Eric Roberts
CS 106A

Handout #45A
May 14, 2012

Solutions to Section #6

1. Sudoku

```
/* Tests whether a Sudoku solution is valid */
   private boolean checkSudokuSolution(int[][] grid) {
      for (int i = 0; i < 9; i++) {
         if (!isValidGroupOfNine(getRow(grid, i))) return false;
         if (!isValidGroupOfNine(getColumn(grid, i))) return false;
         if (!isValidGroupOfNine(get3x3Square(grid, i))) return false;
      }
      return true;
   }

/*
 * Tests whether an array of nine values is valid for Sudoku, which means
 * that it contains no duplicate values or integers outside of the range
 * 1 to 9.
 */
   private boolean isValidGroupOfNine(int[] array) {
      boolean[] alreadySeen = new boolean[10];
      for (int i = 0; i < 9; i++) {
         int value = array[i];
         if (value < 1 || value > 9 || alreadySeen[value]) return false;
         alreadySeen[value] = true;
      }
      return true;
   }

/* Extracts row k from an 9x9 grid */
   private int[] getRow(int[][] grid, int k) {
      return grid[k];
   }

/* Extracts column k from an 9x9 grid */
   private int[] getColumn(int[][] grid, int k) {
      int[] column = new int[9];
      for (int i = 0; i < 9; i++) {
         column[i] = grid[i][k];
      }
      return column;
   }

/*
 * Extracts the elements of the kth 3x3 subsquare as an array of 9
 * elements.  The subsquares are numbered from left to right then
 * top to bottom starting with subsquare 0 in the upper left.
 */
   private int[] get3x3Square(int[][] grid, int k) {
      int baseRow = k / 3 * 3;
      int baseCol = k % 3 * 3;
      int[] array = new int[9];
      for (int i = 0; i < 9; i++) {
         array[i] = grid[baseRow + i / 3][baseCol + i % 3];
      }
      return array;
   }
```

2. Image processing

```
/**
 * Creates a new image which consists of the bits in the original
 * rotated left by 90 degrees.
 * @param image The original image
 * @return The image rotated left by 90 degrees
 */
  private GImage rotateLeft(GImage image) {
     int[][] oldPixels = image.getPixelArray();
     int width = oldPixels[0].length;
     int height = oldPixels.length;
     int[][] newPixels = new int[width][height];
     for (int i = 0; i < height; i++) {
        for (int j = 0; j < width; j++) {
           newPixels[width - j - 1][i] = oldPixels[i][j];
        }
     }
     return new GImage(newPixels);
  }
```