Mehran Sahami
CS 106A

Handout #33
November 6, 2017

Solutions to Midterm Exam

Problem 1: Karel the Robot

```
/* File: CopyPatternKarel.java                                          */
/* Karel copies the pattern of beepers in avenues 2 to 5 into avenues 7 to 10.   */

import stanford.karel.*;

public class CopyPatternKarel extends SuperKarel {
   /* Copy pattern of beepers in avenues 2 to 5 row-by-row into avenues 7 to 10. */
   public void run() {
      move();
      while (leftIsClear()) {
         copyOneLine();
         repositionForNextLine();
      }
      copyOneLine();   // Fence post problem: remember to copy last row of beepers.
   }

   /* Copy the pattern of beepers in the four corners ahead of Karel into the    */
    * corresponding positions 5 corners away.                                    */
   private void copyOneLine() {
      for(int i = 0; i < 4; i++) {
         if (beepersPresent()) {
            copyBeeper();
         }
         move();
      }
   }

   /* Copy a single beeper from the current corner to the one five corners away  */
    * and then return to the current location/direction.                         */
   private void copyBeeper() {
      moveBetweenPatterns();
      putBeeper();
      moveBetweenPatterns();
   }

   /* Move from one pattern to the corresponding position in the other pattern.  */
   private void moveBetweenPatterns() {
      for(int i = 0; i < 5; i++) {
         move();
      }
      turnAround();
   }

   /* Assuming Karel is at end of a row of beepers facing East, move Karel up    */
    * to the beginning of the next higher row and have Karel face East.          */
   private void repositionForNextLine() {
      turnLeft();
      move();
      turnLeft();
      for(int i = 0; i < 4; i++) {
         move();
      }
      turnAround();
   }
}
```

Problem 2: Simple Java expressions, statements, and methods

(2a)

`10 - 6 / 4 / 2.0`	`9.5`
`2 < 5 / 2 && 9 % 2 == 1 || 3 > 4 % 1`	`true`
`('c' - 'a') + "A" + 'b'`	`"2Ab"` (just `2Ab` is fine too)

(2b) What output is printed by the following program:

```
First result: 6
Second result: 7
```

Problem 3: Simple Java program using the random number library

```
/* File: CountRuns.java                                                    */
/* This program simulates a sequence of die rolls and prints               */
/* out the number of runs (consecutives rolls of same number).             */

import acm.program.*;
import acm.util.*;

public class CountRuns extends ConsoleProgram {

   // NUM_ROLLS is the number of times the die is rolled in the simulation
   private static final int NUM_ROLLS = 15;

   // NUM_SIDES is the number of sides on the die rolled (e.g., 6-sided die)
   private static final int NUM_SIDES = 6;

   public void run() {
      int runCount = 0;
      int lastRoll = -1;      // At start, there is no valid previous roll
      boolean inRun = false;  // At start, we are not in a run already

      for (int i = 0; i < NUM_ROLLS; i++) {
         int roll = rgen.nextInt(1, NUM_SIDES);
         println("Roll: " + roll);

         if (roll == lastRoll) { // Current roll matches previous roll
            if (!inRun) {        // If not already in a run, start new run
               runCount++;
               inRun = true;
            }
         } else {
            inRun = false;    // Note that the previous run ended
         }
         lastRoll = roll;     // Keep track of last roll to keep track of run
      }

      println("Number of runs: " + runCount);
   }

   /* Private instance variable */
   private RandomGenerator rgen = RandomGenerator.getInstance();
}
```

Problem 4: Using the graphics libraries

```
/* File: DrawLines.java */
/* This program lets the user draw lines on the graphics canvas. Each line is    */
/* defined by a pair of mouse clicks, marking a line's beginning and ending.     */

import acm.program.*;
import acm.graphics.*;
import java.awt.*;
import java.awt.event.*;

public class DrawLines extends GraphicsProgram {

   // Diameter of circle used to mark location of first mouse click in each pair
   private static final int CIRCLE_DIAM = 10;

   public void init() {    // This could have also been a "run" method
      firstClickX = -1;    // Indicates that we have no first click in a pair
      addMouseListeners();
   }

   // When the mouse is clicked in the graphics canvas we need to determine if
   // it's a first click (and store it), or the second click in a pair (and
   // draw a line).
   public void mouseClicked(MouseEvent e) {
      int x = e.getX();
      int y = e.getY();

      if (firstClickX == -1) {         // If first click, keep track of it
         firstClickX = x;
         firstClickY = y;

         // Draw a red circle centered at the click position
         oval = new GOval(firstClickX - (CIRCLE_DIAM / 2),
                          firstClickY - (CIRCLE_DIAM / 2),
                          CIRCLE_DIAM, CIRCLE_DIAM);
         oval.setColor(Color.RED);
         add(oval);

      } else {                   // This must be a second click

         // Draw line from first click position to current click position
         add(new GLine(firstClickX, firstClickY, x, y));
         remove(oval);              // Remove red circle drawn on first click

         // "Reset" to prepare for next click being a first click
         firstClickX = -1;
      }
   }

   /* Private instance variables */
   private GOval oval;
   private int firstClickX;   // X loc. of "first click" in pair (-1 if none)
   private int firstClickY;   // Y loc. of "first click" in pair
}
```

Problem 5: Strings and characters

```
/* This method returns a String which is the result of interleaving the    */
/* characters in Strings str1 and str2.  If one of the input Strings is    */
/* longer than the other, the extra characters from the longer String are  */
/* simply appended to the end of the resulting String.                     */
private String interleave(String str1, String str2) {
   String smallerString;   // Used to store the shorter of str1 and str2
   String largerString;    // Used to store the longer of str1 and str2
   String result = "";

   // Determine which string (str1 or str2) is longer
   if (str1.length() < str2.length()) {
      smallerString = str1;
      largerString = str2;
   } else {
      smallerString = str2;
      largerString = str1;
   }

   // Append alternating (interleaved) characters to the result string
   for(int i = 0; i < smallerString.length(); i++) {
      result += str1.charAt(i);
      result += str2.charAt(i);
   }

   // Copy over remaining portion of larger string to the result
   result += largerString.substring(smallerString.length());
   return result;
}
```