Java Programming I

Session 2

JAVA PROGRAMS, DATA, VARIABLES, AND CALCULATIONS

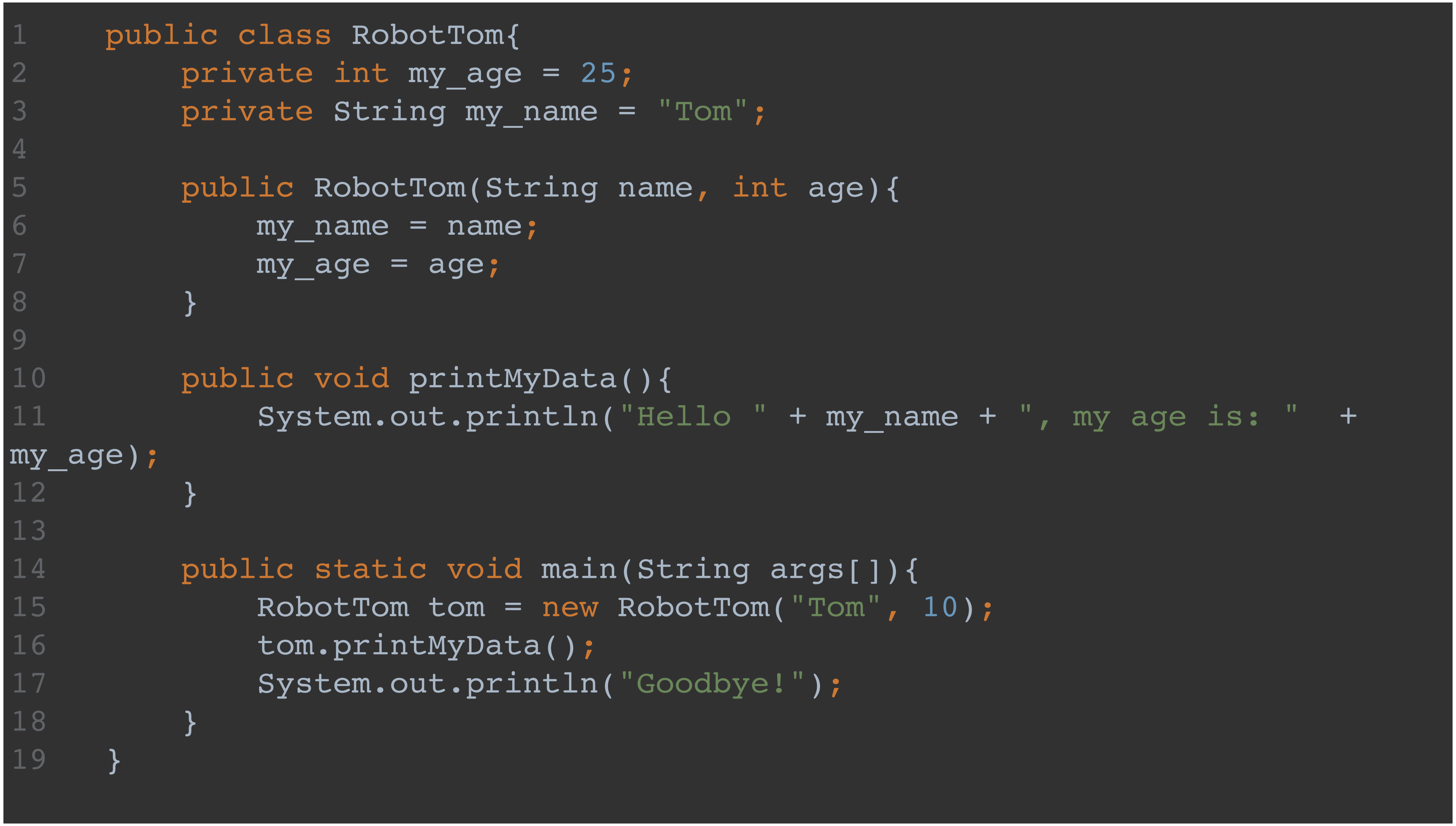
Juan Carlos Moreno - UCLA Ex

Agenda

- Review of last session
- Program structure
- Data and variables
- Data Types
- Arithmetic expressions
- Logical operations

Everything is a class

Object oriented no matter what



```
public class RobotTom{
    private int my_age = 25;
    private String my_name = "Tom";

    public RobotTom(String name, int age){
        my_name = name;
        my_age = age;
    }

    public void printMyData(){
        System.out.println("Hello " + my_name + ", my age is: "  + my_age);
    }

    public static void main(String args[]){
        RobotTom tom = new RobotTom("Tom", 10);
        tom.printMyData();
        System.out.println("Goodbye!");
    }
}
```

Java Keywords

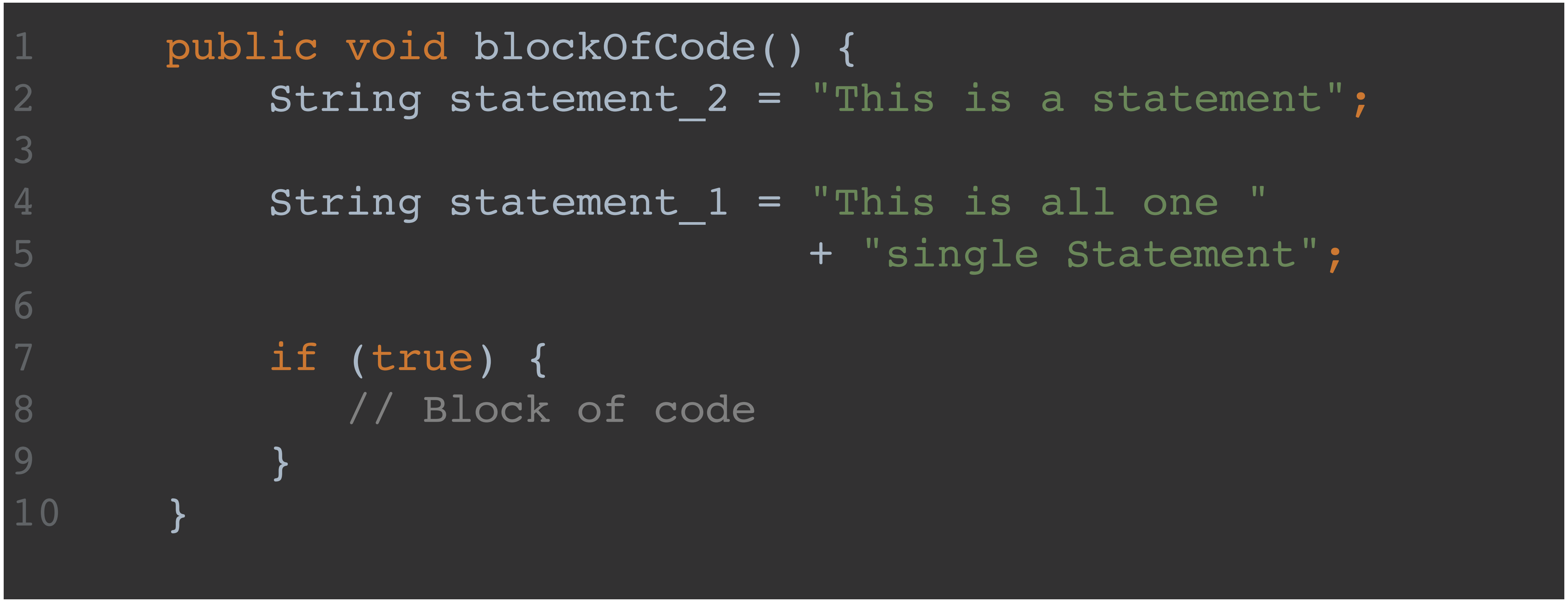
You can’t name variables using these reserved words

abstract	continue	for	new	switch
assert***	default	goto*	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum****	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp**	volatile
const*	float	native	super	while

* not used
** added in 1.2
*** added in 1.4
**** added in 5.0

Statements & Blocks

Ingredients of a program



```
public void blockOfCode() {
    String statement_2 = "This is a statement";

    String statement_1 = "This is all one "
                         + "single Statement";


    if (true) {
       // Block of code
    }
}
```

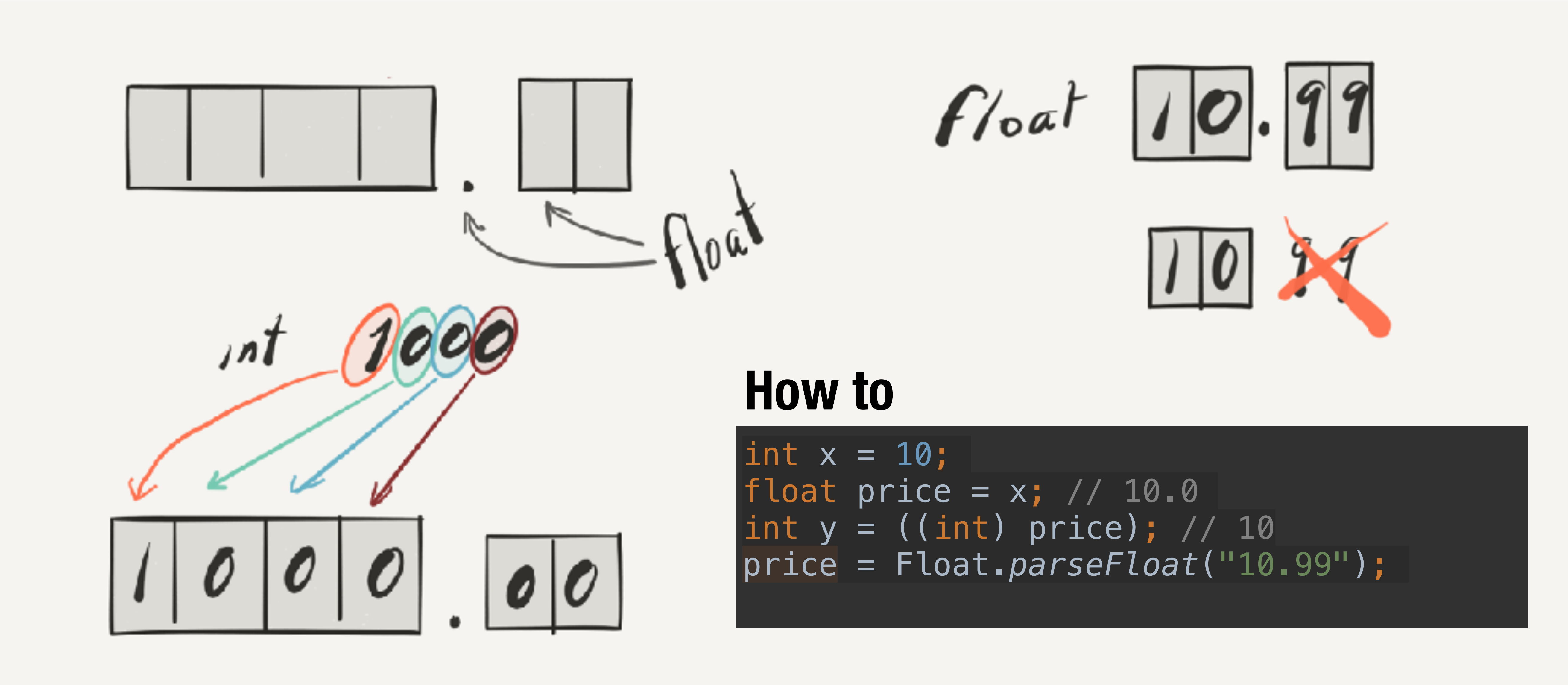
Primitive Data Types

Representing data using text

Type	Meaning	Bits	Range
boolean	True or False		0 or 1
byte	8 bit integer	1 Byte	-128 - 127
char	Character	4 bytes	0 to 1,112,064
double	Very precise number	8 bytes	
float	Precise number	4 bytes	
int	Integer	4 bytes	-2 billion - 2 billion
long	Very long integer	8 bytes	-9 Quintillion - 9 Quintillion
short	Short Integer		-32,768 - 32,767

Casting

Not the Hollywood type

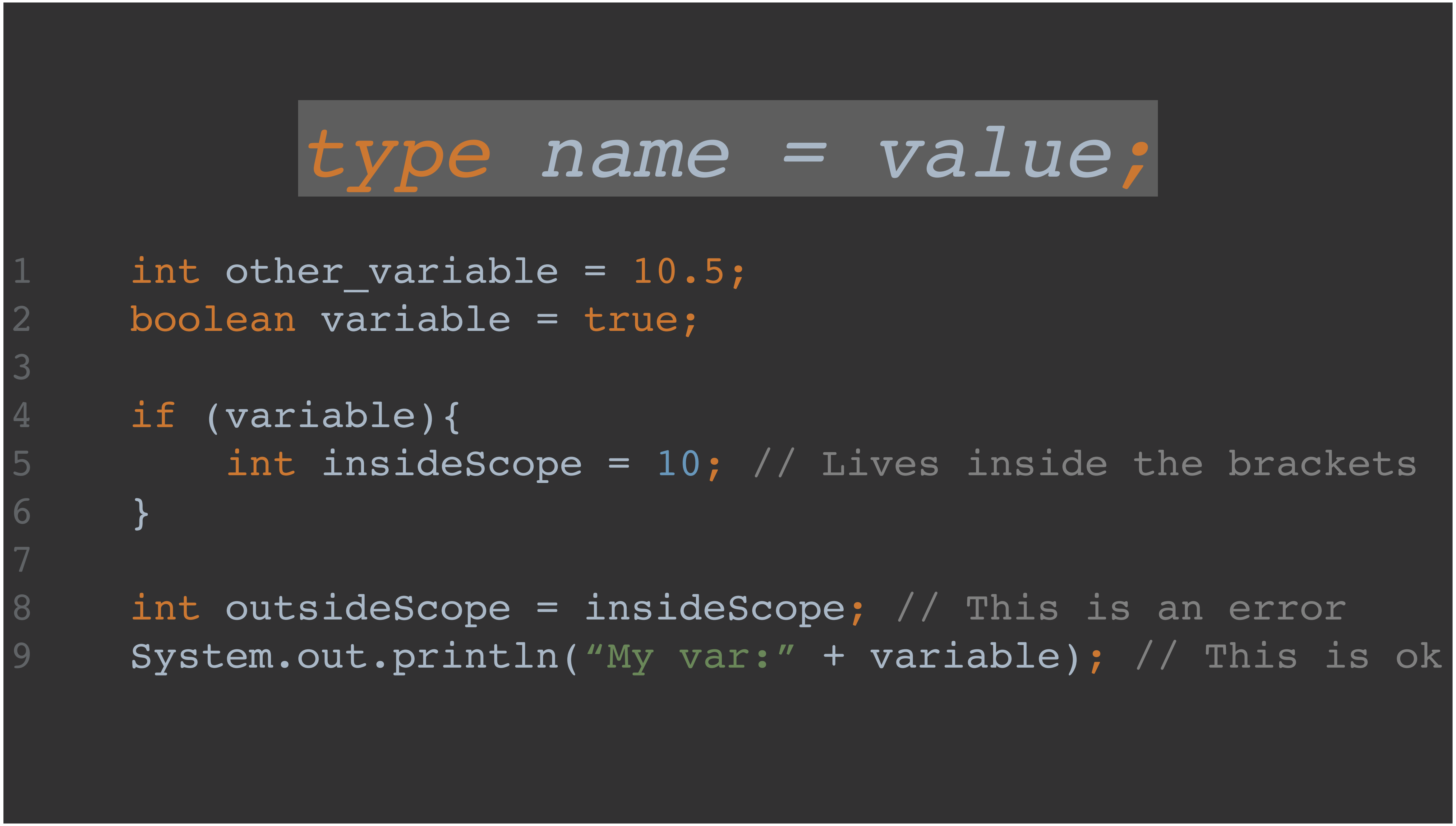


How to

```java
int x = 10;
float price = x; // 10.0
int y = ((int) price); // 10
price = Float.parseFloat("10.99");
```

Variables

Initialization, Scope and lifetime



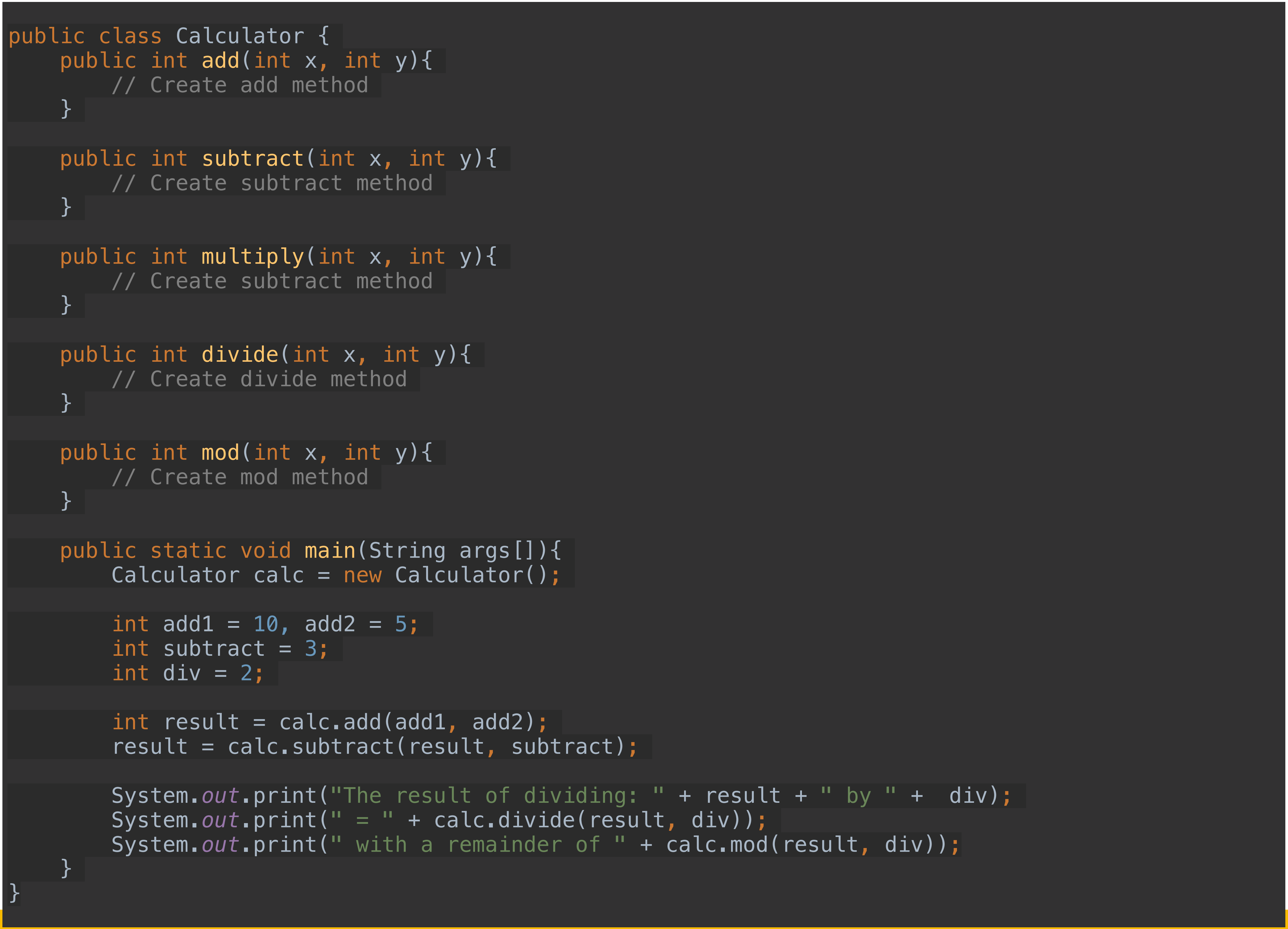
```
type name = value;
```

```
int other_variable = 10.5;
boolean variable = true;

if (variable){
    int insideScope = 10; // Lives inside the brackets
}

int outsideScope = insideScope; // This is an error
System.out.println(“My var:” + variable); // This is ok
```

Arithmetic Operators

How to declare variables

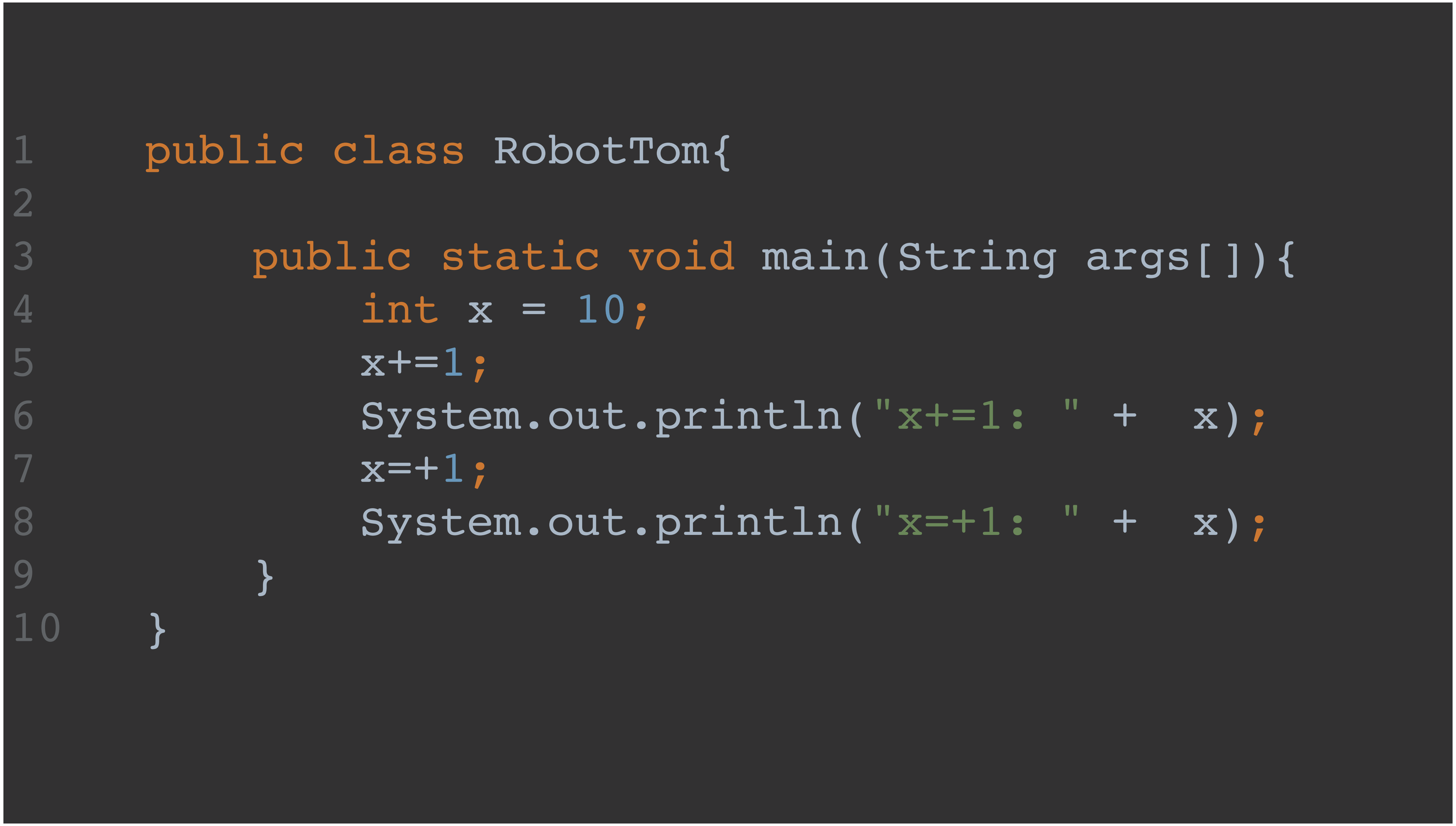
+	Add / Concat
-	Subtract
*****	Multiply
/	Divide
%	Mod
++	+1
—	-1
+=	+ *number*
-=	- *number*



```java
public class Calculator {
    public int add(int x, int y){
        // Create add method
    }

    public int subtract(int x, int y){
        // Create subtract method
    }

    public int multiply(int x, int y){
        // Create subtract method
    }

    public int divide(int x, int y){
        // Create divide method
    }

    public int mod(int x, int y){
        // Create mod method
    }

    public static void main(String args[]){
        Calculator calc = new Calculator();

        int add1 = 10, add2 = 5;
        int subtract = 3;
        int div = 2;

        int result = calc.add(add1, add2);
        result = calc.subtract(result, subtract);

        System.out.print("The result of dividing: " + result + " by " +  div);
        System.out.print(" = " + calc.divide(result, div));
        System.out.print(" with a remainder of " + calc.mod(result, div));
    }
}
```

Arithmetic Operators

A slight difference



```java
public class RobotTom{

    public static void main(String args[]){
        int x = 10;
        x+=1;
        System.out.println("x+=1: " +  x);
        x=+1;
        System.out.println("x=+1: " +  x);
    }
}
```

Arithmetic Operators

Operator precedence

High	**++ or —**
🟩	*** then /**
🟩	**+ then -**
🟨	**== then !=**
🟧	**&&**
🟥	**||**
Low	**=**

100 + 2 * 10 / 4 = 105

2 * 10 = 20 → 20 / 4 = 5 → 5 + 100

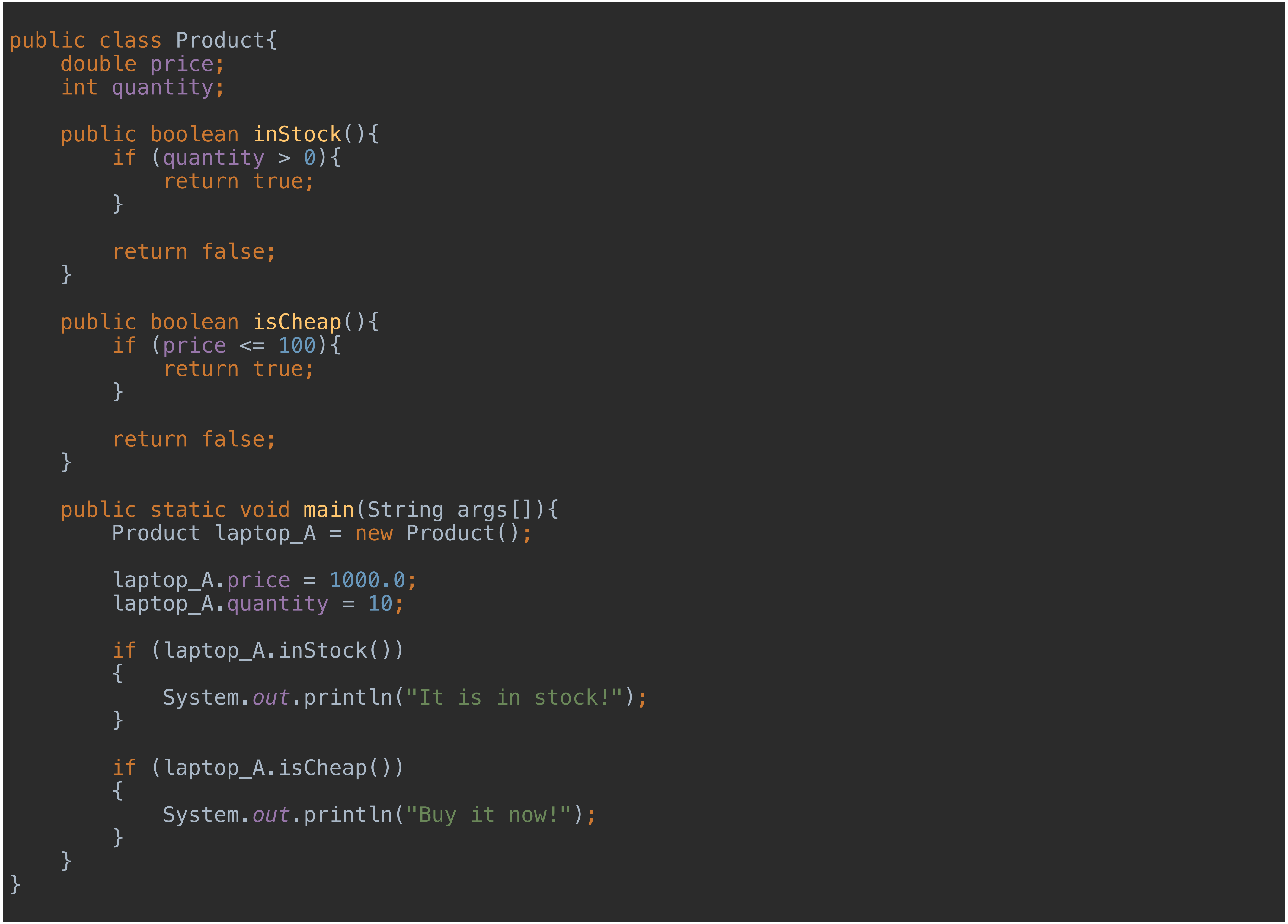
Logical Operators

How to declare variables

==	Equal to
!=	Not equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

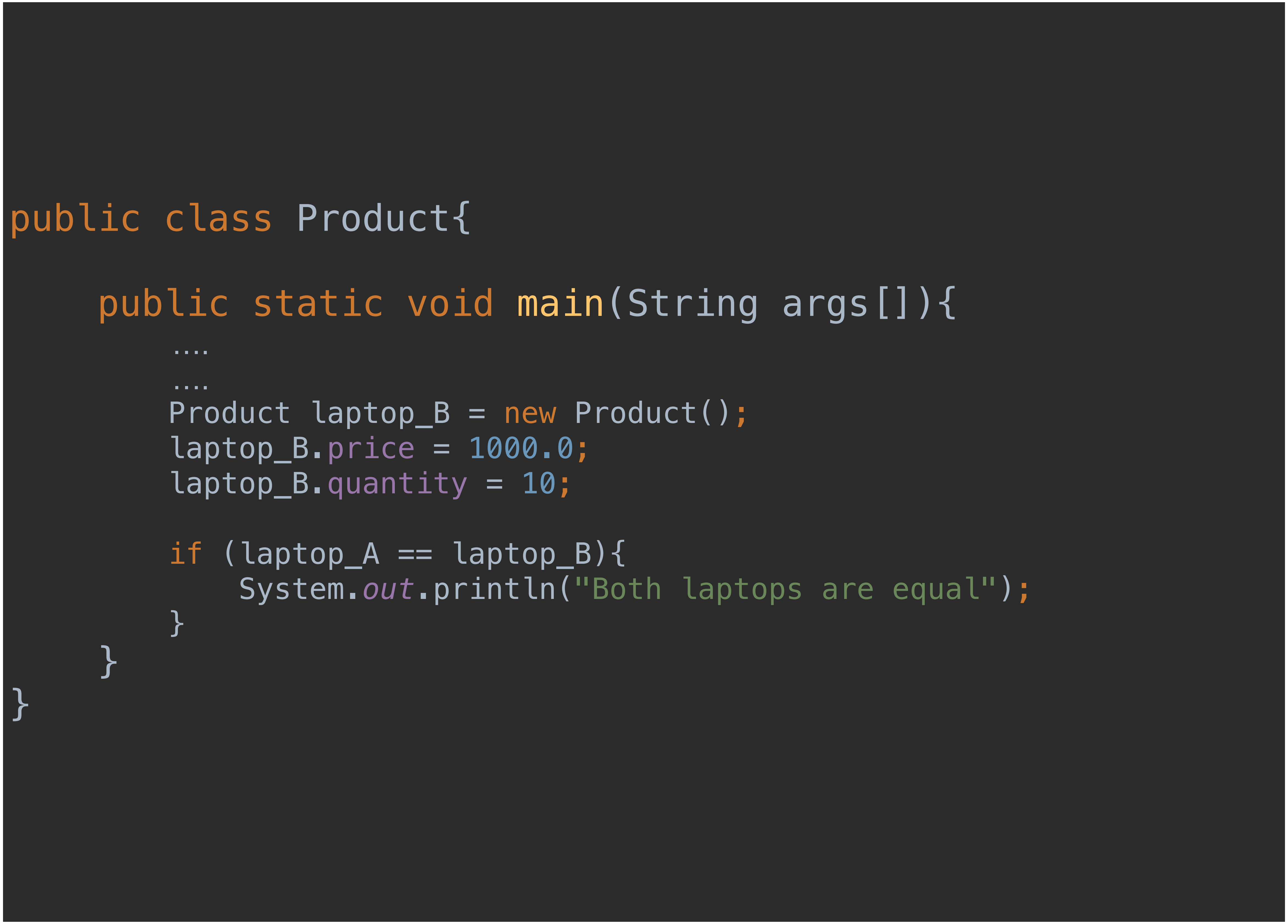
Logical Operators

Follow up



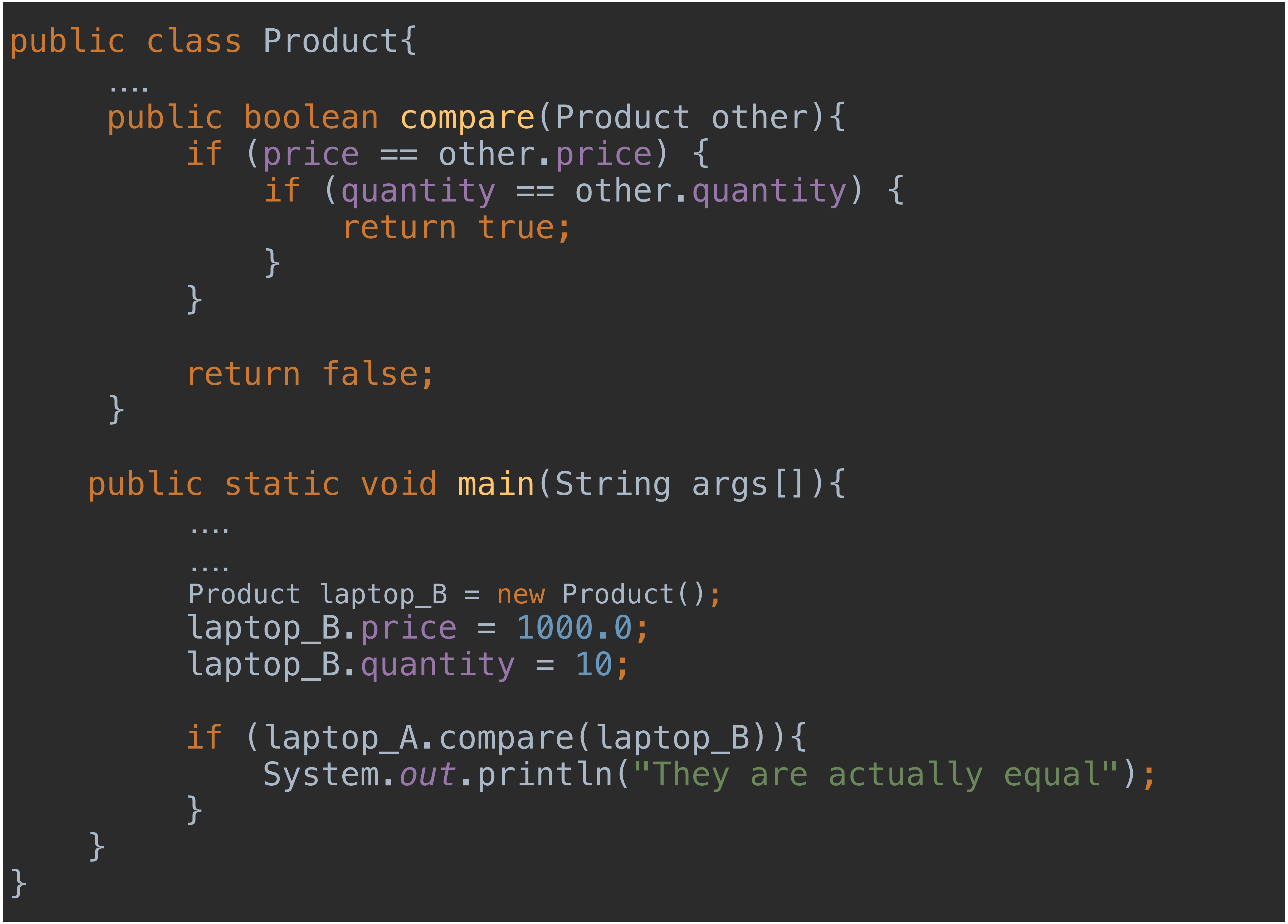
```java
public class Product{
    double price;
    int quantity;

    public boolean inStock(){
        if (quantity > 0){
            return true;
        }

        return false;
    }

    public boolean isCheap(){
        if (price <= 100){
            return true;
        }

        return false;
    }

    public static void main(String args[]){
        Product laptop_A = new Product();

        laptop_A.price = 1000.0;
        laptop_A.quantity = 10;

        if (laptop_A.inStock())
        {
            System.out.println("It is in stock!");
        }

        if (laptop_A.isCheap())
        {
            System.out.println("Buy it now!");
        }
    }
}
```

Logical Operators



```java
public class Product{

    public static void main(String args[]){
        ….
        ….
        Product laptop_B = new Product();
        laptop_B.price = 1000.0;
        laptop_B.quantity = 10;

        if (laptop_A == laptop_B){
            System.out.println("Both laptops are equal");
        }
    }
}
```

Logical Operators



```
public class Product{
     ....
     public boolean compare(Product other){
         if (price == other.price) {
             if (quantity == other.quantity) {
                 return true;
             }
         }

         return false;
     }

    public static void main(String args[]){
         ....
         ....
         Product laptop_B = new Product();
         laptop_B.price = 1000.0;
         laptop_B.quantity = 10;

         if (laptop_A.compare(laptop_B)){
             System.out.println("They are actually equal");
         }
    }
}
```

Logical Operators

Class Exercise

Or

A	B	A or B
0	0	0
0	1	1
1	0	1
1	1	1

And

A	B	A and B
0	0	0
0	1	0
1	0	0
1	1	1

```
public class TruthTable{

    public static void main(String args[]){
        printOrTable();
        printAndTable();
    }
}
```

Comments

A little bonus material

```
// Single line comment

/*
   Multi Line
   Comment
*/
```