Applets, Arrays

23 January 2008
CMPT166
Dr. Sean Ho
Trinity Western University

TRINITY WESTERN UNIVERSITY

Some handy Math methods

- Class methods in Math module
  - sqrt(x)
  - abs(x)
  - max(x, y), min(x, y)
  - ceil(x), floor(x)
  - cos(x), sin(x), etc.
  - exp(x), log(x) (natural log)
  - pow(x, y) (y can be a float)
  - random() (double in range [0, 1))

Some handy standard packages

- java.lang: automatically imported
- java.io: files and streams
- java.net: networking
- java.text: manipulate strings, dates, i8n
- java.util: miscellaneous utilities: strings, etc.

- java.applet: or javax.swing.JApplet for Swing
- java.awt: or javax.swing
- java.awt.event: or javax.swing.event

Argument type promotion

- Java is statically typed but has some flexibility:
  - If pass a lower-precision value as a parameter, it can be promoted:
    - float --> double
    - long --> float or double
    - int --> long, float, or double
    - char --> int long, float, or double
    - short --> int, long, float, or double
    - byte --> short, int, long, float, or double
  - No promotions for boolean

Method overloading

- Overloading is giving multiple definitions for a method with the same name, but different argument types

```
public int square( int x ) {
    return x*x;
}
public double square( double x ) {
    return x*x;
}
int y=5; double z=2.3;
square(y); square(z)
```

  - Do we need a float version as well?

Scope and duration

- The duration (lifetime) of an identifier is the runtime period when it exists in memory
  - Automatic duration
    - Local variables disappear when block finishes
  - Static duration
    - As long as the object/module/program exists
- The scope of an identifier is the lexical extent where it can be referenced
  - Block scope
  - Class scope

Scope example

```
public class ScopeExample {
    int numApples = 0;              // class scope
    public void listApples() {
        int counter = 0;          // block scope
    }
}
```

- numApples is an instance variable with class scope: accessible to all methods of this class
- counter is a local variable with block scope: not accessible outside the listApples() method

Java Swing

- Swing is Java's built-in GUI toolkit
- Can build stand-alone GUI programs
- See "SayHello" example (cmpt166.seanho.com/java)
  - import javax.swing.*;
  - Input dialog: JOptionPane.showInputDialog()
  - Output: JOptionPane.showMessageDialog()
- See Sun's tutorial for more details

Java applets

- Addition example applet (“Lab0”)
  - import java.applet.Applet; // the Applet class
  - import java.awt.*; // abstract window toolkit
  - public class Addition extends Applet implements ActionListener {
- Only one public class per file
  - Named same as the file
- Extends: subclass inherits from Applet
- Implements: ActionListener interface
  - Must implement the following method:
  - public void actionPerformed(ActionEvent e)

Running an applet

- Compile an applet using javac as usual
- Run: in Eclipse: it will pop up a window
- Run: in a web page:
  - Write a small HTML file embedding the applet:
    - <object> (IE), or <applet>, or both
    - See Addition.html and TicTacToe.html
    - See Sun's recommendations
- Run: using appletviewer:
  - appletviewer Addition.html
    - HTML file, not the applet .class file directly

JApplet

- JApplet is Swing's way of doing applets

```
import javax.swing.*;
public class MyApplet extends JApplet {
```

- The abstract superclass JApplet defines various methods that our subclass overrides:
  - `public void init()` // when applet is loaded
  - `public void destroy()` // when applet is removed in memory
  - `public void start()` // after init() finishes: on page load
  - `public void stop()` // on page exit
  - `public void paint( Graphics g )` // on refresh/repaint

Arrays in Java

- Aggregate (compound/container) data type
  - All entries have same type
  - Size of array is fixed when array is allocated
    - But need not be known at compile-time
    - Arrays can be dynamically created
  - Location in memory is usually contiguous
  - Index into array using integer indices from 0 up to (size of array)-1
    - Indexing out-of-bounds raises ArrayIndexOutOfBoundsException

Working with arrays

- Declaring arrays:
  - int numApples[];
- Allocate array in memory:
  - numApples = new int[10];
- Initializing array entries:
  - numApples[3] = 15;
- Size of array:
  - numApples.length // returns 10

Array initializers and constants

- Initialize an array on one line:
  - int numApples[] = {5, 3, 12, 0, 3};
- Declare constants using the keyword final:
  - final int numApples[] = {5, 3, 12, 0, 3};
  - final float pi = 3.14159265358979323846264;
- (Histogram.java example)

Pass-by-value vs. pass-by-reference

- In Java, primitives (int, float, boolean, etc.) are passed by value
- Objects (including arrays) are passed by reference

Multidimensional arrays

- The element type of an array can be any type, including objects, including other arrays:

```
int image[][];
image = new int[width][height];
for (int x=0; x<width; x++)
    for (int y=0; y<width; y++)
        image[x][y] += 10;
```

- Rows may be different lengths:

```
image = new int[width][];
for (int x=0; x<width; x++)
    image[x] = new int[x];      // triangular array
```

Iterating through arrays

- Iterate through an array with a for loop:

```
for (int idx=0; idx < array.length; idx++)
    sum += array[idx];
```

- Java has an enhancement to the for loop:

```
for (int elt : array)
    sum += elt;
```

- But note elt is a copy of each element:
  - Can't use this to modify array

Sorting arrays: bubble sort

- Bubble sort: most straightforward sort algorithm
  - Smaller values "bubble" to start of array
  - Larger values "sink" to end of array
  - Use nested loops to make several passes through array
  - Each pass compares successive pairs of elements:
    - Pairs are swapped if in decreasing order

Sorting arrays: selection sort

- Bubble sort is not so fast but is easy to write
- Selection sort is a little faster and almost as easy:
  - Iterate through the list:
    - Find smallest value in the remainder of the list
    - Swap with current element

- Lots of other, better algorithms for sorting:
  - See CMPT231 and demos @UBC