Jerry Cain
CS 106AJ

Handout #23S
October 24, 2018

Solutions to Practice Midterm #1

Please remember that the midterm is open-book.
Tuesday, October 30, 3:30–5:30 P.M., Educ 128
Tuesday, October 30, 7:00–9:00 P.M., 380-380C

Problem 1: Karel the Robot (10 points)

```
/*
 * File: BreakoutKarel.k
 * ---------------------
 * The BreakoutKarel program plays a simplified form of Breakout.
 */

import "turns";

/* Main program */

function Breakout() {
   while (beepersInBag()) {
      if (beepersPresent()) {
         pickBeeper();
         bounce();
      }
      while (frontIsBlocked()) {
         bounce();
      }
      stepDiagonally();
   }
}

/*
 * Causes Karel to perform a ricochet bounce, which requires
 * no more than turning left.
 */

function bounce() {
   turnLeft();
}

/*
 * Step diagonally.  The precondition for this call is that
 * Karel's front must be clear.  The postcondition has Karel
 * facing in the same direction.
 */

function stepDiagonally() {
   move();
   if (leftIsClear() && noBeepersPresent()) {
      turnLeft();
      move();
      turnRight();
   }
}
```

Problem 2: Simple JavaScript expressions, statements, and functions (10 points)

(2a)

`5 % 4 - 4 % 5`	`-3`
`7 < 9 - 5 && 3 % 0 === 3`	`false`
`"B" + 3 * 4`	`"B12"`

(2b) `"cabbage"`

(2c)

JavaScript Console

```
To care is human!
```

Problem 3: Simple JavaScript programs (15 points)

```
/**
 * File: PythagoreanTriples.js
 * ---------------------------
 * This program lists all of the Pythagorean triples for a < b <= MAX
 * such that a^2 + b^2 = c^2 for integers a, b, and c.
 */

/* Constants */
const MAX = 25;

/* Main program */
function PythagoreanTriples() {
   for (let a = 1; a < MAX; a++) {
      for (let b = a + 1; b <= MAX; b++) {
         let sum = a * a + b * b;
         let c = Math.round(Math.sqrt(sum));
         if (c * c === sum) {
            console.log(a + ", " + b + ", " + c);
         }
      }
   }
}
```

Problem 4: Using graphics and animation (20 points)

```
/* Constants */
const GWINDOW_WIDTH = 500;
const GWINDOW_HEIGHT = 300;
const CROSSBAR_LENGTH = 60;
const CROSSBAR_BREADTH = 20;
const TIME_STEP = 20;
const CROSS_SPEED = 2;

/* Main program */
function RedCross() {
   let gw = GWindow(GWINDOW_WIDTH, GWINDOW_HEIGHT);
   let cross = createRedCross(CROSSBAR_LENGTH, CROSSBAR_BREADTH);
   gw.add(cross, gw.getWidth()/2, gw.getHeight()/2);
   let direction = randomReal(0, 360);
   let step = function() {
      cross.movePolar(CROSS_SPEED, direction);
   };
   setInterval(step, TIME_STEP); // return value can be ignored
   let clickAction = function(e) {
      if (gw.getElementAt(e.getX(), e.getY()) === null) return;
      direction = randomReal(0, 360);
   };
   gw.addEventListener("click", clickAction);
}

/**
 * Function: createRedCross
 * ------------------------
 * Constructs and returns a GCompound consisting of two
 * red rectangles--the first wide by narrow pixels in size, the
 * second narrow by wide pixels in size--such that their centers
 * overlap.
 */
function createRedCross(wide, narrow) {
   let cross = GCompound();
   cross.add(createFilledRectangle(wide, narrow, "Red"));
   cross.add(createFilledRectangle(narrow, wide, "Red"));
   return cross;
}

/**
 * Function: createFilledRectangle
 * -------------------------------
 * Constructs and returns a filled rectangle of the specified
 * width, height, and color (both border and fill).
 */
function createFilledRectangle(width, height, color) {
   let rect = GRect(-width/2, -height/2, width, height);
   rect.setFilled(true);
   rect.setColor(color);
   return rect;
}
```

Problem 5: Strings (15 points)

```
/**
 * Function: spoonerism
 * --------------------
 * Defines the spoonerism function according to the specifications
 * laid out in the first practice midterm.
 */
function spoonerism(phrase) {
   let sp1 = phrase.indexOf(' ');
   let sp2 = phrase.lastIndexOf(' ');
   let orig1 = phrase.substring(0, sp1);
   let orig2 = phrase.substring(sp2 + 1);
   let middle = phrase.substring(sp1, sp2 + 1);

   let vp1 = findFirstVowel(orig1);
   let vp2 = findFirstVowel(orig2);
   let transformed1 = orig2.substring(0, vp2) + orig1.substring(vp1);
   let transformed2 = orig1.substring(0, vp1) + orig2.substring(vp2);
   return transformed1 + middle + transformed2;
}

/**
 * Function: findFirstVowel
 * ------------------------
 * Returns the index of the first lowercase vowel, or -1 if no lowercase
 * vowel could be found.
 */
function findFirstVowel(str) {
   for (let i = 0; i < str.length; i++) {
      if (isEnglishVowel(str.charAt(i))) {
         return i;
      }
   }
}

/**
 * Function: isEnglishVowel
 * ------------------------
 * Returns true if and only if the provided string is of length 1, and
 * its one characters is a lowercase vowel.
 */
function isEnglishVowel(ch) {
   return ch.length === 1 && "aeiou".indexOf(ch) >= 0;
}
```