Using Classes and Objects

CSC 1051 – Data Structures and Algorithms I
Dr. Mary-Angela Papalaskari
Department of Computing Sciences
Villanova University

Course website:
www.csc.villanova.edu/~map/1051/

Packages
• For purposes of accessing them, classes in the Java API are organized into packages

<table>
<thead>
<tr>
<th>Package</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>java.lang</td>
<td>General support</td>
</tr>
<tr>
<td>java.applet</td>
<td>Creating applets for the web</td>
</tr>
<tr>
<td>java.awt</td>
<td>Graphics and graphical user interfaces</td>
</tr>
<tr>
<td>javax.swing</td>
<td>Additional graphics capabilities</td>
</tr>
<tr>
<td>java.net</td>
<td>Network communication</td>
</tr>
<tr>
<td>java.util</td>
<td>Utilities</td>
</tr>
<tr>
<td>javax.xml.parsers</td>
<td>XML document processing</td>
</tr>
</tbody>
</table>

imported automatically, includes String and Math classes

The Math Class
• The Math class is part of the java.lang package and contains methods for mathematical functions
  – No need to import anything!
  – The Math class methods are static
  – Static methods are invoked through the class name

value = Math.cos(phi) + Math.sqrt(delta);

See Quadratic.java
Some methods from the Math class

- `public class Math`<br>- `double abs(double a)` absolute value of a<br>- `double max(double a, double b)` maximum of a and b<br>- `double min(double a, double b)` minimum of a and b<br>- `double sin(double theta)` sine function<br>- `double cos(double theta)` cosine function<br>- `double tan(double theta)` tangent function<br>- `double exp(double a)` exponential function<br>- `double log(double a)` natural log (log, a, or ln a)<br>- `double pow(double a, double b)` raise a to the bth power (ab)<br>- `long round(double a)` round to the nearest integer<br>- `double random()` random number in [0, 1)<br>- `double sqrt(double a)` square root of a<br>- `double E` value of e (constant)<br>- `double PI` value of π (constant)

Example: Global Warming

```java
// Dangerous rising sea levels in front of snowman
page.setColor(Color.blue);

final int APPLET_HEIGHT = 225, APPLET_WIDTH = 300;
final int WAVE_HEIGHT = 25;
final double SCALE_FACTOR = 0.06; // adjust to get more/fewer waves

int position = 0;
while (position < APPLET_WIDTH) {
    double waveFunction = WAVE_HEIGHT * Math.sin(position * SCALE_FACTOR);
    int topOfWave = (int) (waveFunction + APPLET_HEIGHT / 2);
    page.fillRect(position, topOfWave, 1, APPLET_HEIGHT - topOfWave);
    position++;
}
```

The Random Class

- **Part of the java.util package, so import it**

```java
import java.util.Random;
```

- **Create a Random object named gen:**

```java
Random gen = new Random();
```

- **Use Random method nextInt() to generate a random number:**

```java
int a = gen.nextInt(4);
// integer in range [0,1,2,3]
```

What is a random number?

"Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin."
- John Von Neumann

"God does not play dice."
- Albert Einstein

The Random class provides methods that generate **pseudorandom numbers**
Example: Using Random methods

```java
Random gen = new Random();
int a = gen.nextInt(4);
    // integer in range [0,1,2,3]
float b = gen.nextFloat();
    //float in range [0,1), eg: 0.4589
int c = gen.nextInt(4) + 1;
    //int in range [1,2,3,4]
int d = gen.nextInt();
    //int in range [-2147483648 … 2147483647]
```

List of some Random methods: page 126

See RandomNumbers.java

Example: counting “snake eyes”

```java
// Roll two dice 100,000 times and count how many
// times you roll snake eyes, i.e., two 1's.
Random gen = new Random();
int trial = 0, count = 0;
while (trial < 100000)
{
    int die1 = gen.nextInt(6) + 1;
    int die2 = gen.nextInt(6) + 1;
    if (die1 == 1 && die2 == 1)
        count++; // snake eyes
    trial++;
}
System.out.println("Probability of snake eyes = "+
    (double)count/100000);
```
Quick Check
Given a Random object named `gen`, what range of values are produced by the following expressions?

- `gen.nextInt(25)`
- `gen.nextInt(6) + 1`
- `gen.nextInt(50) + 100`
- `gen.nextInt(10) - 5`
- `(int)(gen.nextFloat() * 10 + 1)`
- `(int)(Math.random() * 10 + 1)`

alternative way to generate pseudorandom number in the range 0...1

Quick Check
Given a Random object named `gen`, write an expression that produces a random integer in the following ranges:

- **Range**
  - 0 to 12
  - 1 to 20
  - 15 to 20
  - -10 to 0

The Strings Class
- Strings are objects defined by the `String` class
  - "This is a string literal."
  - "123 Main Street"
  - "x"
- the `String` class has many methods that can be used to process text. Examples:
  - finding the length of a string
  - finding the char at a certain position of a string
  - producing an all-caps version of a string

Invoking String Methods
- As with other kinds of objects, we use the **dot operator** to invoke a String’s methods:

  ```java
  int numOfCharsInName = name.length();
  ```

  `length()` is one of the methods of `String` objects (defined in the `String` class)
More String Methods

- List of some String methods: see textbook, page 119

String name = "Betsy";
char initial = name.charAt(0);
String newName = name.replace('s', 't');
String capsName = name.toUpperCase();
int comp = name.compareTo(newName);

See also textbook example StringMutation.java

Example: Palindrome tester

- Problem: Input a string, determine whether it is a palindrome, i.e.:
  - first char is the same as last char
  - 2nd char is the same as 2nd last char
  - and so on...
- How to express this as an algorithm?
- How to implement it?

System.out.println("Enter a potential palindrome:");
str = scan.nextLine();
left = 0;
right = str.length() - 1;
while (str.charAt(left) == str.charAt(right) && left < right) {
    left++;
    right--;
}    
if (left < right) System.out.println("NOT a palindrome"); else System.out.println("palindrome");
Declaring Variables, revisited
- Examples of variable declarations:
  ```java
  int count = 0;
  double mpg;
  String title;
  Graphics page;
  Color aquamarine;
  Scanner scan;
  ```
- A **class name** can be used as a type to declare an object reference variable
- **The object itself must be created separately**

Creating Objects
- We have already seen something like this:
  ```java
  Scanner scan = new Scanner (System.in);
  ```
  - The **new operator** calls the **Scanner constructor**, which is a special method that sets up the object
  - Variable refers to a **Scanner object**
  - Constructing a new object is called **instantiation**

Creating Objects
- Another example:
  ```java
  String title = new String ("Java Software Solutions");
  ```
  - The **new operator** calls the **String constructor**, which is a special method that sets up the object
  - Variable refers to a **String object**
  - Constructing a new object is called **instantiation**

The String Class is SPECIAL!
- **Exception to the use of new operator**: Because strings are so common, we don't have to use the **new operator** to create a **String object**
  ```java
  String title = "Java Software Solutions";
  ```
  - This is special syntax that works only for strings