

More Types

Sep 18

double, boolean

HW1 notes

- Java conventions
 - classes start with initial capital letter ---- `public class MyDemoClass`
 - variables start with initial lower case letter --- `int aSingleInteger;`
 - methods start with an initial lower case letter --- `public static void main.....`
 - after that, use CaMeL case
- Use
 - `int` rather than `Integer`
 - `double` rather than `Double`
- Spaces in file names -- just say NO
- 100% max
- Concatenation and the `++` problem
- `ls -a` and `cat` to check what you are submitting
- `"package XXX"` appears at the top of your program

Double Fun

```
public class FunWithDouble {
    public static void main(String[] args) {
1        double a = 11.4;
2        double b = 16;
3        double c = a + b;
4        c = 4.2e-5;

5        double k = 19 + 0.25;
6        double m = k / 3.5 ;
7        String valu = "3.14159";
8        double pi = Double.parseDouble(valu);

9        double x = 3 * 4;

10       double y = 3 / 4;

11       int a = 2;
12       double b = a + 3.4;

13       int mi = 8;
14       double n = 7.7;
15       int z = mi + n; //
    }
}
```

	a	b	c	k	m	valu	pi	mi	z
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									

Converting to / from double

- String to double
 - `double baseDouble = Double.parseDouble(baseString);`
- int to double
 - `double baseDouble2 = baseInt; // automatic`
- double to int
 - ~~`int baseInt = baseDouble;`~~ // NOT automatic
 - MUST "cast"
 - `int baseInt = (int)baseDouble;`

Casting

converting one type to another

- Often Java will convert types for you
- Sometimes, you need to tell Java exactly what you want
 - this is called "Casting"
- For example, the code at right fails to compile with the message

```
Casting1.java:6: error: incompatible types: possible  
lossy conversion from double to int  
    int resultInt = aDouble * anInt;
```

- To fix, cast
 - change line to `(int)(aDouble*anInt);`
- When you do this, what does the program print?

```
public static void main(String[] args) {  
    double aDouble = 2.2;  
    int anInt = 4;  
    double resultDouble = aDouble * anInt;  
    int resultInt = aDouble * anInt;  
  
    System.out.println(aDouble + " * " +  
anInt + " = " + resultDouble);  
  
    System.out.println(aDouble + " * " +  
anInt + " = " + resultInt);  
}
```

```
    int resultIntB = (int) aDouble * anInt;  
    System.out.println(aDouble + " * " +  
anInt + " = " + resultIntB);
```


Whiteboards

- The formula for "wind chill" is

$$\text{windChill} = 35.74 + 0.6215 * \text{Temperature} + (0.4274 * \text{Temp} - 35.75) * \text{windSpeed}^{0.16}$$

- To compute a number raised to a "weird" power use Math.pow(a,b). e.g.

```
double number = 16;  
double power = 0.25;  
double res = Math.pow(number, power);  
System.out.println(res);
```

- Recall to get a double from the command line:

```
double argDouble = Double.parseDouble(args[0]);
```

- Write a program to compute (and show) the wind chill temperature given the current temperature and wind speed as command line arguments

Yet more Types, Boolean

- Just true or false.

```
boolean trueValue = true;  
boolean falseValue = false;
```

- if that was all, booleans would be boring



George Boole
1815--1864

Boolean Comparison Operators

- They return true or false
- > (also <, <=, >=)
 - Not =<
- ==
- !=

Using comparison Operators

parentheses can help

```
public class AnalyzeNumber {  
    public static void main(String[] args) {  
        int num = Integer.parseInt(args[0]);  
        boolean isPositive = num > 0;  
        boolean isZero = num == 0;  
        boolean isNotZero = num != 0;  
        boolean isEven = num % 2 == 0;  
    }  
}
```

Combining booleans

and and or and not

- Boolean values can be combined!!!
 - &&
 - ||
 - !

Boolean "Truth Table"

	and		or		not
	TRUE	FALSE	TRUE	FALSE	
TRUE					
FALSE					

If

- Doing things conditionally
- `if (true) {
 do this
}`