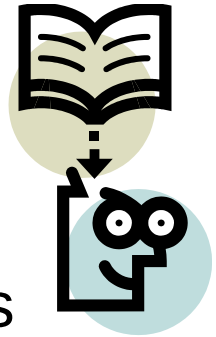
CS620c Structured Programming Lesson 6

Joe Duffin

Email: Joseph.Duffin@nuim.ie

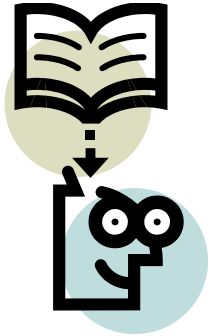
Syntax Errors



- A syntax error occurs when the source code contains something that is not valid “Java” code.

- Classic errors include
  - Missing a ; on end of line.
  - Unpaired\misplaced brackets { }, [ ], ( ).
  - Using a keyword as a variable name.
  - Copy and paste errors, some characters look similar but are not equivalent.
  - Confusing similar looking characters, 0 and O, 1 and l. Since o, O and Q look like zero they are not normally used as a variable name.
  - Java is case sensitive e.g. *String* and *string* are not the same.
  - The class name within the file must be the same as the file name (with .java extension).

Logical (Runtime) Errors



- A logical error occurs when a program compiles but does not run as expected.
- These problems can be very difficult to solve so the best practice is try to get it right from the start, don't let the problem in.
- Use top down design, piecewise refinement to primitives and only then translate to Java.
- Comment the code as you go, use good indentation and obey conventions (e.g. variable names lowercase, class names first letter capital etc).
- Be sure to use test data that you know the correct output for during the commissioning phase of your program.

Numeric Operators (Revisited) (1)

Operator	Result
+	Addition
-	Subtraction
*****	Multiplication
/	Division

- *Examples of the above can be found in PracticeOperatorsEasy.java – download and run this program* before you move on.

+=	Addition assignment
-=	Subtraction assignment
*=	Multiplication assignment
/=	Division assignment
%	Modulus (remainder)

See next set of slides for explanation

The Assignment Operator =

Do not confuse the assignment operator, =, with the equal sign from algebra, or any other pre-conceived idea you have about the equals sign!

This is a new concept, embrace it,….

There can only be one variable on the left hand side of the assignment operator.

The operator copies the value evaluated in the expression to the right hand side of the operator into the variable on the left hand side of the operator.

In maths class the following line which is valid Java seems like a problem.

```
x=x+1
```

As you would rewrite it as follows

```
x-x=1
```

or

```
0 = 1
```

Problem!

```
int a=5;
int b=10;
int c=0;
```

Name	Value
c	0
b	10
a	5

Memory

Step 1

```
c=a*b;
```

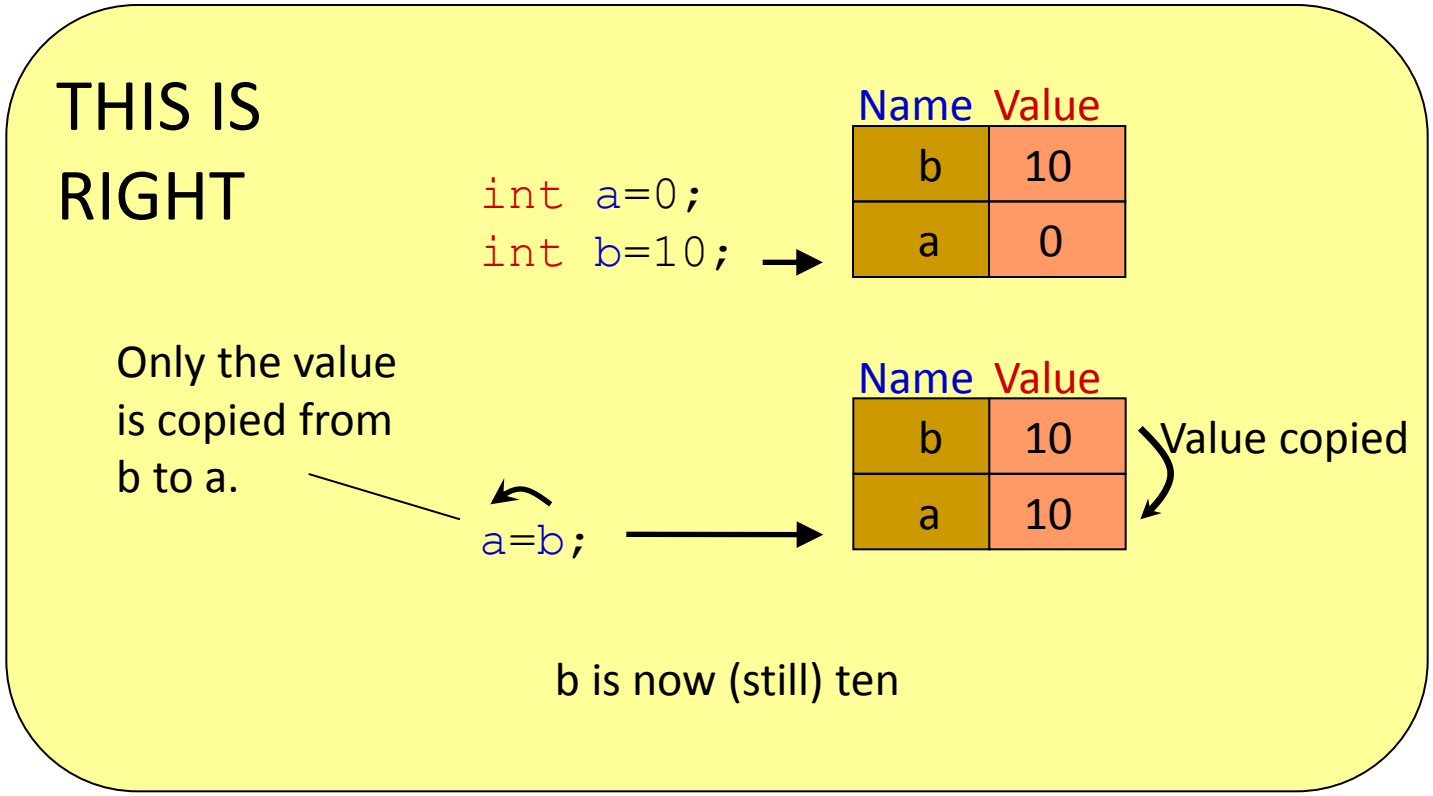
5x10=50

Name	Value
c	50
b	10
a	5

Memory

Step 2

The Assignment Operator,= , eliminating the misconceptions



The above three lines of code produces b equal to 10 after running. About 90% of a class will arrive at this value on first viewing however 10% can see a different result of b equal to zero. It well worth the time make sure you know why b is ten before you proceed if you are one of the 10% who thought it was zero.

The Assignment Operator,= , eliminating the misconceptions

THIS IS WRONG

```
int a=0;
int b=10;
```

Name	Value
b	10
a	0

b is “put into” a

```
a=b;
```

Name	Value
b	
a	10

emptied

b is now empty so it must be zero!

Students who predict that the above three lines of code produces b equal to 0 after running need to revisit their understanding of the assignment operator until they can see why the correct answer is 10, see previous slide.

Observation thanks to D. O’Donoghue

Assignment Operators cont.

- There are a number of ways to (re)assign a value once you have given it a type

```
int num1 =10; // creates an int called num1
              // that stores the value 10
int num2 = 100; // creates an int called num2
                // that stores the value 100
num1 = num1+num2; // adds num1 to num2 and stores
                  // the result in num1 (num1 will
                  // have the value 110)
num1+=num2;// means exactly the same as num1=num1+num2;
           // So if you see num1+=num2 you should
          // interpret it as num1=num1+num2;
```

Run `PracticeOperators.java` now for examples

Modulus

- % means modulus (mod) in Java and it calculates the remainder after division
- It can be used on real and whole numbers. If you use mod on whole numbers, you always get a whole number as a remainder.
- What's the remainder?
  - 100 /10 /* the remainder is */
  - 100/ 9 /*the remainder is 1 (9 divides into 100 eleven times with one left over) */
  - 100.0/ 9.1 /* 9.1 divides into 100.0 ten times with a remainder of approximately 9 (actually the computer representation is 8.999996 –more on this later) */
- Download and run `Modulus.java`

Modulus continued

- Remember:

```
int quotient = 7 / 3; // yields 2
int remainder = 7 % 3; // yields 1
```

- The modulus operator turns out to be surprisingly useful. For example, you can check whether one number is divisible by another: if x % y is zero, then x is divisible by y.

Stop!, Can you answer these?

- Name the eight primitive types in Java
- How to you declare and initialise multiple variables of the same type on the one line
- How many times do you state a variables type in a program?
- What is modulus?
- What type of remainder do you get when you mod whole numbers?