Announcements

● If you have midterm conflicts, email both instructors ASAP and fill out the following form (by July 8th at the latest):
● Please email both instructors as soon as possible if you have academic accommodations from the OAE
● Assignment 1 out!
● “Extra” slides: treat these as required reading
Plan for Today

- Bye, Karel!
- Console programs
- Variables
- Expressions
- Practice: Receipt
Bye, Karel!

I will miss u
Bye, Karel!

I will miss u

Enjoy Java!
Bye, Karel!

I will miss u

Enjoy Java!

See you on the midterm 😊
Hello, Java!

* fun fact: this is the official Java mascot
Hello, Java!

* fun fact: this is the official Java mascot
Hello, Java!

* fun fact: this is the official Java mascot

- Hi all!
- I like Karel a latte 😍...
  but I’m cooler
- Can’t start your day without me ☕️
Plan for Today

- Bye, Karel!
- Console programs
- Variables
- Expressions
- Practice: Receipt
Types of Programs

Program
Types of Programs

- Karel Program
- Console Program
- Graphics Program
- SuperKarel Program
Types of Programs

- Program
  - Karel Program
  - Console Program
  - Graphics Program
  - SuperKarel Program
Our First Java Program

Hello, world!
Hello, World
import acm.program.*;

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

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: textbox into which the behavior is displayed
  ○ Output: messages displayed by the program
  ○ Input: data read by the program that the user types
public class HelloAgain extends ConsoleProgram {
    public void run() {
        println("Hello again!");
        println();
        println("This program produces");
        println("four lines of output");
    }
}
Console Output: println

- **println**: a statement that prints a line of *output* on the console, and goes to the next line

- Two uses:
  
  ```
  println("TEXT");  // prints the string TEXT
  println();        // prints blank line
  ```
● **I/O: Input/Output**

● `println` allows output text to the user via the console

● We can also get input from the user via the console
  ○ But before we can get input from the user, we need a way to store it...
Plan for Today

- Bye, Karel!
- Console programs
- Variables
- Expressions
- Practice: Receipt
What’s a variable?
Variables = Boxes
Variables = Boxes

* my computer has space for about 2 billion boxes
Making a New Variable

```c
int myVar = 22;
```
Making a New Variable

```c
int myVar = 22;
```

(contains an `int`)
Making a New Variable

```cpp
int myVar = 22;
```

(contains an `int`)
Making a New Variable

```java
int myVar = 22;
```

(contains an `int`)
int myVar = 22;

(type contains an int)
3 Properties

```java
int myVar = 22;
```

- **Type**: `int`
- **Name**: `myVar`
- Value: `22`

(Contains an `int`)
3 Properties

int myVar = 22;

(type, name, value)

value

(type, value)

(name)

(contains an int)
Variable Types

// an integer number
int num = 5;

// a real (decimal) number
double fraction = 0.2;

// true or false
boolean isSummer = true;
Variable Types

// a single character
c char letter = ‘c’;

// a “string” of text
String phrase = “Hi!”;
String alsoAString = “5”; // not an int!
**Double vs. Int**

**Double** - answer is a decimal

*How much* do I weigh?

**Int** - answer is an integer

*How many* pets do I have?
Making a Variable

```c
int myVar = 22;
```
Making a Variable

```c
int myVar;
myVar = 22;
```
Making a Variable

```c
int myVar;  // declare
myVar = 22;  // assign
```
Declaration & Assignment

- **Declaration**: sets aside memory for storing a value
  - Variables must be declared before they can be used
• **Declaration**: sets aside memory for storing a value
  ○ Variables must be declared before they can be used

  ```
  int myBox;
  ```
• **Declaration**: sets aside memory for storing a value
  ○ Variables must be declared before they can be used
    ```
    int myBox;
    ```

• **Assignment**: stores a value into an *existing* variable
  ○ value can be an expression; variable stores its result
• **Declaration**: sets aside memory for storing a value
  ○ Variables must be declared before they can be used
    ```
    int myBox;
    ```

• **Assignment**: stores a value into an *existing* variable
  ○ Value can be an expression; variable stores its result
    ```
    double cost; // declaration
    ```
• **Declaration**: sets aside memory for storing a value
  ○ Variables must be declared before they can be used
    ```
    int myBox;
    ```

• **Assignment**: stores a value into an *existing* variable
  ○ Value can be an expression; variable stores its result
    ```
    double cost;       // declaration
    cost = 1.5 + 0.75; // assignment
    ```
• **Declaration**: sets aside memory for storing a value
  ○ Variables must be declared before they can be used
    ```
    int myBox;
    ```

• **Assignment**: stores a value into an *existing* variable
  ○ Value can be an expression; variable stores its result
    ```
    double cost;  // declaration
    cost = 1.5 + 0.75;  // assignment
    ```

• Commonly, a variable is declared + initialized in one statement
• **Declaration**: sets aside memory for storing a value
  - Variables must be declared before they can be used
    
    ```
    int myBox;  
    ```

• **Assignment**: stores a value into an *existing* variable
  - value can be an expression; variable stores its result
    
    ```
    double cost;  // declaration
    cost = 1.5 + 0.75;  // assignment
    ```

• Commonly, a variable is declared + initialized in one statement
  
  ```
  int saraisBox = 4;  
  ```
What Happens Here?

- What happens here?

```plaintext
int x = 3;
x = x + 2;  // ?
```
What Happens Here?

- What happens here?

```java
int x = 3;
x = x + 2;  // 5
```
‘=’ Means Assignment

- What happens here?

```c
int x = 3;
x = x + 2;  // 5
```

- Assignment uses =, but it’s not an algebraic equation
  - = means: store the value at right in the variable at left
  - Right side expression evaluated first, then stored in variable at left
// creates variable called temperature with value 64.8
double temperature = (72.2 + 57.4) / 2.0;
// creates variable called temperature with value 64.8
double temperature = (72.2 + 57.4) / 2.0;

// prints value of temperature variable to the console
println(temperature);  // 64.8
// creates variable called temperature with value 64.8
double temperature = (72.2 + 57.4) / 2.0;

// prints value of temperature variable to the console
println(temperature);  // 64.8

// can use + to print string and variable’s value
println("avg temp is "+ temperature);  // avg temp is 64.8
Using Variables

● Once given a value, variables can be used in expressions

```java
int myVar = 22;
println(2 * myVar + 1);  // 45
```
Using Variables

- You can assign a value more than once
● You can assign a value more than once

```java
// declare
double temperature = (72.2 + 57.4) / 2.0;
println("temp yesterday: " + temperature); // temp yesterday: 64.8
```
Using Variables

- You can assign a value more than once

```java
// declare
double temperature = (72.2 + 57.4) / 2.0;
println("temp yesterday: " + temperature); // temp yesterday: 64.8

// reassign
temperature = 72.1;
```
Using Variables

- You can assign a value more than once

```java
// declare
double temperature = (72.2 + 57.4) / 2.0;
println("temp yesterday: " + temperature); // temp yesterday: 64.8

// reassign
temperature = 72.1;
println("temp today: " + temperature); // temp today: 72.1
```
Assignment and Types

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

```java
int x = 2.5; // Error: incompatible types
```
Assignment and Types

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

```java
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

```java
double gpa = 4;
```
Setting Variable Values

```c
int myBox = 22;
```
Setting Variable Values

```
int myBox = expression;
```
Setting Variable Values

```plaintext
int myBox = expression;
```

Example:
```plaintext
double temperature = (72.2 + 57.4) / 2.0;  // 64.8
```
Let’s now get values from user input!
Setting Variable Values

```plaintext
int myBox = user input;
```

- **Type**: int
- **Name**: myBox
- **Value**: user input
// Prompts user for a whole number. Stores result
// in a variable (aka a box).
int pets = readInt("How many pets? ");

// Prompts user for a decimal number. Stores result
// in a variable (aka a box).
double tip = readDouble("Tip? $ ");

// Prompts user for a boolean & stores result in variable.
boolean isSunny = readBoolean("Sun shining? ");
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 + ".");
    }
}
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 + ".");
    }
}
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 + ".");
    }
}

This program adds two numbers.
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 + ".");
    }
}
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 + ".");
    }
}
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 + ".");
    }
}
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 + ".");
    }
}
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 + ".");
    }
}
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 + ".");
    }
}
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 + ".");
    }
}

This program adds two numbers.
Enter n1: 17
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 + ".");
    }
}

This program adds two numbers.
Enter n1: 17
Enter n2:
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 + ".");
    }
}

This program adds two numbers.
Enter n1: 17
Enter n2: 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 + ".");
    }
}
```

```
17 25
```

This program adds two numbers.
Enter n1: 17
Enter n2: 25
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 + ".");
    }
}

Add2Integers

This program adds two numbers.
Enter n1: 17
Enter n2: 25
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 + ".");
    }
}

This program adds two numbers.
Enter n1: 17
Enter n2: 25
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 + ".");
    }
}
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 + ".");
    }
}
```java
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 + ".");
    }
}
```

This program adds two numbers.
Enter n1: 17
Enter n2: 25
The total is 42.
1. A variable can’t be used until it is assigned a value

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

2. You may not declare the same variable twice

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

3. You may not use a variable until it is declared

```java
z = 10; // Error: z cannot be resolved
```
Plan for Today

- Bye, Karel!
- Console programs
- Variables
- Expressions
- Practice: Receipt
• You can combine literals or variables together into *expressions* using *binary operators*:

  - *Addition*  
  - *Subtraction*  
  - *Multiplication*  
  - *Division*  
  - *Remainder*
int result = 4 + 2 * 3;

<table>
<thead>
<tr>
<th>Priority</th>
<th>Operator</th>
<th>Tie breaker</th>
</tr>
</thead>
<tbody>
<tr>
<td>Highest</td>
<td>()</td>
<td>Left to right</td>
</tr>
<tr>
<td>Middle</td>
<td>* / %</td>
<td>Left to right</td>
</tr>
<tr>
<td>Lowest</td>
<td>+ -</td>
<td>Left to right</td>
</tr>
</tbody>
</table>
// Multiplication before addition
int result = 4 + 2 * 3;  // 10

<table>
<thead>
<tr>
<th>Priority</th>
<th>Operator</th>
<th>Tie breaker</th>
</tr>
</thead>
<tbody>
<tr>
<td>Highest</td>
<td>()</td>
<td>Left to right</td>
</tr>
<tr>
<td>Middle</td>
<td>* / %</td>
<td>Left to right</td>
</tr>
<tr>
<td>Lowest</td>
<td>+ -</td>
<td>Left to right</td>
</tr>
</tbody>
</table>
Order of Operations

// Multiplication before addition
int result = 4 + 2 * 3;    // 10

// PARENS first, then left to right
int tot = 1 + 2 + (3 * 4); // 15

<table>
<thead>
<tr>
<th>Priority</th>
<th>Operator</th>
<th>Tie breaker</th>
</tr>
</thead>
<tbody>
<tr>
<td>Highest</td>
<td>()</td>
<td>Left to right</td>
</tr>
<tr>
<td>Middle</td>
<td>* / %</td>
<td>Left to right</td>
</tr>
<tr>
<td>Lowest</td>
<td>+ -</td>
<td>Left to right</td>
</tr>
</tbody>
</table>
Expressions

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

  + Addition
  - Subtraction
  * Multiplication
  / Division
  % Remainder

What’s THAT? %%%
The % operator computes the remainder from integer division.

14 % 4 is 2

\[
\begin{array}{c}
  3 \\
  4 ) 14 \\
  12 \\
  \underline{2}
\end{array}
\]

218 % 5 is 3

\[
\begin{array}{c}
  43 \\
  5 ) 218 \\
  20 \\
  18 \\
  \underline{15} \\
  \underline{3}
\end{array}
\]
The % operator computes the remainder from integer division

\[ 14 \% 4 \text{ is } 2 \]
\[ 218 \% 5 \text{ is } 3 \]

Applications of % operator:
- Obtain last digit of a number: \[ 857 \% 10 \text{ is } 7 \]
- Obtain last 3 digits: \[ 26489 \% 1000 \text{ is } 489 \]
- See if a number is odd or even: \[ 7 \% 2 \text{ is } 1, \text{ but } 42 \% 2 \text{ is } 0 \]
What do you think this does?

double successRate = 1 / 2;
double successRate = 1 / 2;
Integer Division

- When we divide integers, the quotient is also an integer
  - 14 / 4 is 3, not 3.5 (Java always rounds down)
Integer Division

- When we divide integers, the quotient is also an integer
  - 14 / 4 is 3, not 3.5 (Java always rounds down)

```java
double successRate = 1 / 2;
```
When we divide integers, the quotient is also an integer

- 14 / 4 is 3, not 3.5 (Java always rounds down)

\[
\text{double } \text{successRate} = \frac{\text{int}}{\text{int}}; \\
\text{int} \quad \text{int}
\]
Integer Division

- When we divide integers, the quotient is also an integer
  - 14 / 4 is 3, not 3.5 (Java always rounds down)

\[
\text{double } \text{successRate} = \frac{1}{2};
\]
Integer Division

- When we divide integers, the quotient is also an integer
  - 14 / 4 is 3, not 3.5 (Java always rounds down)

```java
double successRate = 1 / 2;
```

1 / 2 → would be 0.5 → truncated to 0
14 / 4 → would be 3.5 → truncated to 3
199 / 100 → would be 1.99 → truncated to 1
Type Interactions

int and int returns an int  
7 / 2 → 3

int and double returns a double  
7 / 2.0 → 3.5

double and double returns a double  
4.4 * 0.5 → 2.2

* operations return the most expressive type

String > double > int > char > boolean
Pitfalls of Integer Division

Convert 100° Celsius temperature to its Fahrenheit equivalent:

```java
double c = 100;
double f = 9 / 5 * c + 32;
```
Pitfalls of Integer Division

Convert 100° Celsius temperature to its Fahrenheit equivalent:
  
  double c = 100;
  double f = 9 / 5 * c + 32;
Pitfalls of Integer Division

Convert 100° Celsius temperature to its Fahrenheit equivalent:

double c = 100;
double f = \frac{9}{5} * c + 32;
Pitfalls of Integer Division

Convert 100°C Celsius temperature to its Fahrenheit equivalent:

```java
double c = 100;
double f = 9 / 5 * c + 32;  // 132
```

102
Pitfalls of Integer Division

Convert 100° Celsius temperature to its Fahrenheit equivalent:

double c = 100;
double f = 9 / 5 * c + 32; // 132

How can we fix it?

double c = 100;
double f = 9.0 / 5 * c + 32;
Pitfalls of Integer Division

Convert 100° Celsius temperature to its Fahrenheit equivalent:

```java
double c = 100;
double f = 9 / 5 * c + 32; // 132
```

How can we fix it?

```java
double c = 100;
double f = 9.0 / 5 * c + 32; // 212.0
```
Pitfalls of Integer Division

Convert 100° Celsius temperature to its Fahrenheit equivalent:

\[
\text{double } c = 100;
\text{double } f = \frac{9}{5} \times c + 32; \quad // \quad 132
\]

How can we fix it?

\[
\text{double } c = 100;
\text{double } f = \frac{9.0}{5} \times c + 32; \quad // \quad 212.0
\]

* You can fix this problem by converting the fraction to a double, either by inserting decimal points or by using a type cast
Practice

- $5 + 3 / 2 - 4$
- $15 / 2.0 + 6$
- $1 * 2 + 3 * 5 \% 4$
- "abc" + 1 + 2
- "abc" + (1 + 2)
Practice

- $5 + 3 / 2 - 4 = 2$
- $15 / 2.0 + 6$
- $1 * 2 + 3 * 5 \% 4$
- "abc" + 1 + 2
- "abc" + (1 + 2)
Practice

- $5 + 3 / 2 - 4 = 2$
- $15 / 2.0 + 6 = 13.5$
- $1 * 2 + 3 * 5 \% 4$
- “abc” + 1 + 2
- “abc” + (1 + 2)
Practice

- $5 + 3 / 2 - 4 = 2$
- $15 / 2.0 + 6 = 13.5$
- $1 * 2 + 3 * 5 \% 4 = 5$
- "abc" + 1 + 2 = "abc12"
- "abc" + (1 + 2) = "abc3"
Plan for Today

- Bye, Karel!
- Console programs
- Variables
- Expressions
- Practice: Receipt
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.
What's the pseudocode?

What was the meal cost? $45.50
Tax: $3.64
Tip: $9.1
Total: $58.24
Ask user for meal cost (store in variable)
Calculate tax (make variable!)
Calculate tip (make variable)
Calculate total (make variable)
Print out tax, tip, and total
Let’s Code It!
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: ");
        println("Tip: ");
        println("Total: ");
    }
}

Recap

- Bye, Karel!
- Console programs
- Variables
- Expressions
- Practice: Receipt

Next time: Control flow in Java
• 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
[Extra] 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
[Extra] Practice

- $5 + 3 / 2 - 4$
- $15 / 2.0 + 6$
- $1 * 2 + 3 * 5 \% 4$
- "abc" + 1 + 2
- "abc" + (1 + 2)
[Extra] Practice

- $5 + 3 / 2 - 4 \quad 2$
- $15 / 2.0 + 6 \quad 13.5$
- $1 * 2 + 3 * 5 \mod 4 \quad 5$
- "abc" + 1 + 2 \quad "abc12"
- "abc" + (1 + 2) \quad "abc3"
[Extra] Precedence Examples

1 * 2 + 3 * 5 \% 4

“abc” + 1 + 2
[Extra] Precedence Examples

1 * 2 + 3 * 5 % 4

“abc” + 1 + 2

122
[Extra] Precedence Examples

1 * 2 + 3 * 5 % 4

"abc" + 1 + 2

"abc1" + 2

"abc12"
Same as println, but does not go to the next line.
• **escape sequence**: A special sequence of characters used to represent certain special characters in a string.

\t \n " \n backslash character

• Example:

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

• Output:

```
\hello
how       are "you"?\n```