Eric Roberts
CS 106A

Handout #13A
April 9, 2012

Solutions for Section #1

As always, there are many ways to structure the code for the Karel problem. This code takes advantage of the fact that a column of three beepers appears three times in the construction of the house, making it useful to have a method like **`putThreeBeepers`**.

1. Repairing damage from Katrina

```
/*
 * File: KatrinaRepairKarel.java
 * -----------------------------
 * The KatrinaRepairKarel class solves the problem from the midterm
 * exam in which the robot builds houses at corners marked by rubble.
 */

import stanford.karel.*;

public class KatrinaRepairKarel extends SuperKarel {

/* Main program */
   public void run() {
      while (frontIsClear()) {
         if (beepersPresent()) {
            pickBeeper();
            backUp();
            buildHouse();
         }
         if (frontIsClear()) {
            move();
         }
      }
   }

/*
 * Builds a beeper house on stilts.
 * Precondition: Karel facing East at bottom of left stilt
 * Postcondition: Karel facing East at bottom of right stilt
 */
   private void buildHouse() {
      turnLeft();
      putThreeBeepers();
      move();
      turnRight();
      move();
      turnRight();
      putThreeBeepers();
      turnAround();
      move();
      turnRight();
      move();
      turnRight();
      putThreeBeepers();
      turnLeft();
   }
```

☞

```
/*
 * Creates a line of three beepers.
 * Precondition: Karel is in the first square in the line
 * Postcondition: Karel is in the last square in the line
 */
   private void putThreeBeepers() {
      for (int i = 0; i < 2; i++) {
         putBeeper();
         move();
      }
      putBeeper();
   }

/*
 * Backs up one corner, leaving Karel facing in the same direction.
 * If there is no space behind Karel, it will run into a wall.
 */
   private void backUp() {
      turnAround();
      move();
      turnAround();
   }

}
```

2. Simple expressions

```
/*
 * File: FeetAndInchesToCentimeters.java
 * -------------------------------------
 * This program converts a distance measured in feet and inches
 * to the equivalent distance in centimeters.
 */

import acm.program.*;

public class FeetAndInchesToCentimeters extends ConsoleProgram {

   public void run() {
      println("This program converts feet and inches to centimeters.");
      int feet = readInt("Enter number of feet: ");
      int inches = readInt("Enter number of inches: ");
      int totalInches = feet * INCHES_PER_FOOT + inches;
      double cm = totalInches * CENTIMETERS_PER_INCH;
      println(feet + "ft " + inches + "in = " + cm + "cm");
   }

/* Private constants */
   private static final int INCHES_PER_FOOT = 12;
   private static final double CENTIMETERS_PER_INCH = 2.54;

}
```

3. Precedence

The answer with Java's precedence rules is –50.