Topics for today

- Learn how to harness computing power to solve problems.
- To that end:
  - Introduce Java programming language
    - Shared structure with Karel
    - Console programs and printing
    - Variables
    - Data types
    - Assignment
Welcome to Java!
The Java programming language

- Depending on metric, between #1 and top 5 language
  - Android programming
  - Web servers
  - Desktop apps
  - ...

- Syntactically, a close relative to C++ (CS106B) and C (CS107)
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.
The console

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

- Its output:

  Hello, world!

  This program produces
  four lines of output

- **console**: Text box into which the output is printed.
The ConsoleProgram contains these useful methods:

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pause(ms);</td>
<td>halts for the given # of milliseconds</td>
</tr>
<tr>
<td>setFont(&quot;font&quot;);</td>
<td>changes the text size/shape</td>
</tr>
<tr>
<td>setSize(w, h);</td>
<td>sets the console window's onscreen size</td>
</tr>
<tr>
<td>setTitle(&quot;text&quot;);</td>
<td>sets the title bar text</td>
</tr>
</tbody>
</table>

```java
public class Hello extends ConsoleProgram {
    public void run() {
       setFont("Comic Sans MS-Bold-16");
       setSize(500, 300);
       println("Hello, world!");
        ...
    }
}
```
The console

- The name “console” comes from historical computer systems that would take up an entire room, and have one “console” station where the operator would interact with and control the system.
The println statement

- A statement that prints a line of output on the console.
  - pronounced "print-linn"

- Two ways to use println:
  - println("text");
  - Prints the given message as output.
    - 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.

* unless you use a special "escape sequence"
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:
  ```java
  println("\t\nhello\nhow\ntare
""you""?\\\n\n");
  ```

- Output:
  ```java
  hello
  how are "you"?
  ```
Variables
Variables

- **variable**: A piece of the computer's memory that is given a name and type, and can store a value.

  - **THREE COMPONENTS OF A VARIABLE:**
    - Name
    - Type
    - Value

- Three steps for using a variable:
  - *Declare it* - introduce its name and type
  - *Initialize it* - store a value into it
  - *Use it* - print it or use it as part of an expression
Variable declaration: Sets aside memory for storing a value.

- Variables must be declared before they can be used.
- Gives the NAME and the TYPE

Syntax:

```plaintext
type name;
```

```plaintext
int zipcode;
```

```plaintext
double myGPA;
```
Data types

- **type**: A category or set of data values.
  - Constrains the operations that can be performed on data
  - Many languages ask the programmer to specify types
  - Examples: integer, real number, string

- **Note**: Internally, computers store everything as 1s and 0s
  - 104 → 01101000
  - "hi" → 0110100001101001
  - Java gives data types to help humans interpret the values
Java's primitive types

- **primitive types**: some simple types for numbers, text, etc.
  - Java also has **object types**, which we'll talk about later

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>integers (up to $2^{31} - 1$)</td>
<td>42, -3, 0, 926394</td>
</tr>
<tr>
<td>double</td>
<td>real numbers (up to $10^{308}$)</td>
<td>3.1, -0.25, 9.4e3</td>
</tr>
</tbody>
</table>
| char      | single text characters | 'a', 'X', '?', '
'             |
| boolean   | logical values         | true, false              |

- Why does Java distinguish integers vs. real numbers?
Assignment

- **Assignment**: Stores a value into a variable.
  - Gives the variable its **VALUE**
  - The value can be an expression; the variable stores its result.

- Syntax:

  \[
  name = expression;
  \]

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

```
double myGPA;
myGPA = 1.0 + 2.25;
```
Declare + initialize in one

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

- Syntax:

  \[
  \text{type } \text{name} = \text{expression};
  \]

  ```
  double tempF = 98.6;
  int x = (11 / 2) + 3;
  ```
Using variables

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

```java
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:

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

x = 4 + 7;
println("now x is " + x); // now x is 11
```
More on Assignment
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;  // ???
  ```

  (A) x
  (B) x
  (C) x
  (D) Other or causes an error
Assignment

- Assignment uses the `=` symbol that you’ve seen in math class, but it is not an algebraic equation in the sense that you’ve seen in math class.

  `=` 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?

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