CS1110, 8 March 2009
Two topics: elementary graphics (for A5); loops

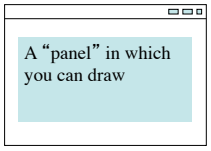
Reading: Sec. 2.3.8 and chapter 7 on loops.
The lectures on the ProgramLive CD can be a big help.
Videonote can be a big help.

Assignment A5: graphics, loops, recursion
- Due date: Thursday Mar 31 at 11:59pm

• **Prelim 1**: Tonight,! 7:30pm, *Uris Hall 101*

1

Graphical User Interfaces (GUIs): graphics.



A JFrame, with a "panel" on which you can draw

On the panel, each pair (x,y) indicates a "pixel" or picture element.

For Assignment 5, you need to understand that

x-coordinates increase rightward
y-coordinates increase downward.

(0,0) (1,0) (2,0) …
(0,1) (1,1) (2,1) …
(0,2) (1,2) (2,2) …
…

2

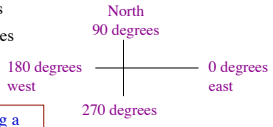
```
import javax.swing.*; import java.awt.*;
jframe= new JFrame("Turtle window");
jpanel= new JPanel();
jframe.getContentPane().add(jpanel, BorderLayout.CENTER);
jframe.pack();
jframe.setVisible(true);
graphics= jpanel.getGraphics(); // contains methods to draw on jpanel
// Draw line from (10, 10) to (50, 40).
graphics.drawLine(10,10,50, 40);

// Draw rectangle: top-left point (2, 5), width 40, height 60
graphics.drawRect(2, 5, 40, 60);

 // Draw string "this" at (40, 30)
 graphics.drawString("this", 40, 30);

 // set the pen color to red
 graphics.setColor(Color.red);

 // Store the current color in c
 Color c= graphics.getColor();
```

You don't have to learn all this now (unless you want to). We will be telling you more about later. For more on graphics, see class Graphics in the Java API and page 1-5 in the CD ProgramLive. For more on GUIs, read chapter 17. The corresponding part of the CD is arguably even more helpful.

3

Assignment A5: drawing with a Turtle

Use a class developed by the ACM:
- point (x, y): where the "Turtle" is
- angle: the direction the Turtle faces
- a pen color
- whether the pen is up or down



Class Turtle has methods for moving a Turtle around, drawing as it goes.

Draw equilateral triangle with side lengths 30; turtle ending up at starting point and facing the same direction:

```
t.forward(30);   t.left(120);
t.forward(30);   t.left(120);
t.forward(30);   t.left(120);
```

In A5, write methods to draw shapes and spirals, and draw things using recursive procedures.

4

From recursion to loops: doing things repeatedly

Write programs to make computers do things.
Often want to make them do things *many times*.

1. Perform *n* trials or get *n* samples.
   - A5: draw a triangle six times to make a hexagon
   - Run a protein-folding simulation for 10^6 time steps
2. Process each item in a given String, Vector, or other "list"
   - Compute aggregate statistics for a dataset, such as the mean, median, standard deviation, etc.
   - Send everyone in a certain (Facebook) group an individual appointment time
3. Do something an unknown number of times
   - CUAUV team, vehicle has to continue moving to get somewhere

From recursion to loops: doing things repeatedly

We haveve talked about *recursion*.
Alternatives:

while-loops

for-loops (special syntax for common special cases)

```
<set things up>;
while (stuff still to do) {
        <process current item>;
        <prepare for next item>;
}
```

6

The for loop, for processing a range of integers

```
x= 0;
// add the squares of ints
// in range 2..200 to x
x= x + 2*2;
x= x + 3*3;
…
x= x + 200*200;
```

for each number i in the range 2..200, add i*i to x.

loop counter: i
initialization: **int** i= 2;
loop condition: i <= 200;
increment: i= i + 1
repetend or body: { x= x + i*i; }

The for-loop:

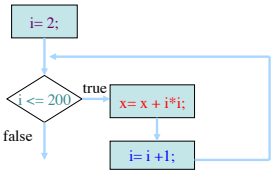
```
for (int i= 2; i <= 200; i= i +1) {
   x= x + i*i;
}
```


Execution of the for-loop

The for-loop:

```
for (int i= 2; i <= 200; i= i +1) {
   x= x + i*i;
}
```

loop counter: i
initialization: **int** i= 2;
loop condition: i <= 200;
increment: i= i + 1
repetend or body: { x= x + i; }



To execute the for-loop.
1. Execute initialization.
2. If loop condition false, terminate execution.
3. Execute repetend.
4. Execute increment, repeat from step 2.

Called a "flow chart"

Note on ranges.

2..5 contains 2, 3, 4, 5. It contains 5+1 – 2 = 4 values
2..4 contains 2, 3, 4. It contains 4+1 – 2 = 3 values
2..3 contains 2, 3. It contains 3+1 – 2 = 2 values
2..2 contains 2. It contains 2+1 – 2 = 1 values
2..1 contains . It contains 1+1 – 2 = 0 values

The number of values in **m..n** is **n+1 – m**.

In the notation m..n, we require always, without saying it, that

m <= n + 1 .

If m = n + 1, the range has 0 values.

Application: URL analysis for search engines

Problem: how does a search engine (e.g., Google) decide which webpages are the most important to present?

(Small) part of the answer: use URL cues

• "Deep" URLs are usually less important, e.g., www.fake.com/this/that/other/minor/tiny/detail.htm

This requires counting the number of slashes in a URL (given as a String).

The pattern for processing range of integers:

range a..b-1

```
for (int i= a; i < b; i= i + 1) {
   Process integer i;
}
```

range c..d

```
for (int i= c; i <= d; i= i + 1) {
   Process integer i;
}
```

```
// store in count # of '/' s in String s
// inv: count is # of '/' s in s[0..i-1]
count= 0;
for (int i= 0; i < s.length(); i= i +1) {
   if (s.charAt(i) == '/' )
         count= count+1;
}
// count is # of '/' s in s[0..s.length()-1]
```

```
// Store in double var. v the sum
//      1/1  + 1/2 + …+ 1/n
v= 0; // call this 1/0 for today
// inv: v is 1/1  + 1/2 + …+ 1/(i-1)
for (int i= 1; i <= n; i= i +1) {
         v= v + 1.0 / i;
}
// v= 1/1  + 1/2 + …+ 1/n
```


Our goal: Provide you with a methodology for the development of loops that process a range of integers.

1. Separate concerns — focus on one thing at a time.
2. Make small steps toward completing the loop.
3. Don't introduce a new variable without a good reason.
4. Keep program simple.

Try these. Test in DrJava

1. Set c to the number of chars is String s that are digits (in 0..9).
2. Store in res a copy of String s but with no blanks.
3. Store in res a copy of String s but with adjacent duplicates removed.
4. Set boolean v to the value of "no integer in 2..n–1 divides x".
5. Set boolean v to the value of "every element in Vector v is an object of class JFrame".
6. Add up the squares of the odd integers in the range m..n.