

CS 106A, Lecture 4

Introduction to Java

suggested reading:

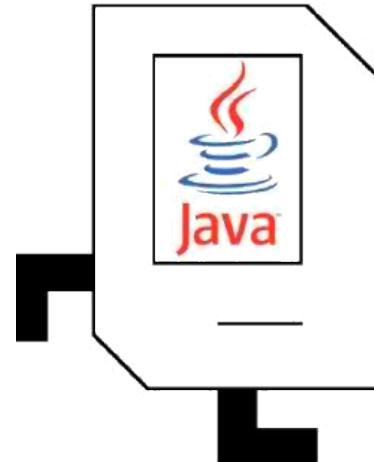
Karel, Ch. 5-6

Plan For Today

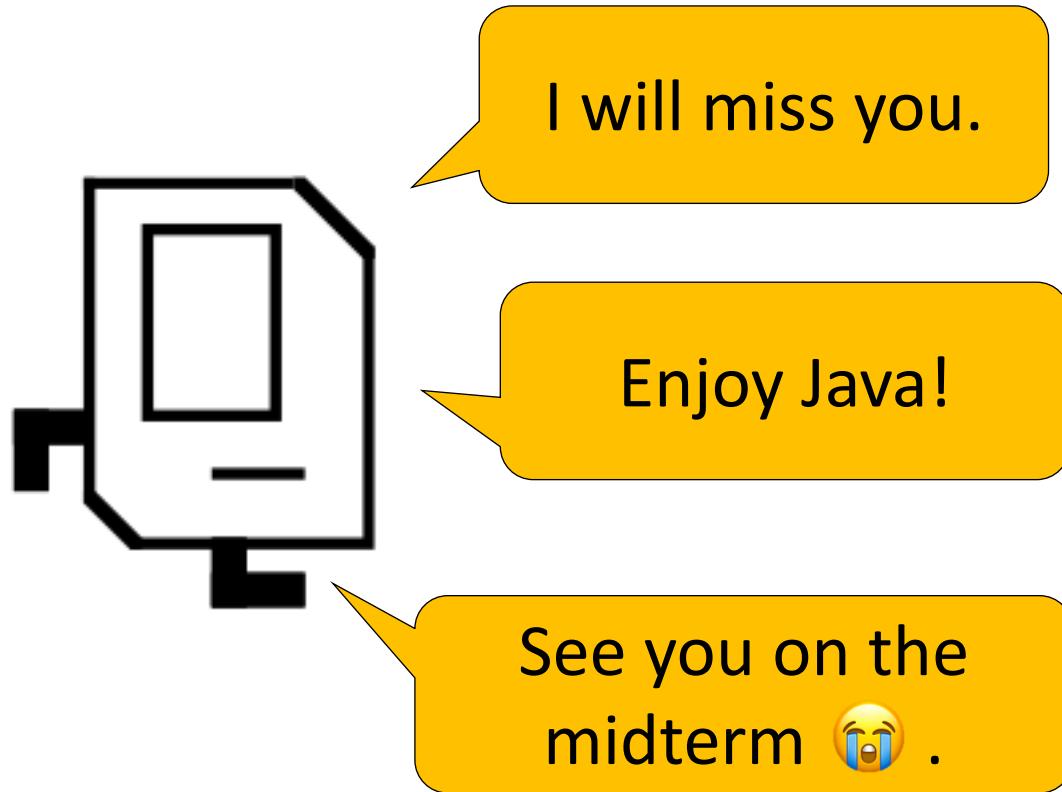
- Announcements
- Bye, Karel!
- Console Programs
- Variables
- Expressions
- Practice: Receipt

Plan For Today

- Announcements
- Bye, Karel!
- Console Programs
- Variables
- Expressions
- Practice: Receipt



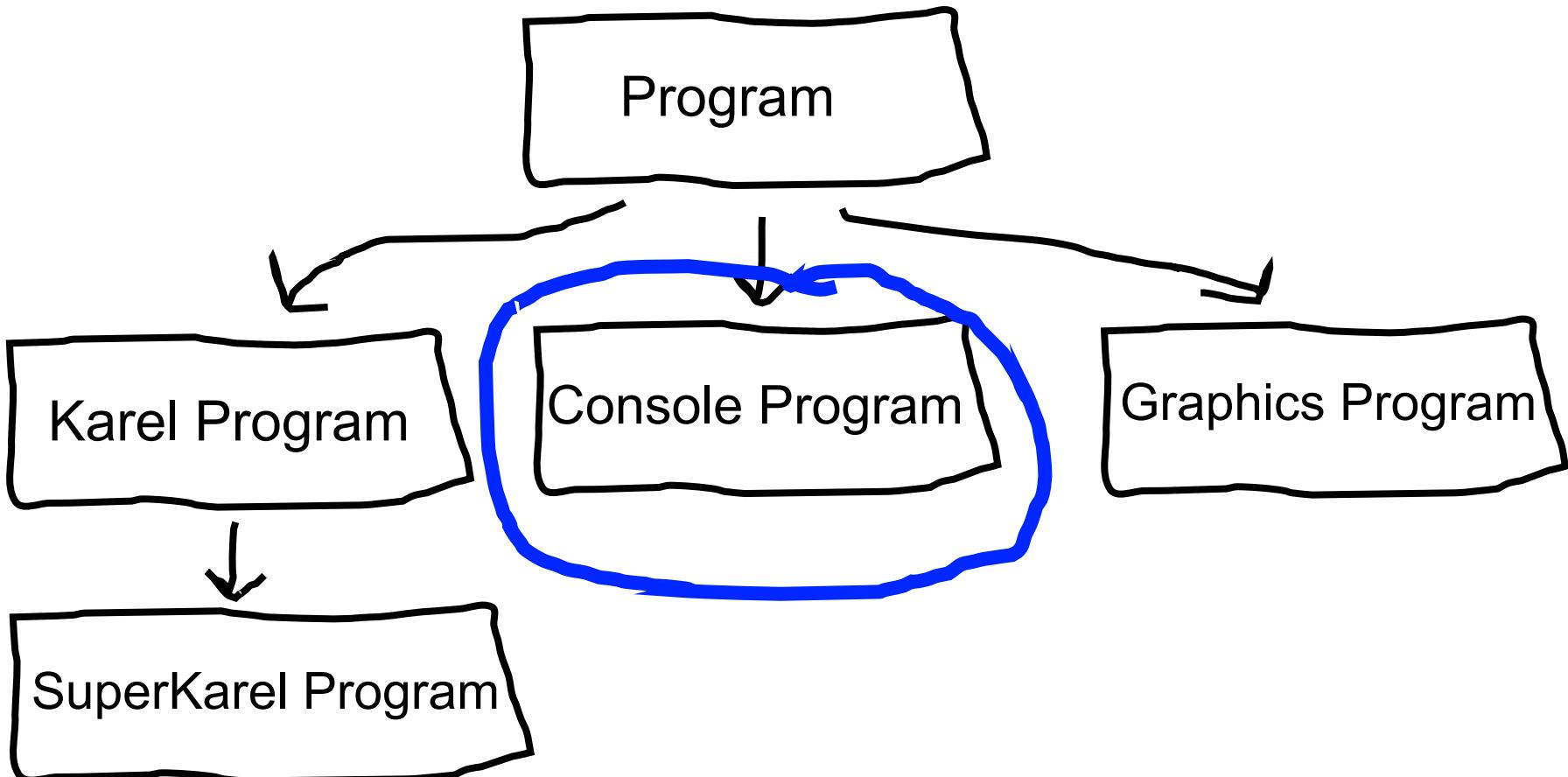
Bye, Karel!



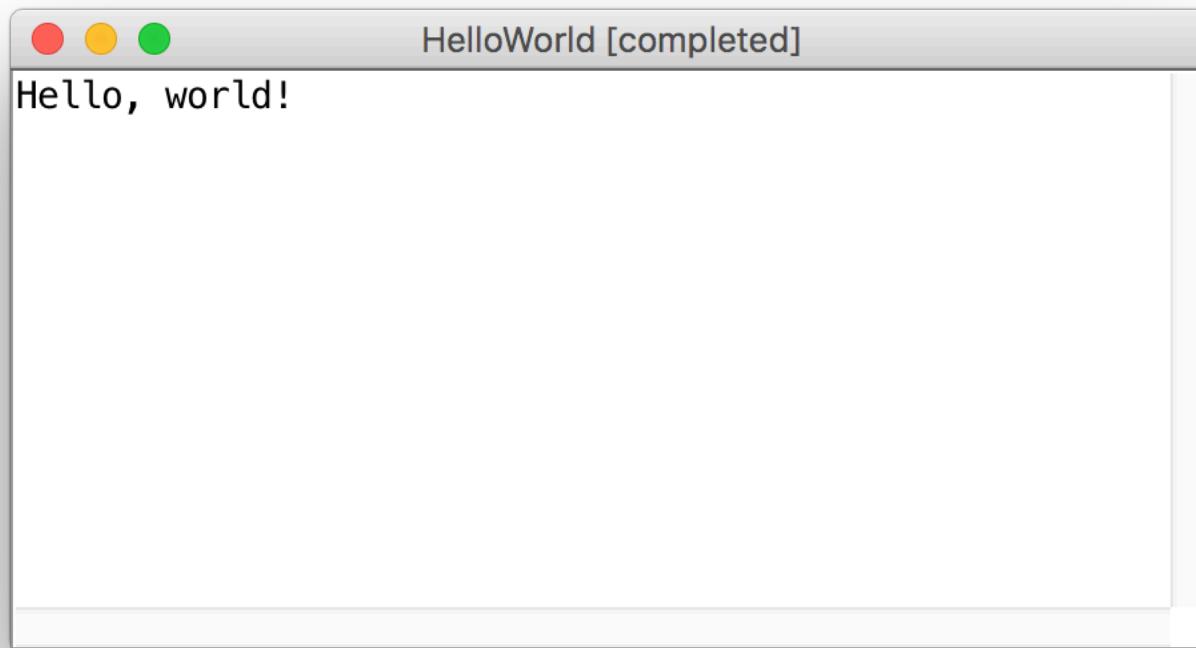
Plan For Today

- Announcements
- Bye, Karel!
- Console Programs
- Variables
- Expressions
- Practice: Receipt

Java



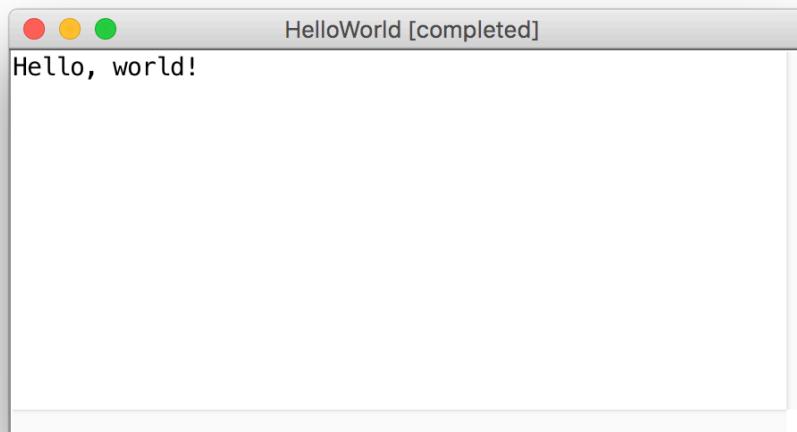
Our First Java Program



Our First Java Program

```
import acm.program.*;

public class HelloWorld extends ConsoleProgram {
    public void run() {
        println("Hello, world!");
    }
}
```



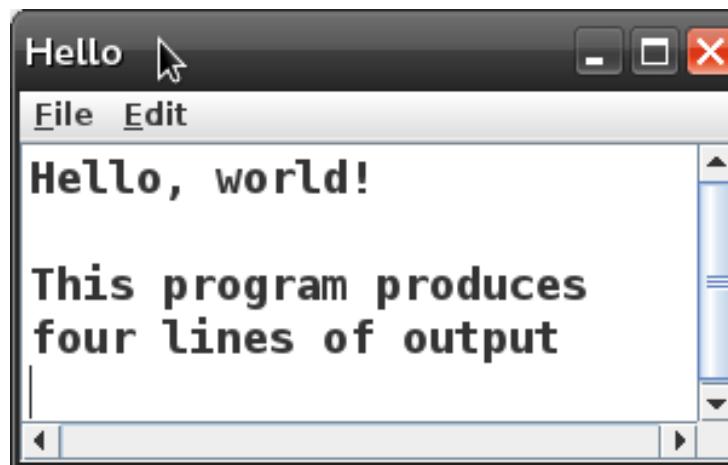
Console Programs

```
import acm.program.*;  
  
public class Name extends ConsoleProgram {  
    public void run() {  
        statements;  
    }  
}
```

- Unlike Karel, many programs produce their behavior as text.
- **console**: Text box into which the behavior is displayed.
 - *output*: Messages displayed by the program.
 - *input*: Data read by the program that the user types.

Console Programs

```
public class Hello extends ConsoleProgram {  
    public void run() {  
        println("Hello, world!");  
        println();  
        println("This program produces");  
        println("four lines of output");  
    }  
}
```

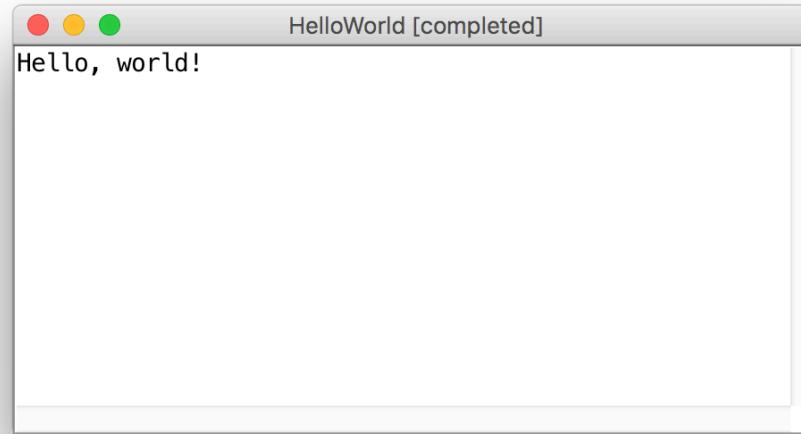


println

- A statement that prints a line of output on the console, and goes to the next line.
 - pronounced "print-linn"
- Two ways to use `println` :
 - `println("text");`
 - Prints the given message as output, and goes to the next line.
 - A message is called a *string*; it starts/ends with a " quote character.
 - The quotes do not appear in the output.
 - A string may not contain a " character.
 - `println();`
Prints a blank line of output.

print

```
public class HelloWorld extends ConsoleProgram {  
    public void run() {  
        print("Hello, ");  
        print("world!");  
    }  
}
```



Same as `println`, but does not go to the next line.

Escape sequences

- **escape sequence:** A special sequence of characters used to represent certain special characters in a string.

\t tab character

\n new line character

\" quotation mark character

\\" backslash character

- Example:

```
println("\\hello\\nhow\\tare \"you\"?\\\\\\");
```

- Output:

\hello

how are "you"?\\

Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

n1	n2	total
<input type="text"/>	<input type="text"/>	<input type="text"/>



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

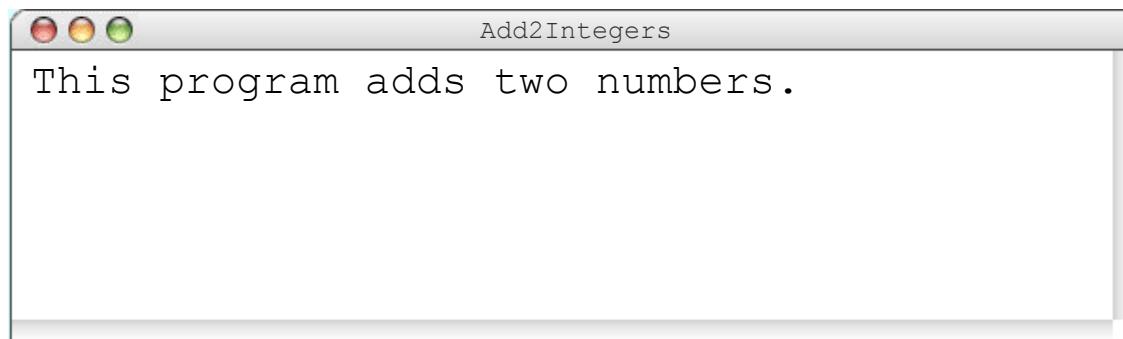
n1	n2	total
<input type="text"/>	<input type="text"/>	<input type="text"/>



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

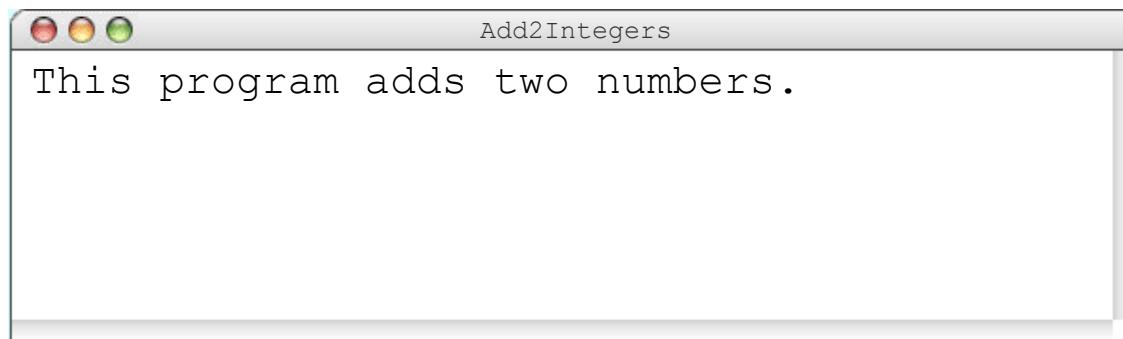
n1	n2	total
<input type="text"/>	<input type="text"/>	<input type="text"/>



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

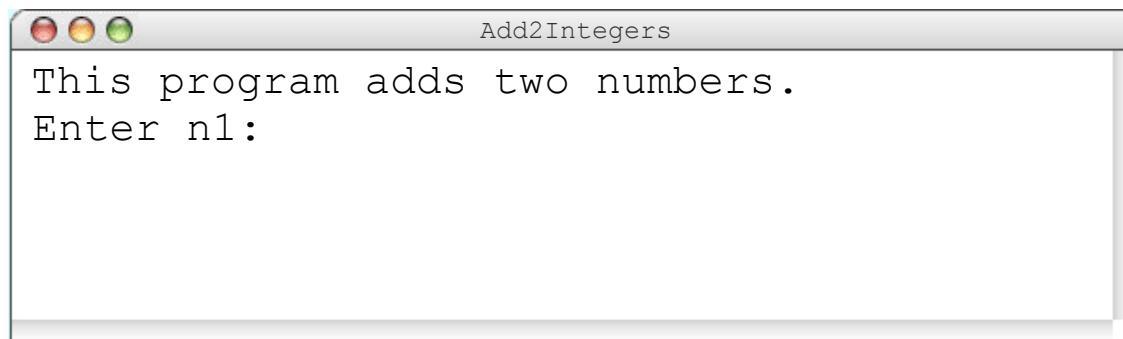
n1	n2	total
<input type="text"/>	<input type="text"/>	<input type="text"/>



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

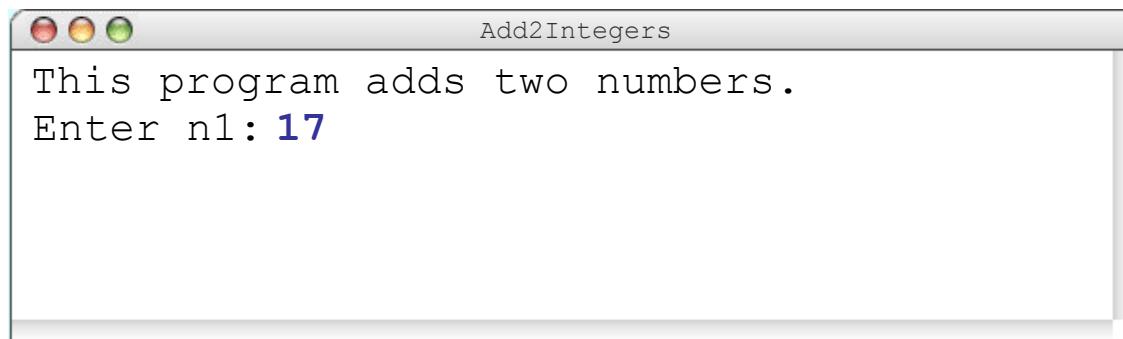
n1	n2	total
<input type="text"/>	<input type="text"/>	<input type="text"/>



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

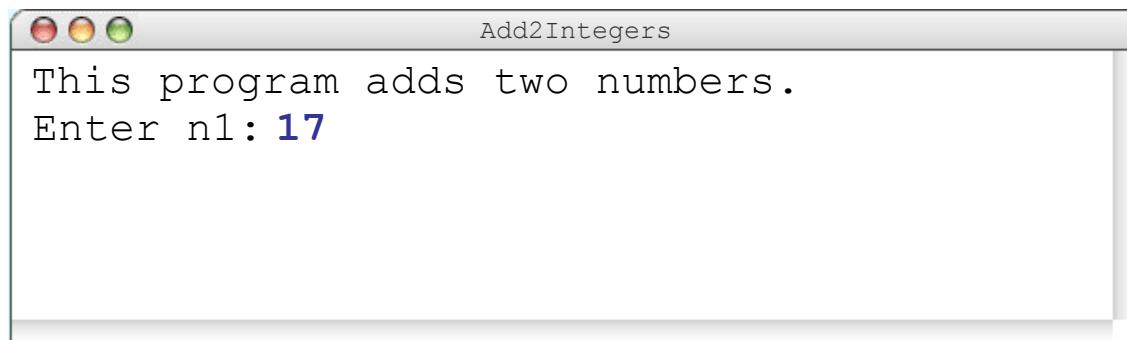
n1	n2	total
<input type="text"/>	<input type="text"/>	<input type="text"/>



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

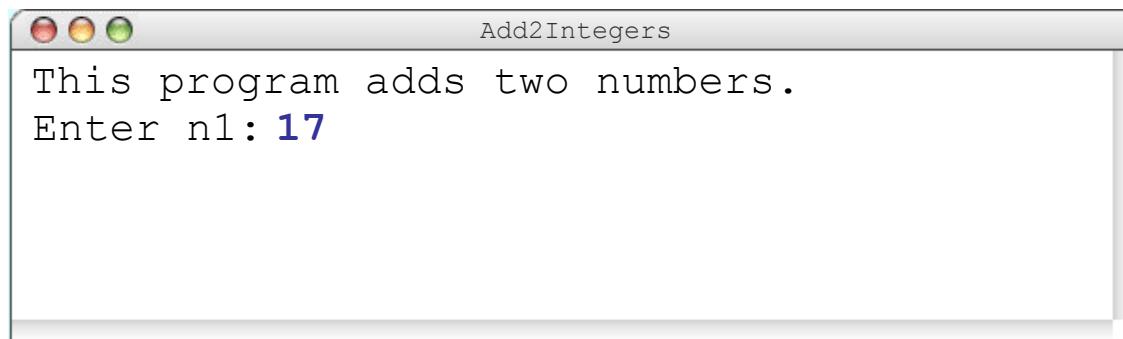
n1	n2	total
17		



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

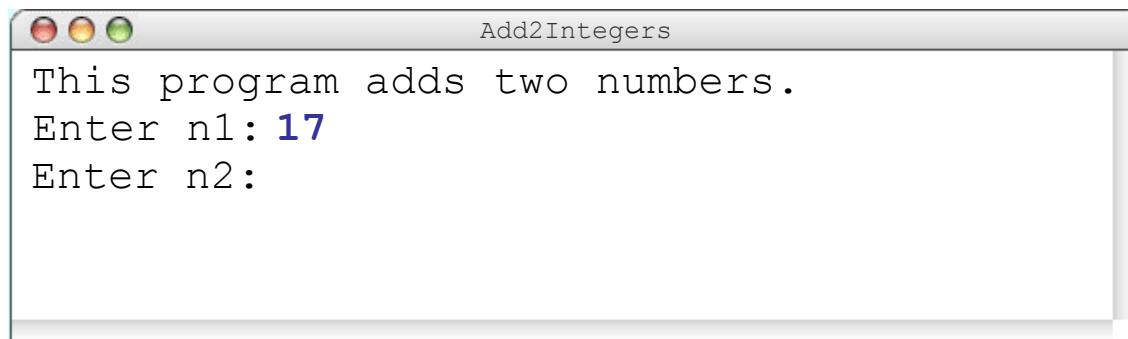
n1	n2	total
17		



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

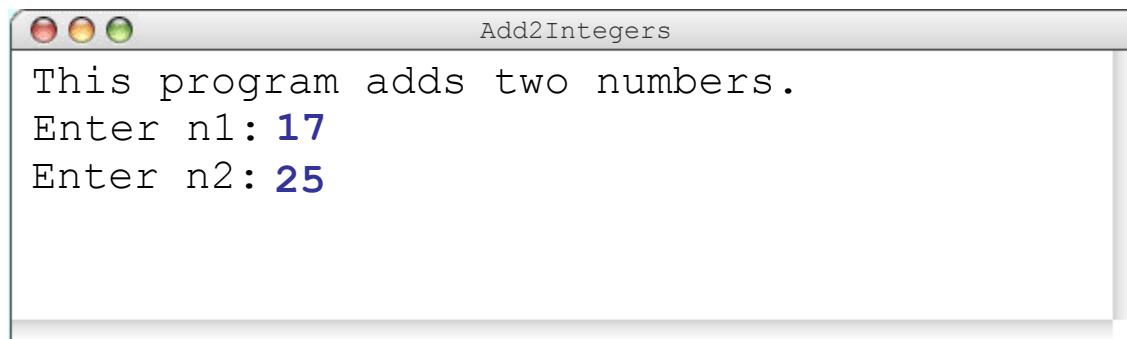
n1	n2	total
17		



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

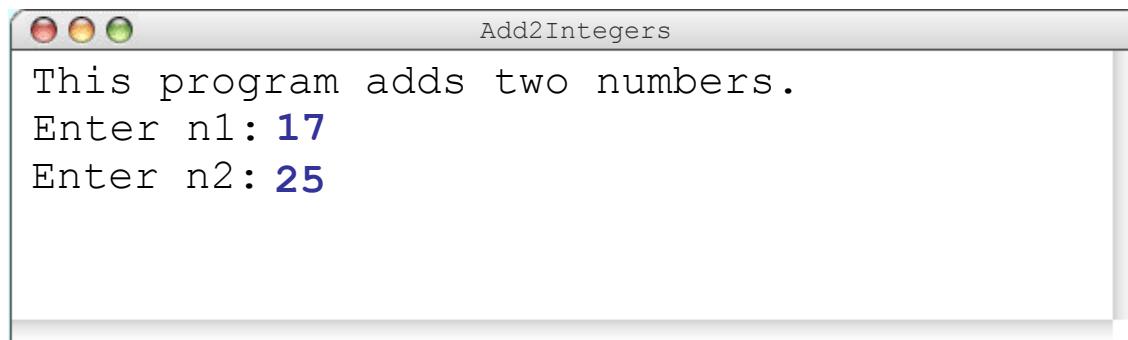
n1	n2	total
17		



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

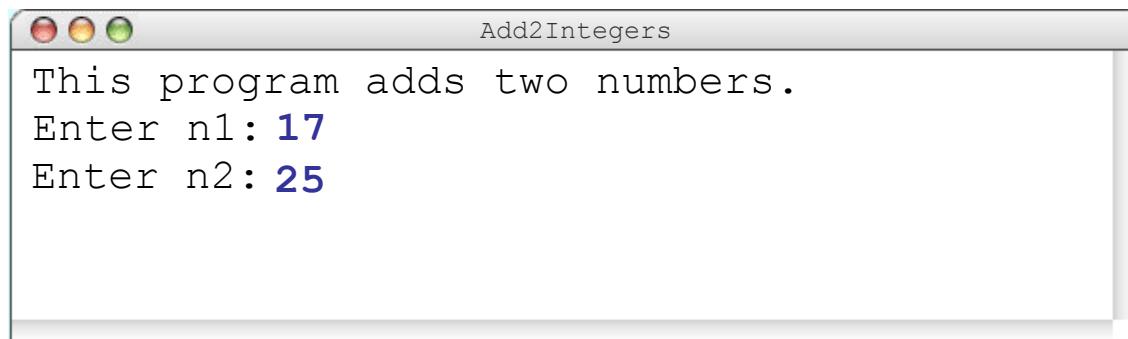
n1	n2	total
17	25	



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

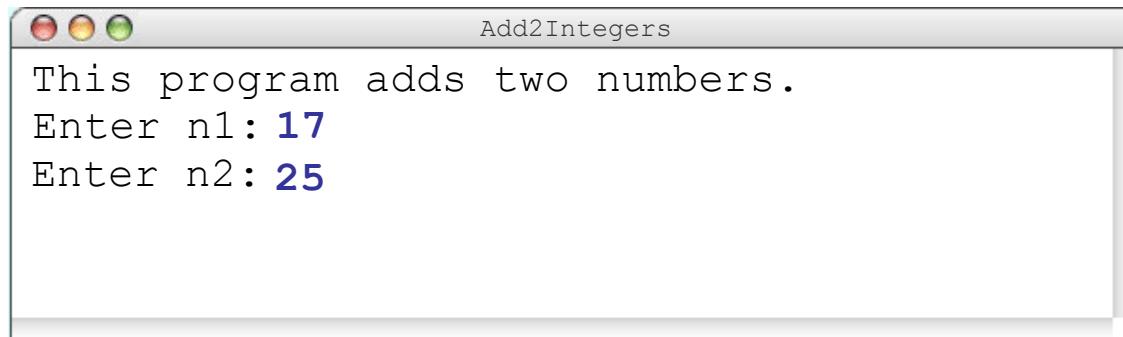
n1	n2	total
17	25	



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

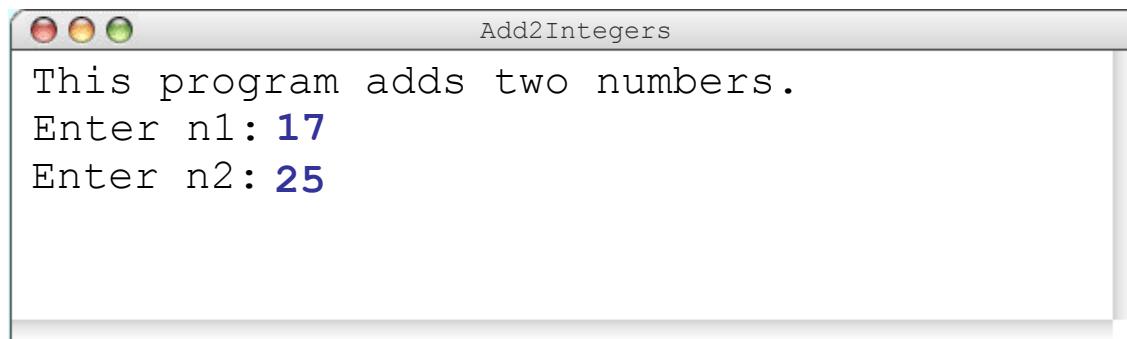
n1	n2	total
17	25	42



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

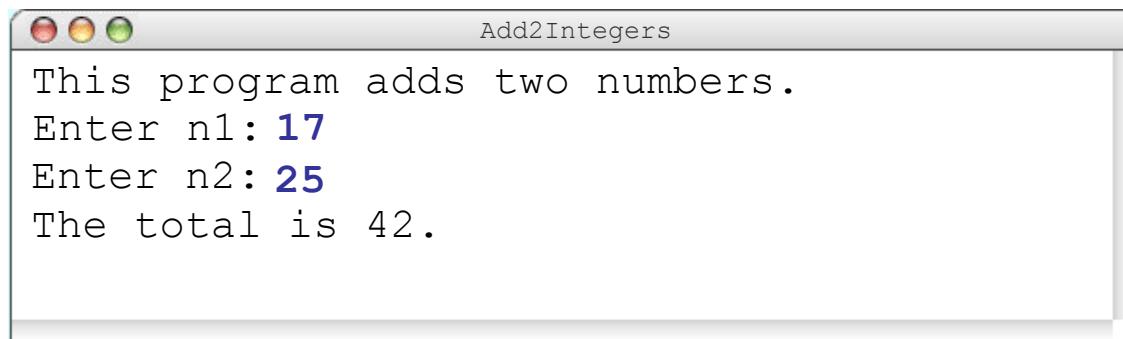
n1	n2	total
17	25	42



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

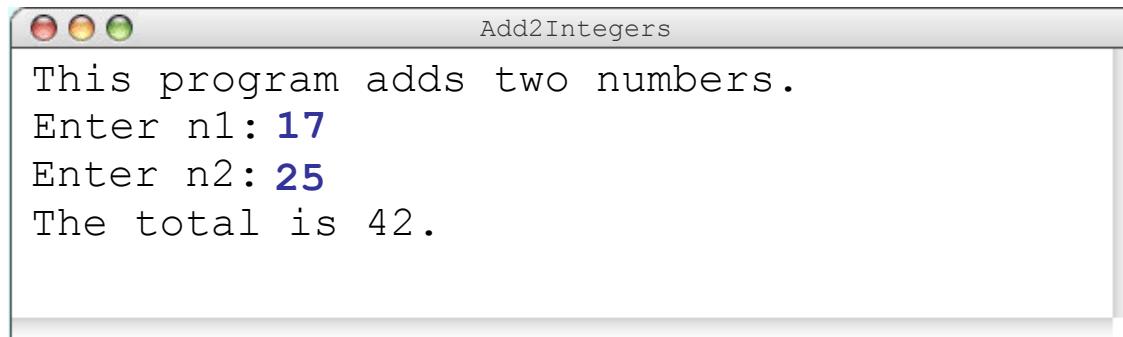
n1	n2	total
17	25	42



Add2Integers

```
public class Add2Integers extends ConsoleProgram {  
    public void run() {  
        println("This program adds two numbers.");  
        int n1 = readInt("Enter n1: ");  
        int n2 = readInt("Enter n2: ");  
        int total = n1 + n2;  
        println("The total is " + total + ".");  
    }  
}
```

n1	n2	total
17	25	42



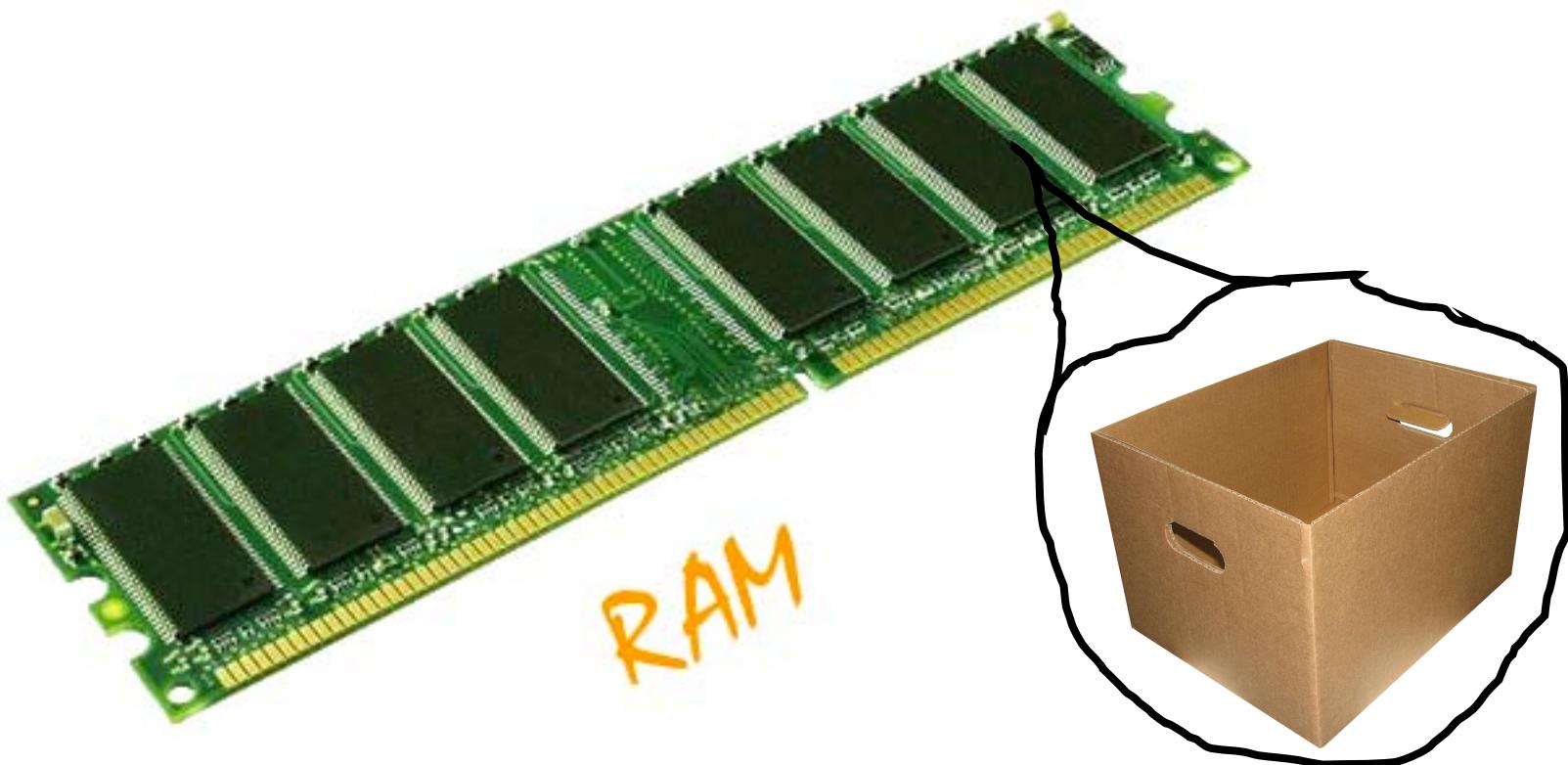
Plan For Today

- Announcements
- Bye, Karel!
- Console Programs
- **Variables**
- Expressions
- Practice: Receipt

Variables = Boxes



Variables = Boxes



My computer has space for about 2 billion boxes

Making a new Variable

```
int myVariable;
```

Making a new Variable

type



name



```
int myVariable;
```

Variable Types

int – an integer number

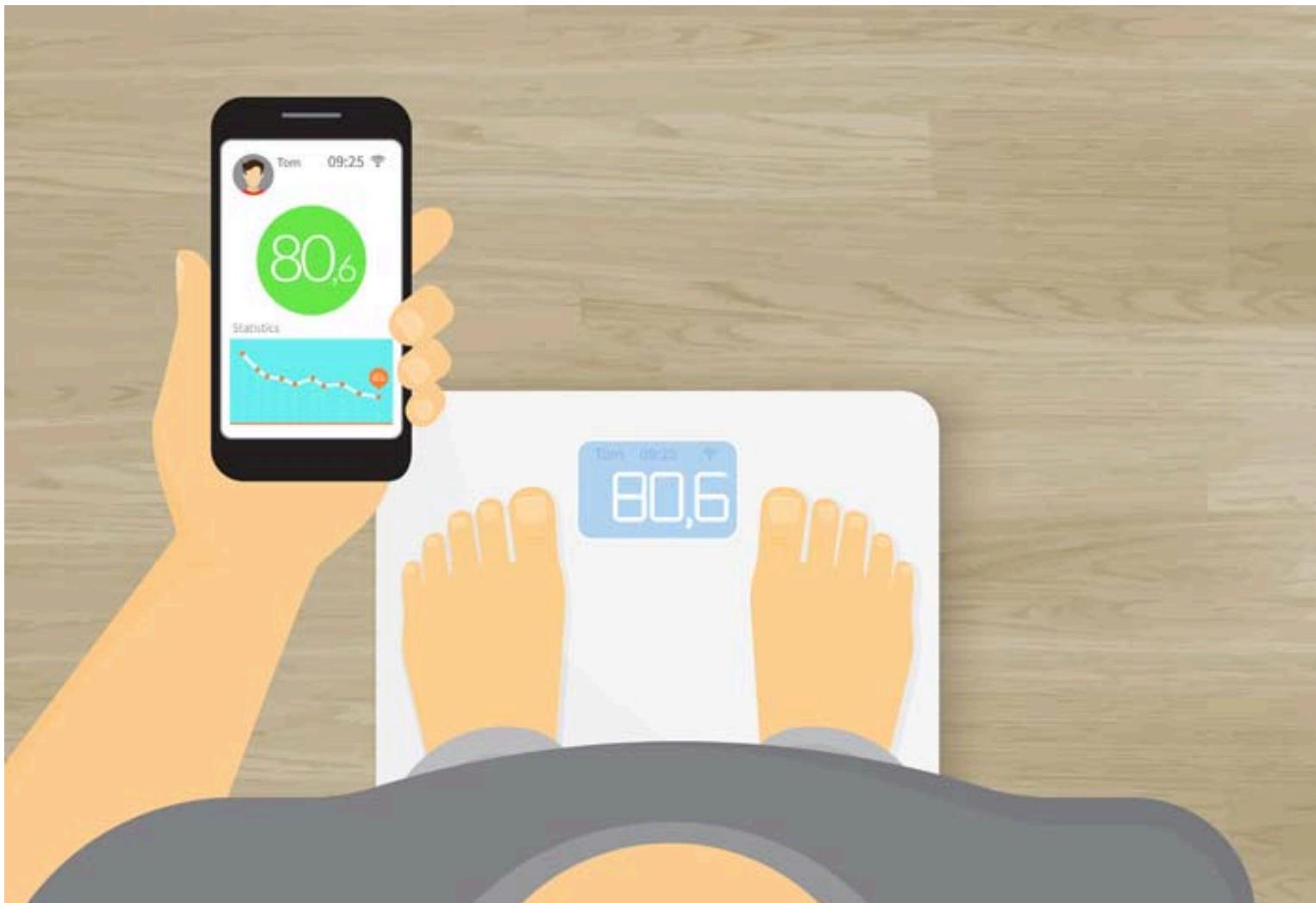
double – a decimal number

char – a single character

boolean – true or false

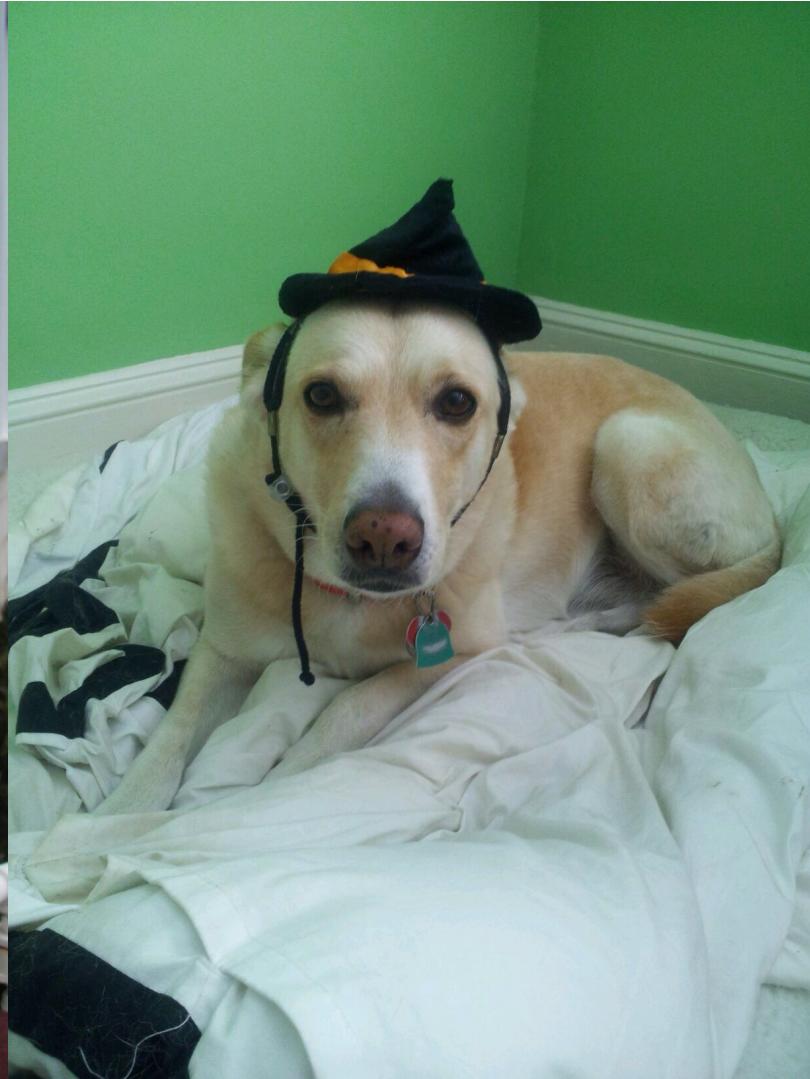
How much do I weigh?

double – answer is real number



How many pets do I have?

int – answer is an integer



double vs. int

How much vs. how many

Declaration

- **variable declaration:** Sets aside memory for storing a value.
 - Variables must be declared before they can be used.
- Syntax:

type name;

int zipcode;

zipcode



double myGPA;

myGPA



Assignment

Existing variable name



value



```
myVariable = 2;
```

Assignment

```
// integer values
```

```
int num;
```

```
num = 5;
```

```
// real values
```

```
double fraction;
```

```
fraction = 0.2;
```

```
// a single letter
```

```
char letter;
```

```
letter = 'c';
```

```
// true or false
```

```
boolean isLove;
```

```
isLove = true;
```

Assignment

- **assignment:** Stores a value into a variable.
 - The value can be an expression; the variable stores its result.
- Syntax:

name = *expression*;

```
int zipcode;  
zipcode = 90210;
```

zipcode 90210

```
double myGPA;  
myGPA = 1.0 + 2.25;
```

myGPA 3.25

Declare / initialize

- A variable can be declared-initialized in one statement.
 - This is probably the most commonly used declaration syntax.
- Syntax:

type name = expression;

double tempF = 98.6;

tempF

98.6

int x = (12 / 2) + 3;

x

9

'=' Means Assignment

- Assignment uses = , but it is not an algebraic equation.

= means, "*store the value at right in the variable at left*"

- The right side expression is evaluated first,
and then its result is stored in the variable at left.

- What happens here?

```
int x = 3;  
x = x + 2;    // ???
```

x



Assignment and types

- A variable can only store a value of its own type.

```
int x = 2.5;      // Error: incompatible types
```

- An int value can be stored in a double variable.
 - The value is converted into the equivalent real number.

```
double myGPA = 4;
```

myGPA

4.0

Compiler Errors

- A variable can't be used until it is assigned a value.

```
int x;  
println(x); // Error: x has no value
```

- You may not declare the same variable twice.

```
int x;  
int x; // ERROR: x already exists
```

```
int y = 3;  
int y = 5; // Error: y already exists
```

Using Variables

```
// Asks the user for an integer by
// displaying the given message
// and stores it in the variable 'a'
int a = readInt(message);

// Asks the user for a double by
// displaying the given message and
// stores it in the variable 'b'
double b = readDouble(message);
```

Using Variables

- Use + to print a string and a variable's value on one line.

```
double temperature = (95 + 71 + 82.7) / 3.0;  
println("The average temp. was " + temperature);
```

```
int enrolled = 11 + 17 + 4 + 19 + 14;  
println("There are " + enrolled + " students.");
```

- Output:

The average temp. was 82.9
There are 65 students.

Using Variables

- Once given a value, a variable can be used in expressions:

```
int x = 3;  
println("x is " + x);      // x is 3  
println(5 * x - 1);      // 5 * 3 - 1
```

- You can assign a value more than once:

```
int x = 3;  
println(x + " here");    // 3 here
```

x

11

```
x = 4 + 7;  
println("now x is " + x); // now x is 11
```

Plan For Today

- Announcements
- Bye, Karel!
- Console Programs
- Variables
- Expressions
- Practice: Receipt

Expressions

- You can combine literals or variables together into **expressions** using binary operators:

+	Addition	*	Multiplication
-	Subtraction	/	Division
		%	Remainder

Precedence

- **precedence:** Order in which operators are evaluated.

- Generally operators evaluate left-to-right.

- $1 - 2 - 3$ is $(1 - 2) - 3$ which is -4

- But $*$ / % have a higher level of precedence than + -

- $1 + 3 * 4$ is 13

- $6 + 8 / 2 * 3$

- $6 + 4 * 3$

- $6 + 12$ is 18

- Parentheses can alter order of evaluation, but spacing does not:

- $(1 + 3) * 4$ is 16

- $1+3 * 4-2$ is 11

Integer remainder %

- The % operator computes the remainder from integer division.

$$14 \% 4 \quad \text{is } 2$$

$$218 \% 5 \quad \text{is } 3$$

$$\begin{array}{r} 3 \\ 4) 14 \\ \underline{12} \\ 2 \end{array}$$

$$\begin{array}{r} 43 \\ 5) 218 \\ \underline{20} \\ 18 \\ \underline{15} \\ 3 \end{array}$$

- Applications of % operator:

- Obtain last digit of a number: $230857 \% 10$ is 7

- Obtain last 4 digits: $658236489 \% 10000$ is 6489

- See whether a number is odd: $7 \% 2$ is 1, but $42 \% 2$ is 0



String concatenation

- **string concatenation:** Using + between a string and another value to make a longer string.

"hello" + 42	is "hello42"
1 + "abc" + 2	is "1abc2"
"abc" + 1 + 2	is "abc12"
1 + 2 + "abc"	is "3abc"
"abc" + 9 * 3	is "abc27"
"1" + 1	is "11"
4 - 1 + "abc"	is "3abc"

- Use + to print a string and an expression's value together.

```
println("Average: " + (95.1 + 71.9) / 2);
```

Output: Average: 83.5

What does this do?

```
println(1 / 2);
```

What does this do?

```
println(1 / 2);
```

0!

Integer division

- When we divide integers, the quotient is also an integer.

$14 / 4$ is 3, not 3.5. (*Java ALWAYS rounds down.*)

$$\begin{array}{r} 3 \\ 4) 14 \\ \underline{12} \\ 2 \end{array}$$

$$\begin{array}{r} 4 \\ 10) 45 \\ \underline{40} \\ 5 \end{array}$$

$$\begin{array}{r} 52 \\ 27) 1425 \\ \underline{135} \\ 75 \\ \underline{54} \\ 21 \end{array}$$

- More examples:

– $32 / 5$ is 6

– $84 / 10$ is 8

– $156 / 100$ is 1

– Dividing by 0 causes an error when your program runs.

Type Interactions

int and **int** results in an **int**

double and **double** results in a **double**

int and **double** results in a **double**

* The general rule is: operations always return the most expressive type

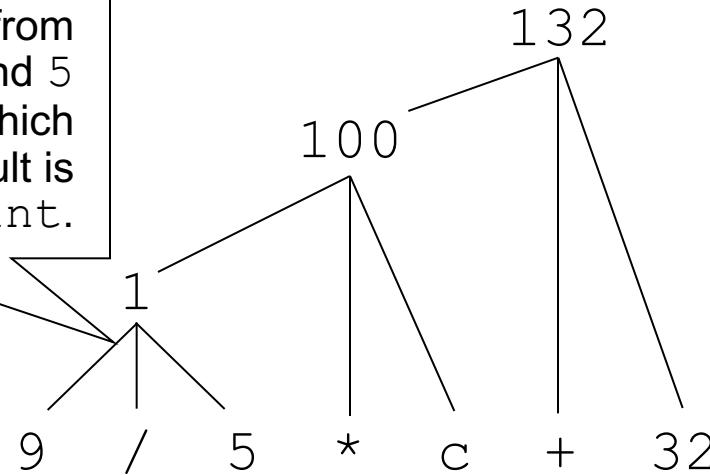
Type Interactions

Convert 100° Celsius temperature to its Fahrenheit equivalent:

```
double c = 100;  
double f = 9 / 5 * c + 32;
```

The computation consists of evaluating the following expression:

The problem arises from the fact that both 9 and 5 are of type int, which means that the result is also an int.

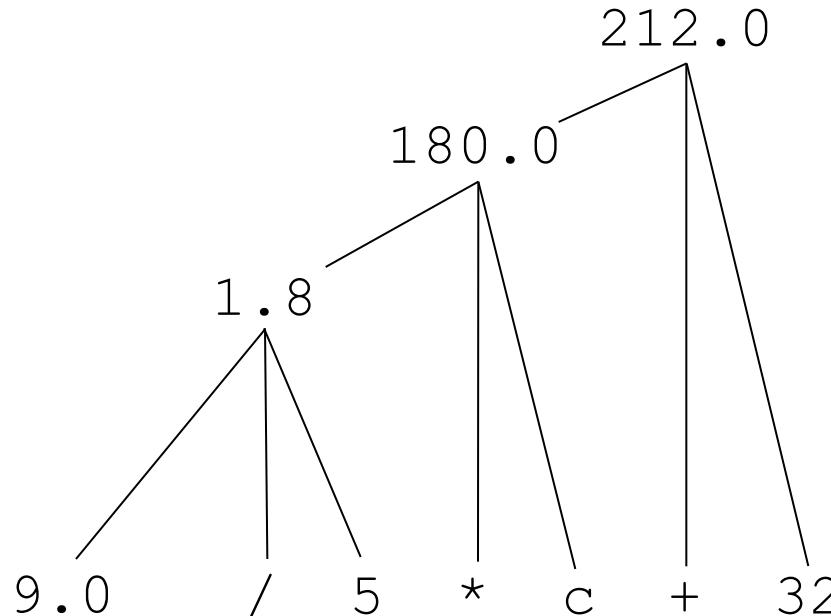


Pitfalls of Integer Division

You can fix this problem by converting the fraction to a double:

```
double c = 100;  
double f = 9.0 / 5 * c + 32;
```

The computation now looks like this:

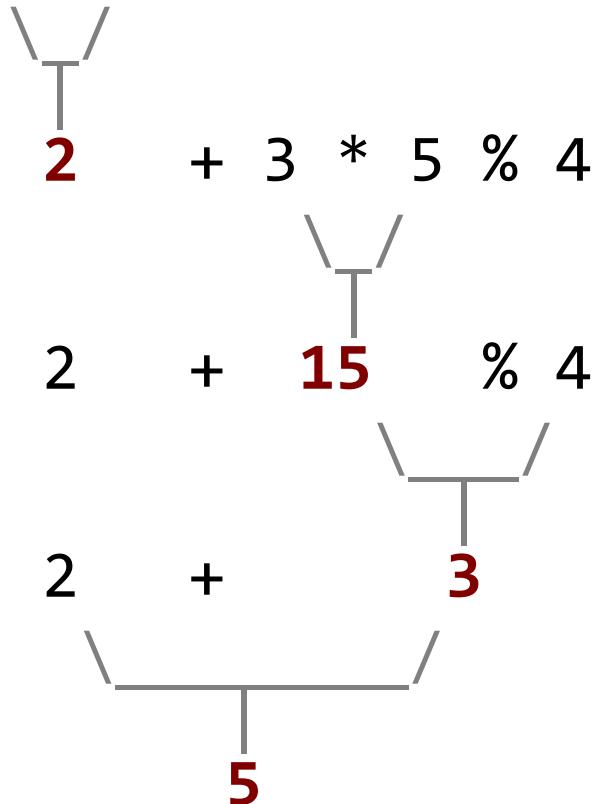


Practice

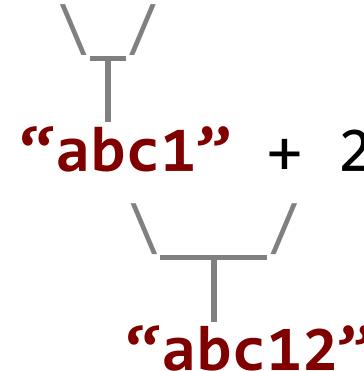
- $5 + 3 / 2 - 4$ 2
- $15 / 2.0 + 6$ 13.5
- $1 * 2 + 3 * 5 \% 4$ 11
- “abc” + 1 + 2 "abc12"
- “abc” + (1 + 2) "abc3"

Precedence examples

1 * 2 + 3 * 5 % 4



"abc" + 1 + 2

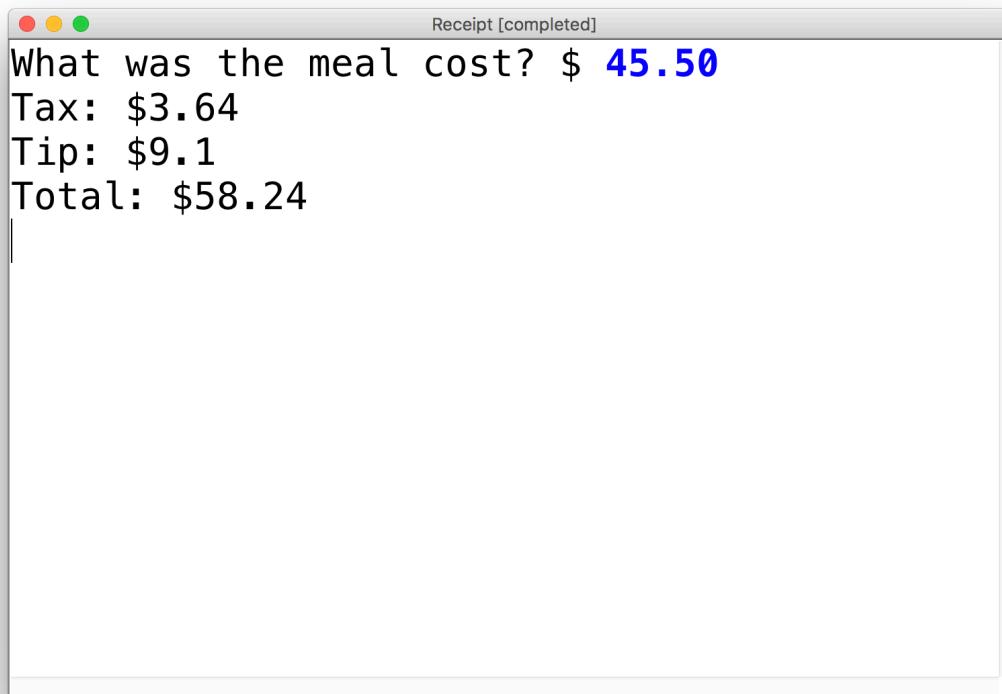


Plan For Today

- Announcements
- Bye, Karel!
- Console Programs
- Variables
- Expressions
- Practice: Receipt

Practice: Receipt Program

- Let's write a `ConsoleProgram` that calculates the tax, tip and total bill for us at a restaurant.
- The program should ask the user for the subtotal, and then calculate and print out the tax, tip and total.



Practice: Receipt Program

```
public class Receipt extends ConsoleProgram {  
    public void run() {  
        double subtotal = readDouble("Meal cost? $");  
        double tax = subtotal * 0.08;  
        double tip = subtotal * 0.20;  
        double total = subtotal + tax + tip;  
  
        println("Tax : $" + tax);  
        println("Tip: $" + tip);  
        println("Total: $" + total);  
    }  
}
```

Recap

- Announcements
- Bye, Karel!
- Console Programs
- Variables
- Expressions
- Practice: Receipt

Next time: Control flow in Java