();

        System.out.println(fullNameUpper);

    }
}
```

Javadocs

- You can (and should) provide *Javadocs* (*Java documentation*) just *before* the definition of a class (or method)
    - *Javadocs* describe the operation of the class (or method)

Javadocs

- You can (and should) provide *Javadocs* (*Java documentation*) just *before* the definition of a class (or method)
  - *Javadocs* describe the operation of the class (or method)
- *Javadocs* are for someone who is using your class (or method) and wants to know “what it does” at a high level and/or “how to use it”

Javadocs

- You can (and should) provide *Javadocs* (*Java documentation*) just *before* the definition of a class (or method)
    - *Javadocs* describe the operation of the class (or method)
- *Javadocs* are for someone who is using your class (or method) and wants to know “what it does” at a high level and/or “how to use it”
- This is different from *comments*, which are for a programmer who might be reading your code and wants to know the details of “how it works”

Javadocs

- As a shortcut, you can type the following right above a class (or method)

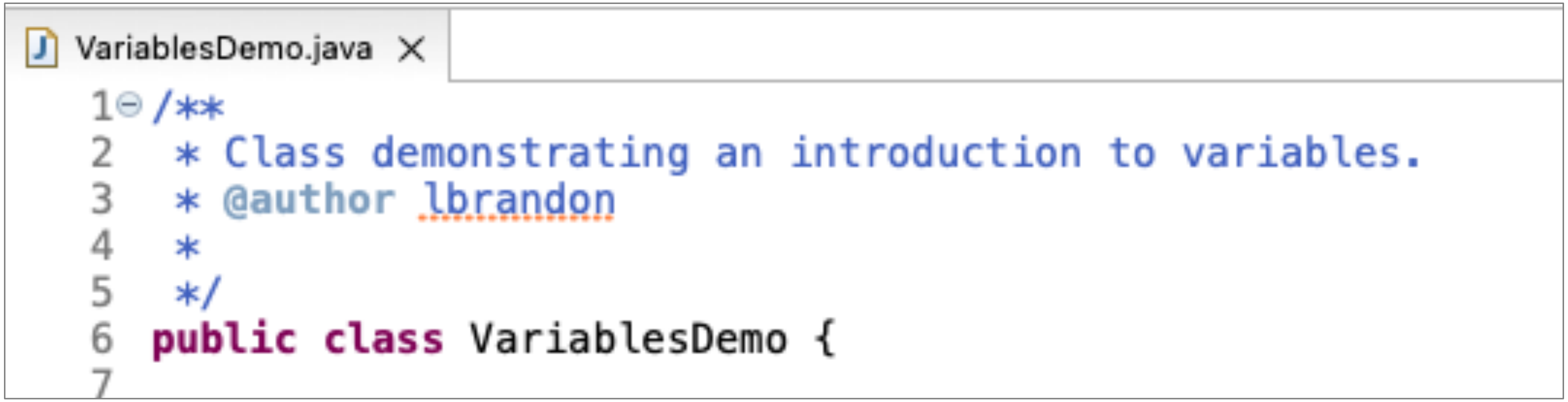
```
/**
```

  and then hit Enter

- It will add a javadoc block and you can fill in the rest

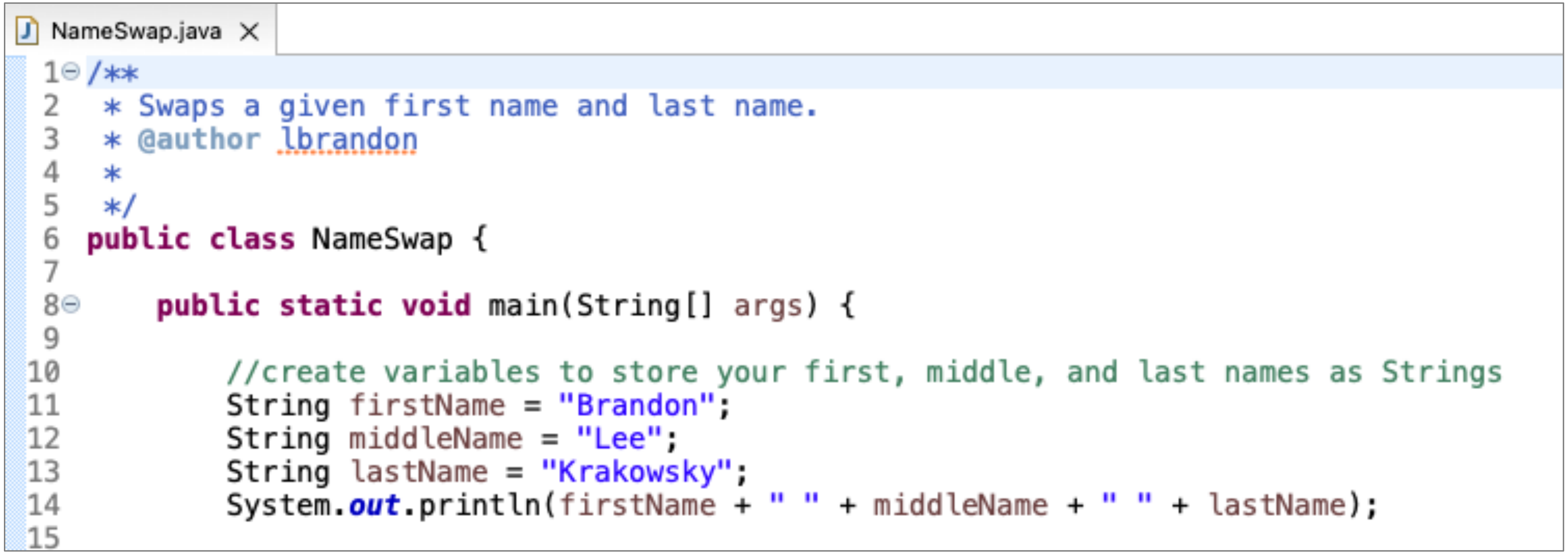
```
/**
 * Class demonstrating an introduction to variables.
 * @author lbrandon
 */
public class VariablesDemo {


}
```

Javadocs


VariablesDemo.java

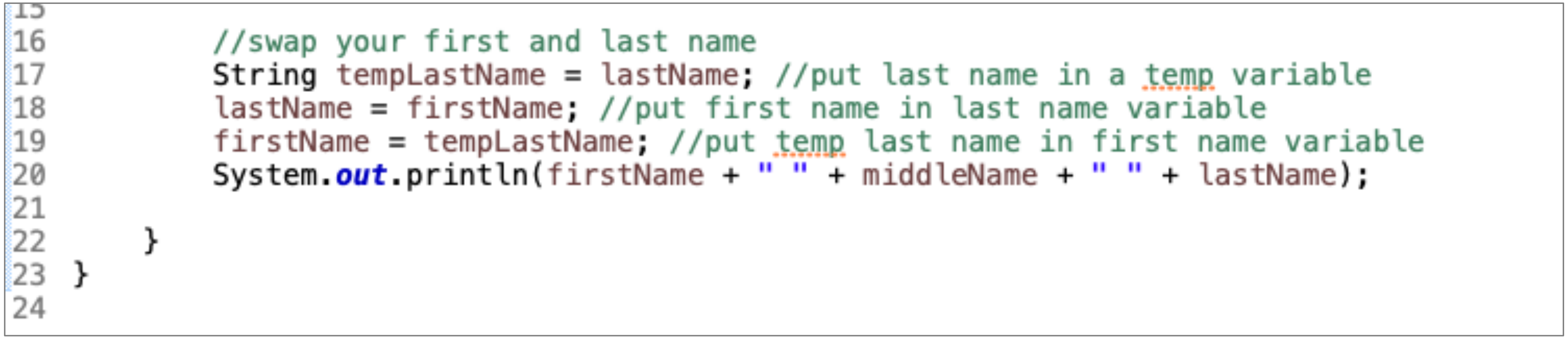
```
/**
 * Class demonstrating an introduction to variables.
 * @author lbrandon
 *
 */
public class VariablesDemo {

```

Variables – Name Swap Exercise


NameSwap.java

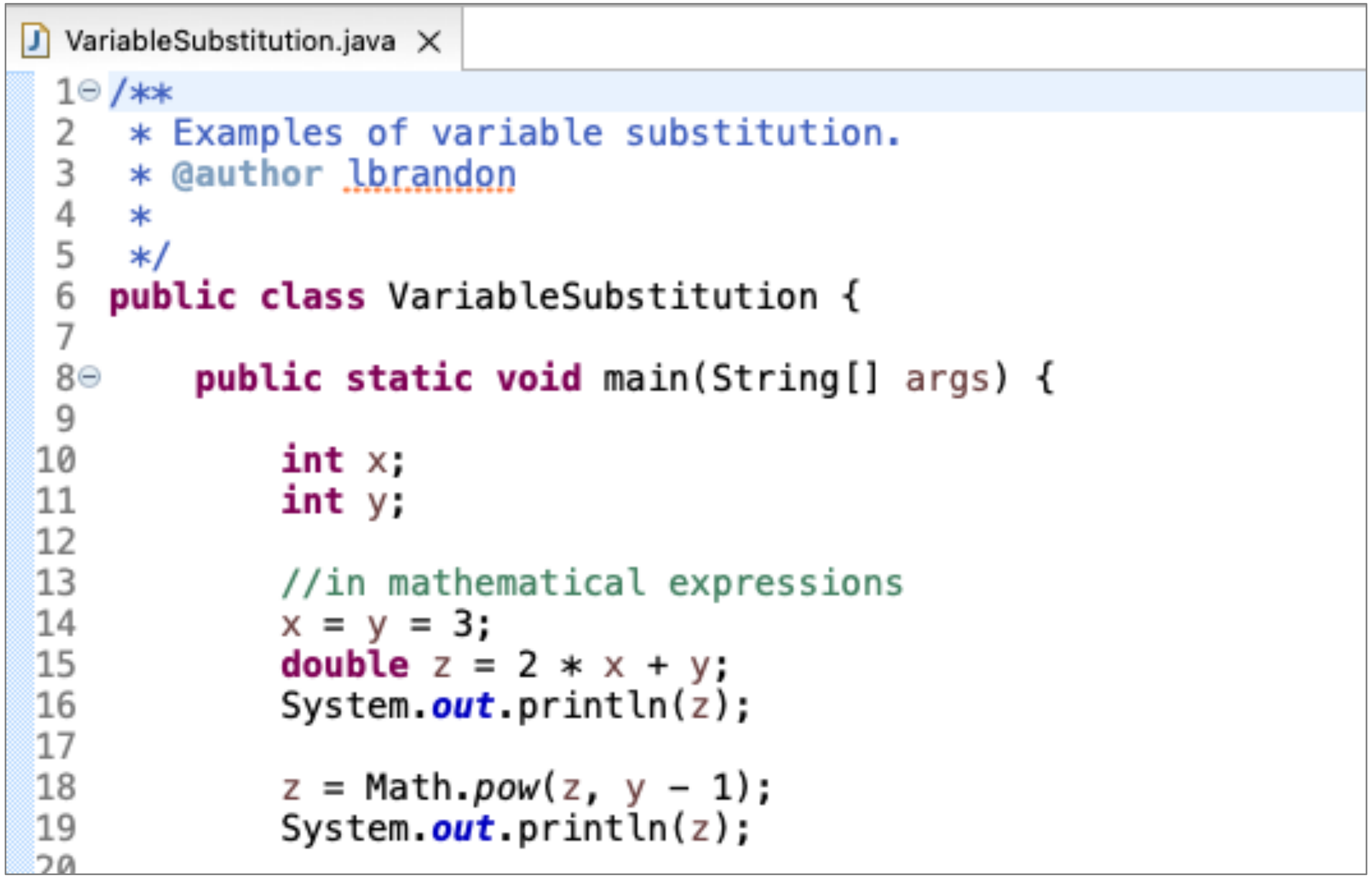
```java
/**
 * Swaps a given first name and last name.
 * @author lbrandon
 *
 */
public class NameSwap {

    public static void main(String[] args) {

        //create variables to store your first, middle, and last names as Strings
        String firstName = "Brandon";
        String middleName = "Lee";
        String lastName = "Krakowsky";
        System.out.println(firstName + " " + middleName + " " + lastName);

```

Variables – Name Swap Exercise



```
        //swap your first and last name
        String tempLastName = lastName; //put last name in a temp variable
        lastName = firstName; //put first name in last name variable
        firstName = tempLastName; //put temp last name in first name variable
        System.out.println(firstName + " " + middleName + " " + lastName);

    }
}
```

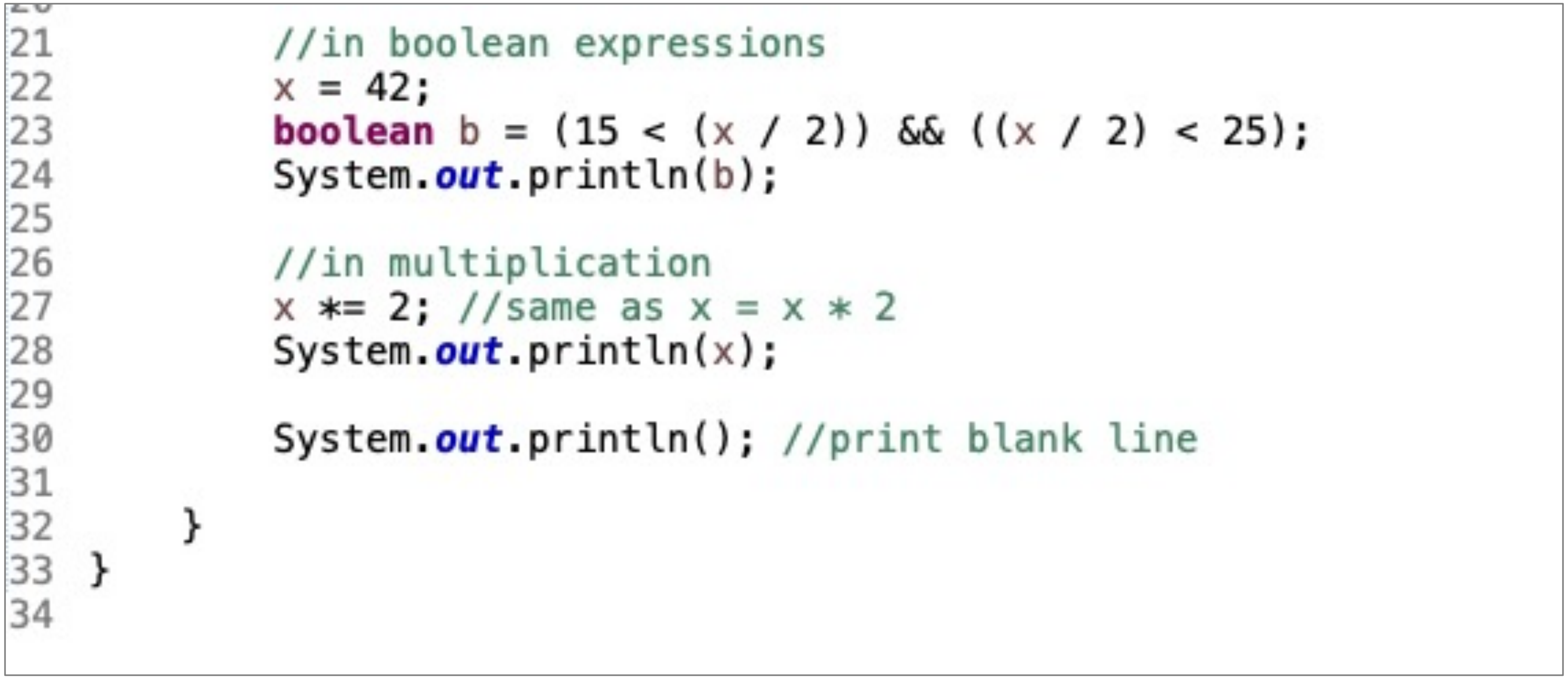
Variables – Variable Substitution



VariableSubstitution.java

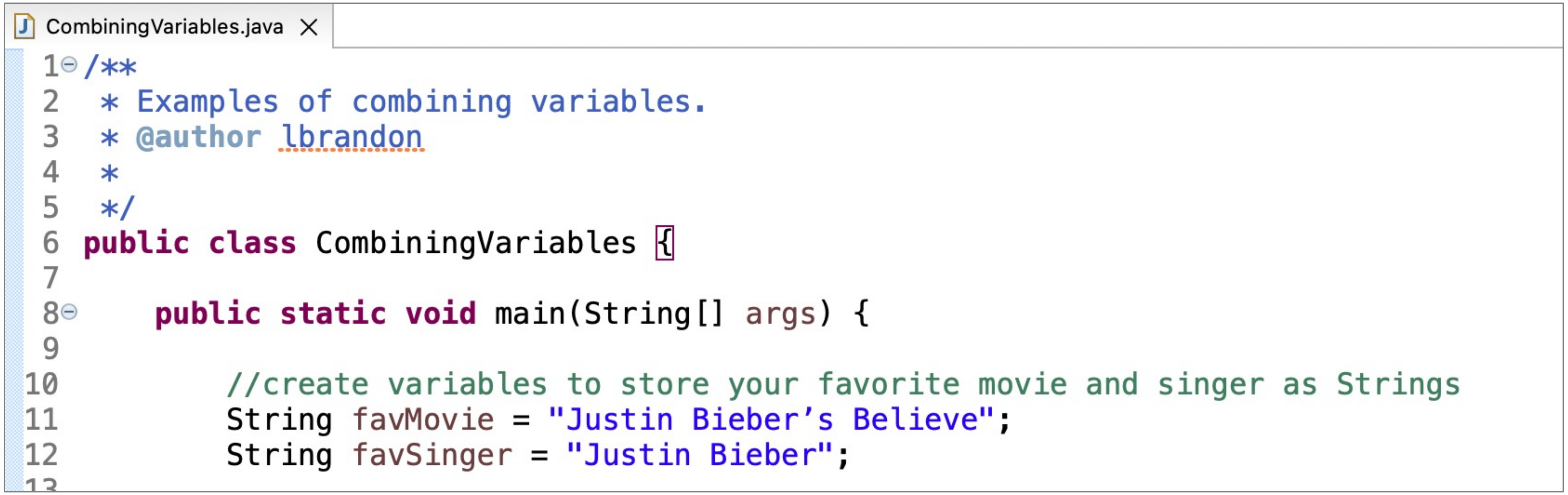
```java
/**
 * Examples of variable substitution.
 * @author lbrandon
 *
 */
public class VariableSubstitution {

    public static void main(String[] args) {

        int x;
        int y;

        //in mathematical expressions
        x = y = 3;
        double z = 2 * x + y;
        System.out.println(z);

        z = Math.pow(z, y - 1);
        System.out.println(z);
```

Variables – Variable Substitution



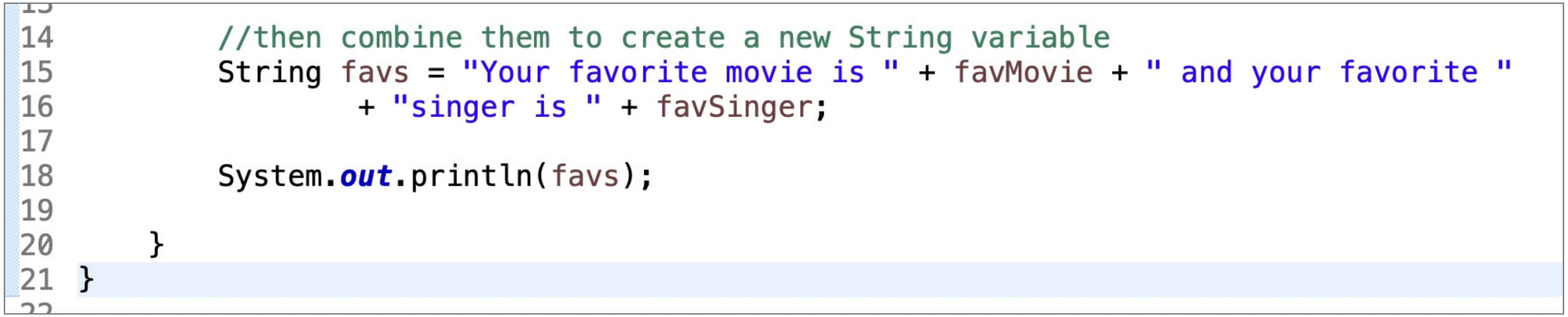
```java
        //in boolean expressions
        x = 42;
        boolean b = (15 < (x / 2)) && ((x / 2) < 25);
        System.out.println(b);

        //in multiplication
        x *= 2; //same as x = x * 2
        System.out.println(x);

        System.out.println(); //print blank line

    }
}
```

Combining Variables


CombiningVariables.java

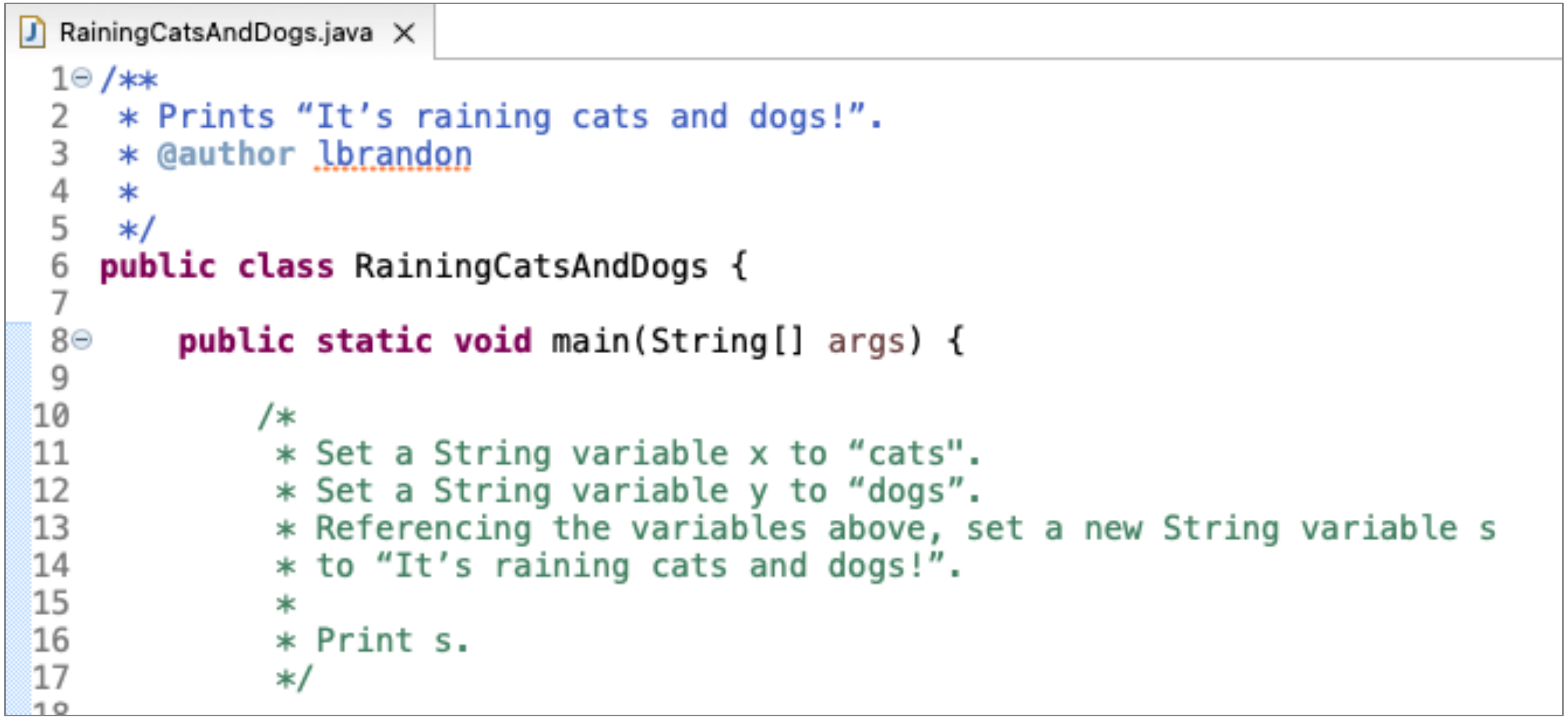
```
/**
 * Examples of combining variables.
 * @author lbrandon
 *
 */
public class CombiningVariables {

    public static void main(String[] args) {

        //create variables to store your favorite movie and singer as Strings
        String favMovie = "Justin Bieber’s Believe";
        String favSinger = "Justin Bieber";
```

Combining Variables



```
        //then combine them to create a new String variable
        String favs = "Your favorite movie is " + favMovie + " and your favorite "
                + "singer is " + favSinger;

        System.out.println(favs);

    }
}
```

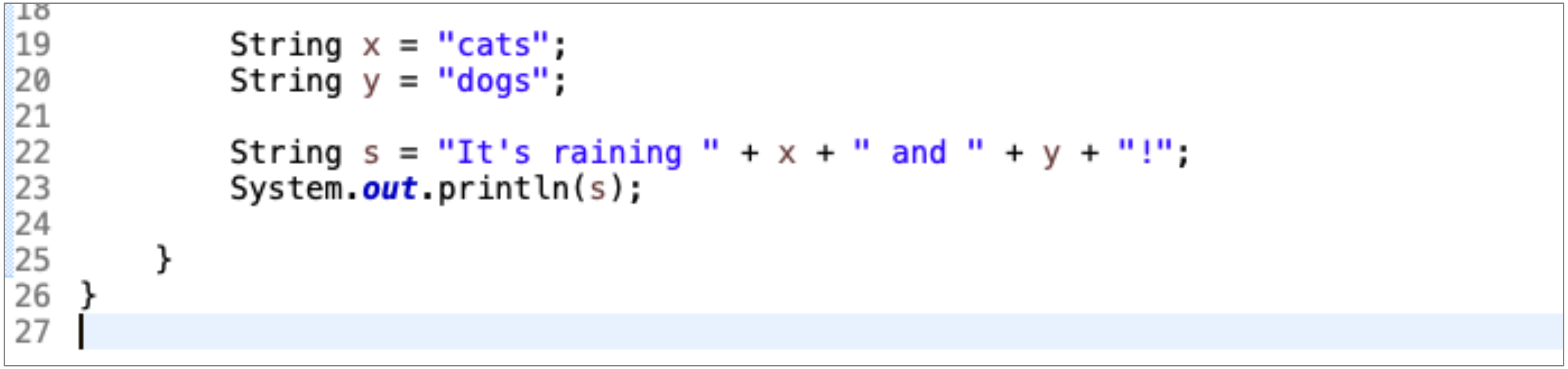
Variables – VERY Simple Exercise



RainingCatsAndDogs.java

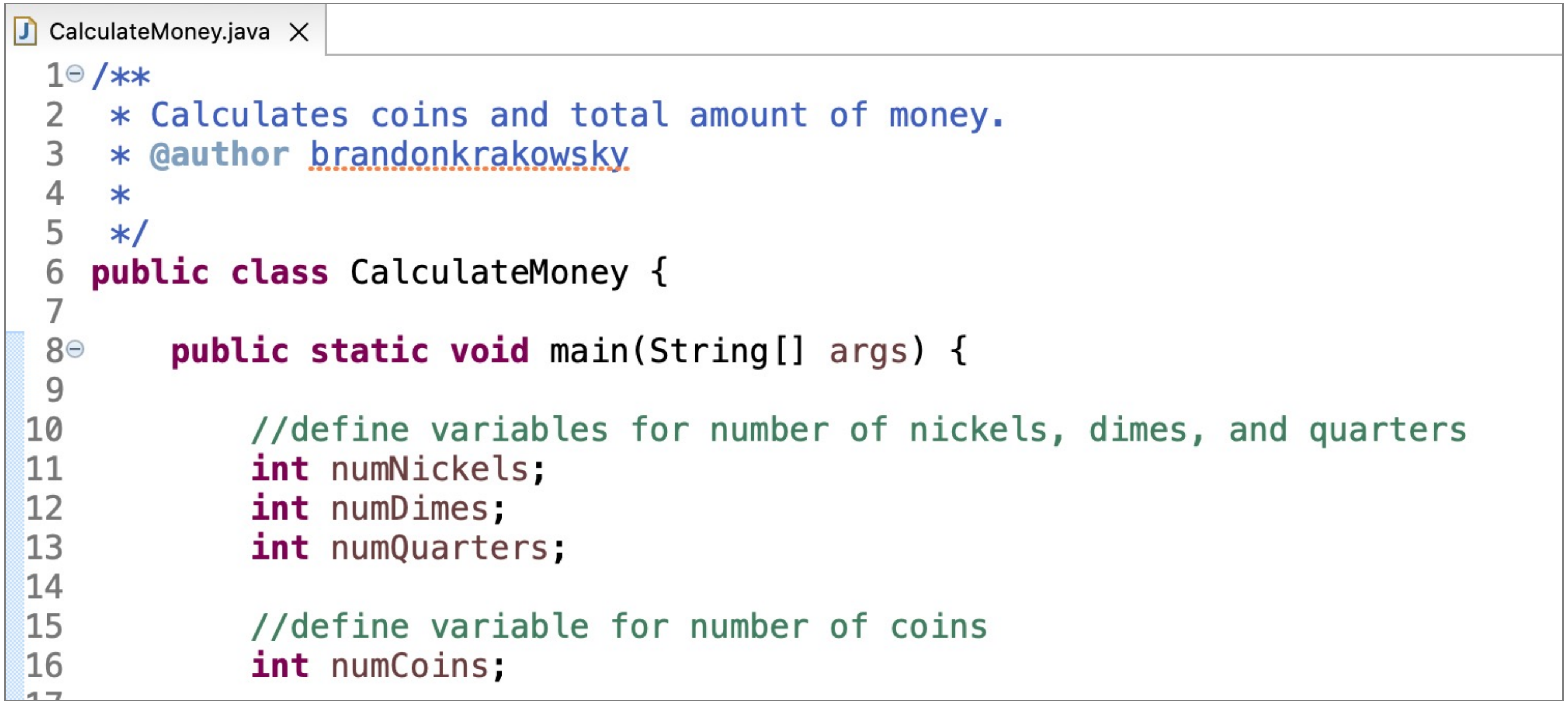
```java
/**
 * Prints “It’s raining cats and dogs!”.
 * @author lbrandon
 *
 */
public class RainingCatsAndDogs {

    public static void main(String[] args) {

        /*
         * Set a String variable x to “cats".
         * Set a String variable y to “dogs”.
         * Referencing the variables above, set a new String variable s
         * to “It’s raining cats and dogs!”.
         *
         * Print s.
         */
```

Variables – VERY Simple Exercise



```
        String x = "cats";
        String y = "dogs";

        String s = "It's raining " + x + " and " + y + "!";
        System.out.println(s);

    }
}
```

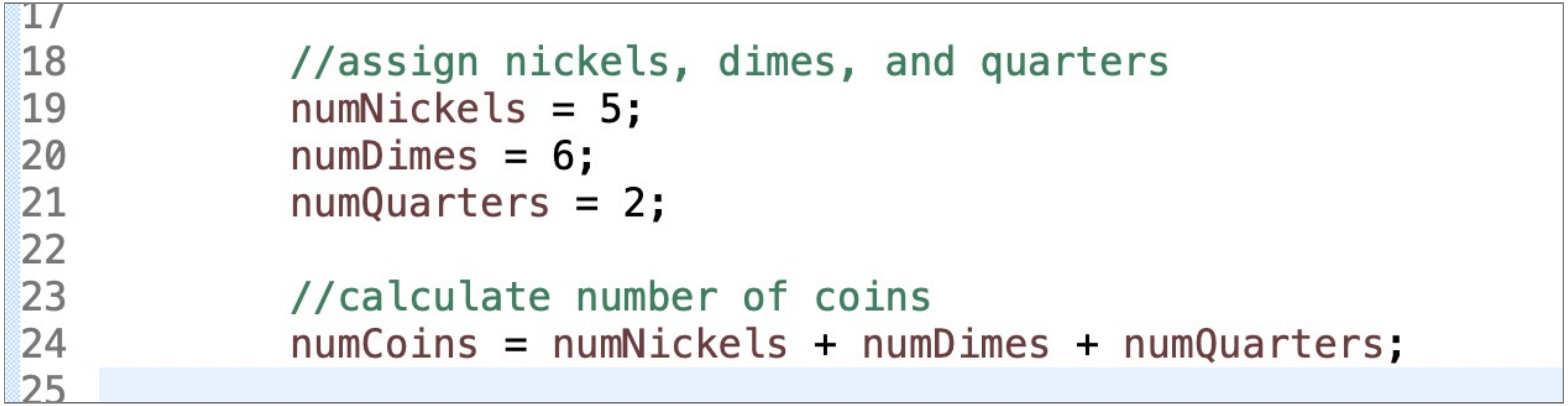
Calculating Total Amount of Money



CalculateMoney.java

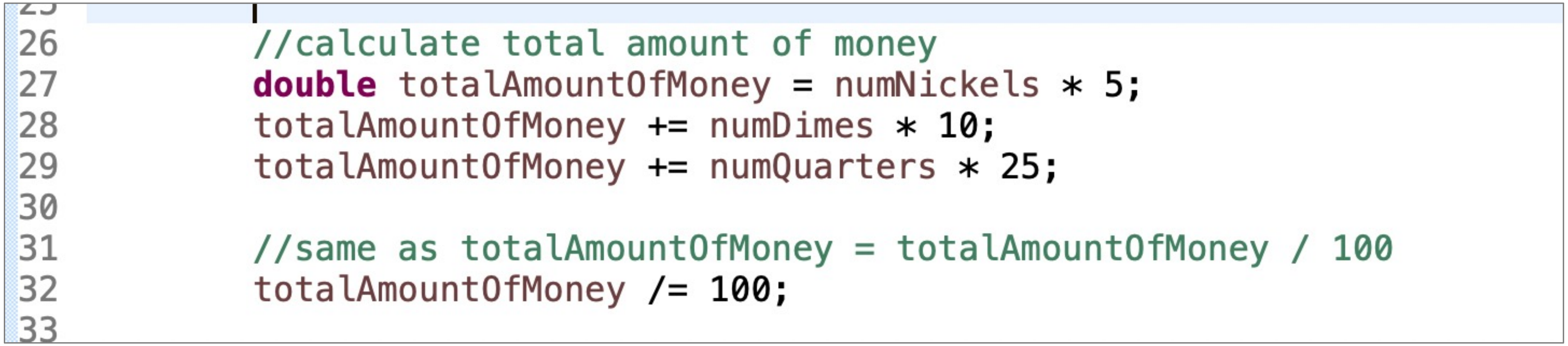
```
/**
 * Calculates coins and total amount of money.
 * @author brandonkrakowsky
 *
 */
public class CalculateMoney {

    public static void main(String[] args) {

        //define variables for number of nickels, dimes, and quarters
        int numNickels;
        int numDimes;
        int numQuarters;

        //define variable for number of coins
        int numCoins;
```

Calculating Total Amount of Money



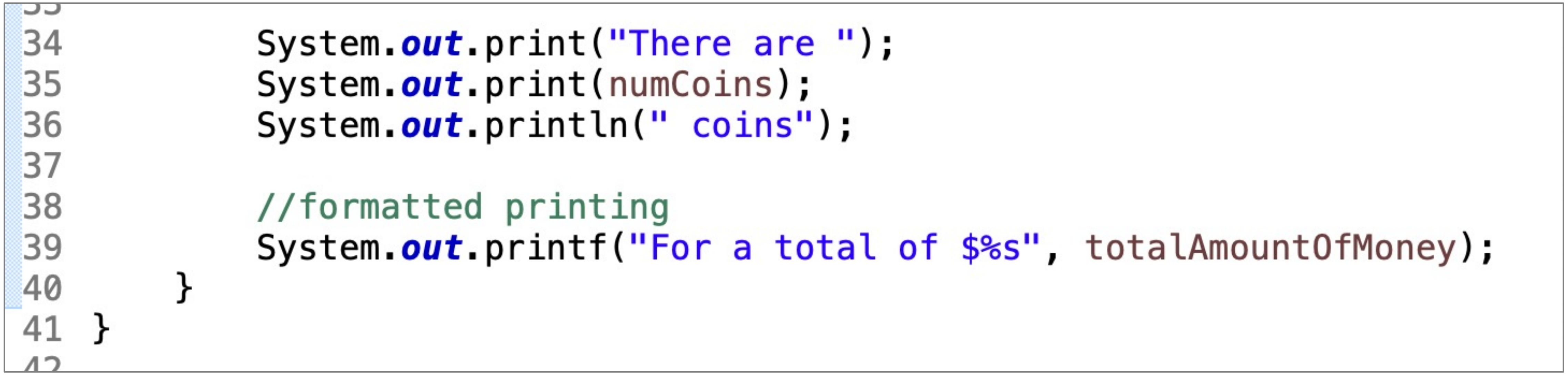
```
		//assign nickels, dimes, and quarters
		numNickels = 5;
		numDimes = 6;
		numQuarters = 2;

		//calculate number of coins
		numCoins = numNickels + numDimes + numQuarters;

```

Calculating Total Amount of Money



```
        //calculate total amount of money
        double totalAmountOfMoney = numNickels * 5;
        totalAmountOfMoney += numDimes * 10;
        totalAmountOfMoney += numQuarters * 25;

        //same as totalAmountOfMoney = totalAmountOfMoney / 100
        totalAmountOfMoney /= 100;

```

Calculating Total Amount of Money



```
        System.out.print("There are ");
        System.out.print(numCoins);
        System.out.println(" coins");

        //formatted printing
        System.out.printf("For a total of $%s", totalAmountOfMoney);
    }
}
```

Homework 2

Homework 2

Will be assigned by tonight, Wednesday, 09/07/22 at midnight and due Wednesday, 09/14/22 at midnight

- For this Java assignment, there are the five mini-programs we want you to code
- The topics are: Variables and data types

To complete the assignment:

- Download the provided class files from Canvas
- Complete the programs by writing code
- Submit your completed class files to Canvas