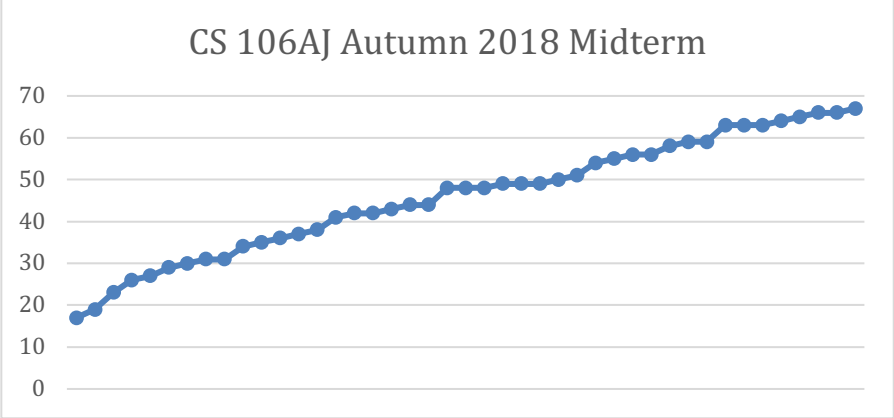
Jerry Cain
CS 106AJ

Handout #32S
November 5, 2018

Answers to Midterm Exam

The course staff spent several hours grading your midterms on Sunday afternoon and evening, so I'm happy to report they've been graded and will be returned during lecture on Monday. The exam was intended to be challenging, but many of you did brilliantly, and most of you did well enough that I'm happy to go with a traditional CS106AJ curve, where I set the median grade to sit at the A-/B+ border.

The complete histogram of grades is presented below, where each dot represents a single exam score, where lower scores are on the left and higher scores are on the right.



You can determine your letter grade by looking up your score in the following table:

Median = 48 (68.6%)
Mean = 45.9 (65.6%)

Range	Grade	N
65–70	A+	4
54–64	A	11
48–53	A–	8
41–47	B+	6
34–40	B	5
29–33	B–	4
26–28	C+	2
23–25	C	1
19–22	C–	1
15–18	D	1
00–12	NP	0

Solution 1: Karel the Robot (10 points)

```
/**
 * Function: TrickOrTreat
 * ----------------------
 * Solves the trick-or-treat problem outlined
 * in Problem 1.
 */
function TrickOrTreat() {
   sweepOneStreetSide();
   turnAround();
   sweepOneStreetSide();
}

/**
 * Function: sweepOneStreetSide
 * ----------------------------
 * Prompts Karel to walk the length of the street
 * and peeking in to every porch and lifting any
 * treats that have been left there.
 */
function sweepOneStreetSide() {
   while (frontIsClear()) {
      collectTreat();
      move();
   }
   collectTreat();
}

/**
 * Function: collectTreat
 * ----------------------
 * Prompts Karel to look to its right, and
 * if there's a porch, moves in to the porch,
 * collects any treats, and returns to the street.
 */
function collectTreat() {
   if (rightIsClear()) {
      turnRight();
      move();
      if (beepersPresent()) {
         pickBeeper();
      }
      turnAround();
      move();
      turnRight();
   }
}
```

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

(2a)

`3 + 2 * 2 - 15 % 5 * 100`	`7`
`"B" === "b" || "H" < "GGG"`	`false`
`20 + 7 + "1" + 8 + 4 * 7`	`"271828"`

(2b) `"mior"`

(2c) `"IGHOWEEN24"`

Note: The call to `toLowerCase` returns the lowercase version of the string, but it does not change the string itself, because JavaScript strings are immutable.

Solution 3: Simple JavaScript programs (15 points)

```
/**
 * Function: Narcissistic
 * ----------------------
 * Returns true if and only if the supplied number is
 * narcissistic, as outlined in the statement of Problem 3.
 */
function Narcissistic(number) {
   let len = countDigits(number);
   let original = number;
   let sum = 0;
   while (number > 0) {
      let digit = number % 10;
      sum += Math.pow(digit, len);
      number = Math.floor(number/10);
   }
   return sum == original;
}

/**
 * Function: countDigits
 * ---------------------
 * Returns the number of digits in the supplied number.
 */
function countDigits(number) {
   let count = 0;
   while (number > 0) {
      count++;
      number = Math.floor(number/10);
   }
   return count;
}
```

Solution 4: Using graphics and animation (20 points)

```
/**
 * Function: Stopwatch
 * -------------------
 * Draws a simple stopwatch that responds to mouse clicks.
 * The first mouse click starts the stopwatch, the second
 * stops it, the third starts it, and so forth.
 *
 * Note that my solution installs and clears a new timer
 * function with every pair of clicks.  It's perfectly fine
 * to use a Boolean variable along with a single timer that
 * remains installed forever.
 */
function Stopwatch() {
   let gw = GWindow(GWINDOW_WIDTH, GWINDOW_HEIGHT);
   let stopwatch = createStopwatch();
   gw.add(stopwatch, GWINDOW_WIDTH/2, GWINDOW_HEIGHT/2);
   let timer = 0;
   let clickAction = function(e) {
      if (timer === 0) {
         let step = function() {
            stopwatch.rotate(-DTHETA);
         };
         timer = setInterval(step, STEP_TIME);
      } else {
         clearInterval(timer);
         timer = 0;
      }
   };
   gw.addEventListener("click", clickAction);
}

/**
 * Function: createStopwatch
 * --------------------------
 * Constructs and returns a GCompound representing the stopwatch.
 * The reference point is taken to the watch face's center.
 */
function createStopwatch() {
   let stopwatch = GCompound();
   stopwatch.add(GOval(-STOPWATCH_RADIUS, -STOPWATCH_RADIUS,
                       STOPWATCH_DIAMETER, STOPWATCH_DIAMETER));
   stopwatch.add(GLine(0, 0, 0, -STOPWATCH_HANDLENGTH));
   return stopwatch;
}
```

Solution 5: Strings (15 points)

```
/**
 * Function: exchange
 * ------------------
 * Translates the supplied word into the equivalent word where all
 * neighboring vowels have been swapped, as outlined in Problem 5.
 */
function exchange(str) {
   let result = "";
   let curr = 0;
   while (true) {
      let first = findFirstVowel(str, curr);
      if (first === -1) {
         result += str.substring(curr);
         break;
      }
      let second = findFirstVowel(str, first + 1);
      if (second === -1) {
         result += str.substring(curr);
         break;
      }
      result +=
         str.substring(curr, first) + str.charAt(second) +
         str.substring(first + 1, second) + str.charAt(first);
      curr = second + 1;
   }
   return result;
}

/**
 * Funtion: findFirstVowel
 * -----------------------
 * Returns the index of the first vowel at or beyond the
 * supplied start position.  If start is omitted, then it's
 * assumed to be 0.
 */
function findFirstVowel(str, start) {
   if (start === undefined) start = 0;
   for (let i = start; i < str.length; i++) {
      if (isEnglishVowel(str.charAt(i))) {
         return i;
      }
   }
   return -1;
}

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