Solutions to Practice Midterm

The midterm is scheduled for the following time and room:

Tuesday, July 17, 7:00–8:30 p.m., Hewlett 200

Please remember that the midterm is open-book.

Problem 1: Karel the Robot

There are many strategies for solving this 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`.

```java
import stanford.karel.*;

public class KatrinaRepairKarel extends SuperKarel {

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

/** Backs up one corner.
* Precondition: There is an open square behind Karel
* Postcondition: Karel is in that square facing the same direction.
*/
private void backUp() {
   turnAround();
   move();
   turnAround();
}
```

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

/**
 * 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();
}

Problem 2: Simple C expressions, statements, and functions

(2a)  \[
      \frac{5.0}{4} - \frac{4}{5} \quad \frac{7}{9} - \frac{5}{3} \mod 0 = 3
    \]
    \[
      1.25, \quad \text{false}
    \]

(2b)  "cabbage"

(2c)

![Problem2c](image-url)
Problem 3: Simple Java – Most in a Row

```java
private RandomGenerator rgen = RandomGenerator.getInstance();

private void run() {
    int station = STARTING_STATION;
    int numStopsLeft = NUM_STOPS;
    println("Start at station 
    while(numStopsLeft > 0) {
        boolean south = rgen.nextBoolean();
        int numStations = rgen.nextInt(1,6);
        String dir = "north";
        int dstStation = station + numStations;
        if(south) {
            dstStation = station - numStations;
            dir = "south";
        }

        String msg = "Go " + dir + " " + numStations + " stations ";
        if(dstStation >= 0 && dstStation < NUM_STATIONS) {
            msg += "to station ";
            numStopsLeft--;
        } else {
            msg += "-- Out of bounds. Roll again.";
        }
        println(msg);
    }

    println("End at station ");
}
```
Problem 4: Graphics – Fail Stamp

public class FailStamp extends GraphicsProgram {

    private static final String STAMP = "stamp.jpg";
    private static final String PRESSED = "stampPressed.jpg";
    private static final String FAIL = "fail.jpg";

    // Add instance variables here
    private GImage stamp;

    public void run() {
        stamp = addImageCenteredAt(STAMP, 0, 0);
        addMouseListeners();
    }

    private GImage addImageCenteredAt(String image, int x, int y) {
        GImage r = new GImage(image);
        centerAt(r, x, y);
        return r;
    }

    private void centerAt(GObject obj, int x, int y) {
        add(obj, x - obj.getWidth()/2, y - obj.getHeight()/2);
    }

    public void mouseMoved(MouseEvent e) {
        centerAt(stamp, e.getX(), e.getY());
    }

    public void mousePressed(MouseEvent e) {
        stamp.setImage(PRESSED);
    }

    public void mouseReleased(MouseEvent e) {
        addImageCenteredAt(FAIL, e.getX(), e.getY());
        stamp.setImage(STAMP);
    }
}