Java Basics to Learn on Your Own*

• Legal identifier formats
• Variable and constant naming conventions
• Arithmetic operators
• Binary operators
• Unary operators
• Ternary operator
• Logical operators
• Relational operators
• Pre and post-increment/decrement operators
• Operator precedence
• if-then, if-then-else, while

*A link to the spring 2012 slides is on the 202 web site’s main page under Recent Announcements. These topics may be found there. And don’t forget Google!
Java Data Types

Primitive

FF00 25
x

FF52 3.14
pi

Reference

A123 FF00 “Bubba” FF00
name

More later on reference variables!
## Java Primitive Types

<table>
<thead>
<tr>
<th>Type Name</th>
<th>Type of Value</th>
<th>Memory Used</th>
<th>Value Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>byte</td>
<td>integer</td>
<td>1 byte</td>
<td>-128 to 127</td>
</tr>
<tr>
<td>short</td>
<td>integer</td>
<td>2 bytes</td>
<td>-32768 to 32767</td>
</tr>
<tr>
<td>int</td>
<td>integer</td>
<td>4 bytes</td>
<td>-2147483648 to 2147483647</td>
</tr>
<tr>
<td>long</td>
<td>integer</td>
<td>8 bytes</td>
<td>-9223372036854775808 to 9223372036854775807</td>
</tr>
<tr>
<td>float</td>
<td>floating point</td>
<td>4 bytes</td>
<td>-3.40282347 X 10^{+38} to -1.40239846 X 10^{-45}</td>
</tr>
<tr>
<td>double</td>
<td>floating point</td>
<td>8 bytes</td>
<td>+-1.76769313486231570 X 10^{+308} to +-4.94065645841246544 X 10^{-324}</td>
</tr>
<tr>
<td>boolean</td>
<td>true, false</td>
<td>1 byte</td>
<td>not applicable</td>
</tr>
<tr>
<td>char</td>
<td>single character</td>
<td>2 bytes</td>
<td>all Unicode characters</td>
</tr>
</tbody>
</table>
Declaring and Using Variables (Basics)

Python

```python
salary = 25000.0
age = 32
name = 'Joe'
print(age)
print(myAge)
```

Java*

```java
(double) salary = 25000.0;
int age;
age = 32;
String name = "Joe";
System.out.println(age);
System.out.println(myAge);
```

```
linux3[9]% python declarations.py
32
Traceback (most recent call last):
  File "declarations.py", line 5, in <module>
    print (myAge)
NameError: name 'myAge' is not defined
```

```
linux3[28]% javac Declarations.java
Declarations.java:9: cannot find symbol
  symbol  : variable myAge
location: class Declarations
    System.out.println(myAge);
         ^

*no global variables allowed in Java
```
Printing to the Screen

• Unformatted Output
  • System.out.print( ...); leaves cursor on same line
  • System.out.println( ... ); cursor moves to next line

• Example Snippet:
  System.out.print("Hello");
  System.out.print(" there");
  System.out.println("Hello");
  System.out.println(" there");

Output:
Hello there
Hello there
Printing to the Screen (con’t)

• Formatted Output
  
  System.out.printf(\"format-string, parameter_1, ..., parameter_n\\);

• Example Snippet
  
  System.out.printf("Printing an integer: %d%n", 5);
  System.out.printf("%d %c %5.2f", 1, 'a', 2.123);
  String word = "Hello";
  System.out.printf("%s", word);

  **Output:**
  
  Printing an integer: 5
  1 a  2.12Hello

• Reference
  
  – Java API, PrintStream class, printf method
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, world!");
    }
}

linux3[95]% ls
Declarations.java HelloWorld2.java HelloWorld.java
linux3[96]% javac HelloWorld.java
linux3[97]% ls
Declarations.java HelloWorld2.java HelloWorld.class HelloWorld.java
linux3[98]% java HelloWorld
Hello, world!
More on Data Types

• **Simple** (integral, scalar)
  – Contains only one value at any given time
  – Java Examples:
    
    ```java
    int x = 5;  float e = 2.71;  char initial = ‘S’;
    ```

• **Complex** (aggregate, composite)
  – May contain one or more values
  – *Heterogeneous* or *homogenous*
  – Examples:
    • A *list* in Python (heterogeneous)
    • An *array* in Java (homogeneous)
Java Arrays

• Homogeneous
• Reference type
• Declaration Format:
  
  <data-type>[ ] <variable-name>;

• Example Declarations:
  
  int scores[ ];  char grades[ ];

So, what’s in memory?
public class ArraysOneDim {
    public static void main(String[] args) {
        int scores[];
        char grades[];
        System.out.println("scores: ", scores);
        System.out.println("grades: ", grades);
    }
}

java -classpath .:jars/* ArraysOneDim
arraysonedi
arraysonedi:2: cannot find symbol
arraysonedi:2: symbol : method println(java.lang.String,int[])
arraysonedi:2: location: class java.io.PrintStream
    System.out.println("scores: ", scores);
       ^
arraysonedi:3: cannot find symbol
arraysonedi:3: symbol : method println(java.lang.String,char[])
arraysonedi:3: location: class java.io.PrintStream
    System.out.println("grades: ", grades);
       ^
Java Arrays (con’t)

• Initializing an Array

\[
<\text{data-type}>[\ ] <\text{variable-name}> = \text{new} <\text{data-type}>[\text{number-of-elements}];
\]

• Example:

\[
\text{int scores[ ] = new int[8]; OR}\\
\text{int scores[ ]; scores = new int[8];}
\]
## Java Arrays (con’t)

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Default Value (for elements)</th>
</tr>
</thead>
<tbody>
<tr>
<td>byte</td>
<td>0</td>
</tr>
<tr>
<td>short</td>
<td>0</td>
</tr>
<tr>
<td>int</td>
<td>0</td>
</tr>
<tr>
<td>long</td>
<td>0L</td>
</tr>
<tr>
<td>float</td>
<td>0.0f</td>
</tr>
<tr>
<td>double</td>
<td>0.0d</td>
</tr>
<tr>
<td>char</td>
<td>'\u0000'</td>
</tr>
<tr>
<td>String (or any reference)</td>
<td>null</td>
</tr>
<tr>
<td>boolean</td>
<td>false</td>
</tr>
</tbody>
</table>
scores[0] = 28;

scores[4] = -1;

System.out.print(scores[3]);

System.out.print(scores.length);
Multi-dimensional Arrays

• Two-dimensional Array Declaration:
  
  `<data-type>[ ][ ] <variable-name>`;

• Example Declaration:
  
  `char [ ][ ] ticTacToeBoard;`

• Initializing:
  
  `ticTacToeBoard = new char[3][3];    OR`
  `three = 3;`
  `ticTacToeBoard = new char[three][three];`

```
<table>
<thead>
<tr>
<th>[0]</th>
<th>[1]</th>
<th>[2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[0]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[2]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
Multi-dimensional Arrays (con’t)

ticTacToeBoard[0][2] = ‘X’;

char move = ‘O’;
ticTacToeBoard[1][1] = move;

System.out.print(ticTacToeBoard[0][0]);

int square1 = 2, square2 = 1;
ticTacToeBoard[square1][square2] = ‘O’;
Multi-dimensional Arrays (con’t)

Best to think of a two-dimensional array as an “array of arrays”

Each contains the memory address of its corresponding row