Eric Roberts and Jerry Cain
CS 106J

Handout #39S
May 17, 2017

Solution to Section #6

Portions of this handout by Eric Roberts

1. 2D Array Warmup

```
/*
 * Given a 2D array sheets and a rowNum, this returns
 * the sum of the row rowNum of sheets if the rowNum
 * is a valid row index of sheet. If not, an error string
 * is returned.
 */
function sumRow(sheet, rowNum) {
    var error = "Error: row index " + rowNum + " out of bounds!";
    if (rowNum >= sheet.length) return error;
    var sum = 0;
    for (var i = 0; i < sheet[rowNum].length; i++) {
        sum += sheet[rowNum][i];
    }
    return sum;
}

/*
 * Given a 2D array sheets and a colNum, this returns
 * the sum of the column colNum of sheets if the colNum
 * is a valid col index of sheet. If not, an error string
 * is returned.
 */
function sumCol(sheet, colNum) {
    var error = "Error: col index " + colNum + " out of bounds!";
    if (colNum >= sheet[0].length) return error;
    var sum = 0;
    for (var i = 0; i < sheet.length; i++) {
        sum += sheet[i][colNum];
    }
    return sum;
}

/*
 * Given a 2D array sheets, this returns the
 * the sum of all the elements in sheet.
 */
function sumAll(sheet) {
    var sum = 0;
    for (var i = 0; i < sheet.length; i++) {
        for (var j = 0; j < sheet[0].length; j++) {
            sum += sheet[i][j];
        }
    }
    return sum;
}
```

2. Flip Horizontal

```
/*
 * Takes as input a GImage and returns the
 * same image but flipped horizontally.
 */
function flipHorizontal(image) {
   var array = image.getPixelArray();
   var width = array[0].length;
   var height = array.length;
   for (var row = 0; row < height; row++) {
      for (var p1 = 0; p1 < width / 2; p1++) {
         var p2 = width - p1 - 1;
         var temp = array[row][p1];
         array[row][p1] = array[row][p2];
         array[row][p2] = temp;
       }
    }
    return GImage(array);
 }
```

4. How Prime (swapped places with problem 3 to save space on paper)

```
/*
 * Returns all the prime numbers between 2 and UPPER_LIMIT.
 */
function howPrime() {
    var primeNumbers = [];
    const UPPER_LIMIT = 1000;
    var crossedOut = createArray(UPPER_LIMIT + 1, false);
    for (var i = 2; i <= UPPER_LIMIT; i++) {
        if (!crossedOut[i]) {
            console.log(i + " is prime.");
            primeNumbers.push(i);
            for (var multiples = i; multiples <= UPPER_LIMIT;
multiples += i) {
                crossedOut[multiples] = true;
            }
        }
    }
    return primeNumbers;
}
```

3. Image scale

```
/*
 * Simple helper function that creates a 2D array of size
 * (rows, cols) where each element is initialized to value.
 */
function create2DArray(rows, cols, value) {
   var twoDArray = [];
   for (var i = 0; i < rows; i++) {
      var new_row = [];
      for (var j = 0; j < cols; j++) {
         new_row.push(value);
      }
      twoDArray.push(new_row);
   }
   return twoDArray;
}

/*
 * Takes as input a GImage and scale factors in the
 * x and y directions. Returns the same image but
 * scaled by the provided factors.
 */
function scale(image, x_scale, y_scale) {
   if (x_scale === 0 || y_scale === 0) return null;

   var array = image.getPixelArray();
   var new_width = array[0].length * x_scale;
   var new_height = array.length * y_scale;
   var new_array = create2DArray(new_height, new_width);

   for (var new_row = 0; new_row < new_height; new_row++) {
      for (var new_col = 0; new_col < new_width; new_col++) {
         var old_row = Math.floor(new_row/y_scale);
         var old_col = Math.floor(new_col/x_scale);
         new_array[new_row][new_col] = array[old_row][old_col];
      }
   }
   return GImage(new_array);
}
```