Variables and Types
Java in one slide

Up Next:

<table>
<thead>
<tr>
<th>Built-In Types</th>
<th>Punctuation</th>
<th>Numeric Operations</th>
<th>String Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>{ ( )</td>
<td>+ - * / % ++</td>
<td>+ &quot; &quot;</td>
</tr>
<tr>
<td>double</td>
<td>( )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>, ;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>char</td>
<td></td>
<td>(int) x (double) x</td>
<td></td>
</tr>
<tr>
<td>boolean</td>
<td>=</td>
<td>(char) x</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Integer.parseInt()</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Double.parseDouble()</td>
<td></td>
</tr>
</tbody>
</table>

Then:

<table>
<thead>
<tr>
<th>Math Library</th>
<th>Boolean Operations</th>
<th>Flow Control</th>
<th>Arrays</th>
</tr>
</thead>
<tbody>
<tr>
<td>Math.sin()</td>
<td>true</td>
<td>if</td>
<td>arr[i]</td>
</tr>
<tr>
<td>Math.cos()</td>
<td>false</td>
<td>else</td>
<td>new</td>
</tr>
<tr>
<td>Math.log()</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Math.exp()</td>
<td>&amp;&amp;</td>
<td>for</td>
<td></td>
</tr>
<tr>
<td>Math.sqrt()</td>
<td>!</td>
<td>while</td>
<td></td>
</tr>
<tr>
<td>Math.min()</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Math.max()</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Math.abs()</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Math.PI</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

After Fall break:

<table>
<thead>
<tr>
<th>Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>class</td>
</tr>
<tr>
<td>public</td>
</tr>
<tr>
<td>new</td>
</tr>
<tr>
<td>this</td>
</tr>
</tbody>
</table>

Section 1.2
Variables and Types

```c
int a, b;

a = 1234;

b = 99;

int c = a + b;
```

- **Declaration statement**: `int a, b;` defines `a` and `b` as `int` variables.
- **Assignment statement**: `a = 1234;` assigns the value 1234 to `a`.
- **Literal**: The value `1234` is a literal.
- **Combined declaration and assignment statement**: `int c = a + b;` declares `c` as an `int` and assigns the sum of `a` and `b` to `c`.

Section 1.2
Variables and Types

Section 1.2

```
int a, b;
a = 1234;
b = 99;
int c = a + b;
```
Variables and Types

Section 1.2

Declaration statement

```
int a, b;
```

Literal

```
a = 1234;  
```

Assignment statement

```
b = 99;    
```

Combined declaration and assignment statement

```
int c = a + b;
```
Variables and Types

Section 1.2

```c
int a, b;
a = 1234;
b = 99;
int c = a + b;
```
Variables and Types

Section 1.2

```
int a, b;
int c = a + b;
```

- **Declaration statement**
- **Variable name**
- **Literal**
- **Assignment statement**
- **Combined declaration and assignment statement**

1234

a

99

b

1333

c
Variables and Types

```
int a, b;
a = 1234;
b = 99;
int c = a + b;
```

"int" means the variable will always hold an integer

Section 1.2
Assignment

Test with "pseudo-java"

Section 1.2
Assignment

Test with "pseudo-java"

Section 1.2
Assignment

Test with "pseudo-java"

Section 1.2
Assignment

Test with "pseudo-java"

Welcome to DrJava. Working directory is /Users/pjbrown/introcs
> int a, b;
> a = 1234
1234
> b = 99
 99
> a = b
> b = t
> 1234
> 
Resetting Interactions 1:0

1234 a

99 b

1234 t

Section 1.2
Assignment

Test with "pseudo-java"

Section 1.2
Assignment

Test with "pseudo-java"

Welcome to DrJava. Working directory is /Users/pjbrown/introcs
> int a, b;
> a = 1234
1234
> b = 99
99
> int t = a
> t
1234
> a = b
99
> b = t

Section 1.2
Assignment

= stores a value in a variable: not like math!

Welcome to DrJava. Working directory is /Users/pjbrown/introcs
> int a, b;
> a = 1234
1234
> b = 99
99
> int t = a
> t
1234
> a = b
99
> b = t
1234
>
Section 1.2
# int: Integers (whole numbers)

+, −, *, /, % (modulo), (), `Integer.parseInt()`

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result?</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 + 3</td>
<td>8</td>
</tr>
<tr>
<td>5 − 3</td>
<td>2</td>
</tr>
<tr>
<td>5 * 3</td>
<td>15</td>
</tr>
<tr>
<td>5 / 3</td>
<td>1.6666</td>
</tr>
<tr>
<td>5 % 3</td>
<td>2</td>
</tr>
<tr>
<td>5 % -3</td>
<td></td>
</tr>
<tr>
<td>1 / 0</td>
<td></td>
</tr>
<tr>
<td>3 * 5 − 2</td>
<td>13</td>
</tr>
<tr>
<td>3 + 5 / 2</td>
<td>8</td>
</tr>
<tr>
<td>3 − 5 / 2</td>
<td>2</td>
</tr>
<tr>
<td>(3 − 5) / 2</td>
<td>2</td>
</tr>
<tr>
<td>3 − (5 − 2) / 2</td>
<td>3</td>
</tr>
<tr>
<td><code>Integer.parseInt(&quot;3&quot;)</code></td>
<td></td>
</tr>
<tr>
<td><code>Integer.parseInt(3)</code></td>
<td></td>
</tr>
</tbody>
</table>

*Section 1.2*
Integers: Example Program

```java
public class IntOps {
    public static void main(String[] args) {
        int a = Integer.parseInt(args[0]);
        int b = Integer.parseInt(args[1]);
        int sum = a + b;
        int prod = a * b;
        int quot = a / b;
        int rem = a % b;
        System.out.println(a + " + " + b + " = " + sum);
        System.out.println(a + " * " + b + " = " + prod);
        System.out.println(a + " / " + b + " = " + quot);
        System.out.println(a + " % " + b + " = " + rem);
        System.out.println(a + " = " + quot + " * " + b + " + " + rem);
    }
}
```

Download IntOps.java from book site, section 1.2

Welcome to DrJava. Working directory is /Users/bjbrown/introcs

```
> java IntOps 5 3
5 + 3 = 8
5 * 3 = 15
5 / 3 = 1
5 % 3 = 2
5 = 1 * 3 + 2
```

Section 1.2
Integers: Example Program

```java
public class IntOps {
    public static void main(String[] args) {
        int a = Integer.parseInt(args[0]);
        int b = Integer.parseInt(args[1]);
        int sum = a + b;
        int prod = a * b;
        int quot = a / b;
        int rem = a % b;
        System.out.println(a + " + " + b + " = " + sum);
        System.out.println(a + " * " + b + " = " + prod);
        System.out.println(a + " / " + b + " = " + quot);
        System.out.println(a + " % " + b + " = " + rem);
        System.out.println(a + " = " + quot + " * " + b + " + " + rem);
    }
}
```

Download `IntOps.java` from book site, section 1.2.

Program Arguments:
- `args[0]` = 5
- `args[1]` = 3

Section 1.2
**Section 1.2**

**double**: Floating-Point (fractions)

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result?</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.141 + 0.03</td>
<td></td>
</tr>
<tr>
<td>6.02e23 / 2.0</td>
<td></td>
</tr>
<tr>
<td>5.0 / 3</td>
<td></td>
</tr>
<tr>
<td>(int) 5.0 / 3</td>
<td></td>
</tr>
<tr>
<td>5.0 / (int) 3</td>
<td></td>
</tr>
<tr>
<td>10.0 % 3.141</td>
<td></td>
</tr>
<tr>
<td>1.0 / 0.0</td>
<td></td>
</tr>
<tr>
<td>-1.0 / 0.0</td>
<td></td>
</tr>
<tr>
<td>0.0 / 0.0</td>
<td></td>
</tr>
<tr>
<td>Math.sqrt(2)</td>
<td></td>
</tr>
<tr>
<td>Math.sqrt(-1)</td>
<td></td>
</tr>
<tr>
<td>Math.sqrt(2) * Math.sqrt(2)</td>
<td></td>
</tr>
<tr>
<td>Math.PI</td>
<td></td>
</tr>
<tr>
<td>Math.pi</td>
<td></td>
</tr>
</tbody>
</table>
Doubles: Example Program

```java
public class Quadratic {
    public static void main(String[] args) {
        // parse coefficients from command-line
        double b = Double.parseDouble(args[0]);
        double c = Double.parseDouble(args[1]);

        // calculate roots
        double discriminant = b*b - 4.0*c;
        double d = Math.sqrt(discriminant);
        double root1 = (-b + d) / 2.0;
        double root2 = (-b - d) / 2.0;

        // print them out
        System.out.println(root1);
        System.out.println(root2);
    }
}
```

Solve: \( x^2 + bx + c = 0 \)

Quadratic Formula: \( \frac{-b \pm \sqrt{b^2 - 4c}}{2} \)

Section 1.2
### Java Math Library (Excerpts)

```java
public class Math {
    double abs(double a)  // absolute value of a
    double max(double a, double b)  // maximum of a and b
    double min(double a, double b)  // minimum of a and b

    Note 1: abs(), max(), and min() are defined also for int, long, and float.

    double sin(double theta)  // sine function
    double cos(double theta)  // cosine function
    double tan(double theta)  // tangent function

    Note 2: Angles are expressed in radians. Use toDegrees() and toRadians() to convert.
    Note 3: Use asin(), acos(), and atan() for inverse functions.

    double exp(double a)  // exponential (e^a)
    double log(double a)  // natural log (log_e a, or ln a)
    double pow(double a, double b)  // raise a to the bth power (a^b)

    long round(double a)  // round to the nearest integer
    double random()  // random number in [0, 1)
    double sqrt(double a)  // square root of a

    double E  // value of e (constant)
    double PI  // value of π (constant)
}
```
**char**: Single Characters

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result?</th>
</tr>
</thead>
<tbody>
<tr>
<td>'A'</td>
<td></td>
</tr>
<tr>
<td>'A' + 0</td>
<td></td>
</tr>
<tr>
<td>(int) 'A'</td>
<td></td>
</tr>
<tr>
<td>(char) 65</td>
<td></td>
</tr>
<tr>
<td>(int) 'a'</td>
<td></td>
</tr>
<tr>
<td>(int) '0'</td>
<td></td>
</tr>
<tr>
<td>'3' – '0'</td>
<td></td>
</tr>
</tbody>
</table>
**char**: Single Characters

Single characters are stored as (small) integers!

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result?</th>
</tr>
</thead>
<tbody>
<tr>
<td>'A'</td>
<td></td>
</tr>
<tr>
<td>'A' + 0</td>
<td></td>
</tr>
<tr>
<td>(int) 'A'</td>
<td></td>
</tr>
<tr>
<td>(char) 65</td>
<td></td>
</tr>
<tr>
<td>(int) 'a'</td>
<td></td>
</tr>
<tr>
<td>(int) '0'</td>
<td></td>
</tr>
<tr>
<td>'3' - '0'</td>
<td></td>
</tr>
</tbody>
</table>
**char**: Single Characters

Single characters are stored as (small) integers!

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result?</th>
</tr>
</thead>
<tbody>
<tr>
<td>'A'</td>
<td></td>
</tr>
<tr>
<td>'A' + 0</td>
<td></td>
</tr>
<tr>
<td>(int) 'A'</td>
<td></td>
</tr>
<tr>
<td>(char) 65</td>
<td></td>
</tr>
<tr>
<td>(int) 'a'</td>
<td></td>
</tr>
<tr>
<td>(int) '0'</td>
<td></td>
</tr>
<tr>
<td>'3' – '0'</td>
<td></td>
</tr>
</tbody>
</table>

**Character codes are defined by the ASCII and Unicode standards.**

*Section 1.2*
### boolean: True/False

**true, false, ==, !=, <, >, <=, >=, && (and), || (or), ! (not)**

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result?</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
<tr>
<td>!false</td>
<td></td>
</tr>
<tr>
<td>'A' == 'a'</td>
<td></td>
</tr>
<tr>
<td>Math.PI != 3.14</td>
<td></td>
</tr>
<tr>
<td>'a' &gt; 'b'</td>
<td></td>
</tr>
<tr>
<td>1.7 &lt;= (17 / 10)</td>
<td></td>
</tr>
<tr>
<td>true &amp;&amp; true</td>
<td></td>
</tr>
<tr>
<td>true &amp;&amp; false</td>
<td></td>
</tr>
<tr>
<td>false &amp;&amp; false</td>
<td></td>
</tr>
<tr>
<td>true</td>
<td></td>
</tr>
<tr>
<td>true</td>
<td></td>
</tr>
<tr>
<td>false</td>
<td></td>
</tr>
<tr>
<td>(1 &lt; 3) &amp;&amp; (3 == (6 / 2))</td>
<td></td>
</tr>
<tr>
<td>(1 &gt;= 3)</td>
<td></td>
</tr>
</tbody>
</table>
Booleans: Example Program

```java
public class LeapYear {
    public static void main(String[] args) {
        int year = Integer.parseInt(args[0]);
        boolean isLeapYear;

        // divisible by 4 but not 100
        isLeapYear = (year % 4 == 0) && (year % 100 != 0);

        // or divisible by 400
        isLeapYear = isLeapYear || (year % 400 == 0);

        System.out.println(isLeapYear);
    }
}
```

Welcome to DrJava. Working directory is /Users/bjbrown/introcs
> java LeapYear 2004
true
> java LeapYear 1900
false
> java LeapYear 2000
true

Download LeapYear.java from booksite, section 1.2

George Boole
1815 – 1864

Section 1.2
Booleans: Example Program

```java
public class LeapYear {
    public static void main(String[] args) {
        int year = Integer.parseInt(args[0]);
        boolean isLeapYear;

        // divisible by 4 but not 100
        isLeapYear = (year % 4 == 0) && (year % 100 != 0);

        // or divisible by 400
        isLeapYear = isLeapYear || (year % 400 == 0);

        System.out.println(isLeapYear);
    }
}
```

Leap Years are:
- Divisible by 4
- But not divisible by 100
- Except if they're divisible by 400

Download LeapYear.java from booksite, section 1.2
## String: Text

### Table of Expressions and Results

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result?</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;This is a string literal.&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;1&quot; + &quot;2&quot;</td>
<td></td>
</tr>
<tr>
<td>1 + &quot; + &quot; + 2 + &quot; = &quot; + 3</td>
<td></td>
</tr>
<tr>
<td>'1' + &quot;2&quot;</td>
<td></td>
</tr>
<tr>
<td>0 + '1' + &quot;2&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;&quot; + Math.sqrt(2)</td>
<td></td>
</tr>
<tr>
<td>(String) Math.sqrt(2)</td>
<td></td>
</tr>
<tr>
<td>(string) Math.sqrt(2)</td>
<td></td>
</tr>
<tr>
<td>&quot;A&quot; == &quot;A&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;A&quot;.equals(&quot;A&quot;)</td>
<td></td>
</tr>
<tr>
<td>&quot;B&quot; &lt; &quot;A&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;B&quot;.compareTo(&quot;A&quot;)</td>
<td></td>
</tr>
<tr>
<td>&quot;B&quot;.compareTo(&quot;B&quot;)</td>
<td></td>
</tr>
<tr>
<td>&quot;B&quot;.compareTo(&quot;C&quot;)</td>
<td></td>
</tr>
</tbody>
</table>
Strings: Example Program

```java
public class Ruler {
    public static void main(String[] args) {
        String ruler1 = "1";
        String ruler2 = ruler1 + " 2 " + ruler1;
        String ruler3 = ruler2 + " 3 " + ruler2;
        String ruler4 = ruler3 + " 4 " + ruler3;
        System.out.println(ruler4);
    }
}
```

Welcome to DrJava. Working directory is /Users/bjbrown/introcs
> java Ruler
1 2 1 3 1 2 1 4 1 2 1 3 1 2 1
>

Download Ruler.java from booksite, section 1.2
Data Types

• `int`, `double`, `char`, `boolean`, `String`, ...

• Help avoid errors and ambiguities
  – What does `a + b` do?

• Not perfect:

Ariane 5: Bad type conversion

Mars Climate Orbiter: Bad unit conversion