Eric Roberts
CS 106A

Handout #45
May 14, 2012

Section Handout #6—Multidimensional Arrays

1. Sudoku (Chapter 11, exercise 5, page 453)

In the last several years, a new logic puzzle called *Sudoku* has become quite popular throughout the world. In Sudoku, you start with a 9 x 9 grid of numbers in which some of the cells have been filled in with digits between 1 and 9. Your job in solving the puzzle is to fill in each of the empty spaces with a digit between 1 and 9 so that each digit appears exactly once in each row, each column, and each of the smaller 3×3 squares. A Sudoku puzzle is constructed so that there is only one solution. For example, given the puzzle shown on the left of the following diagram, the one solution is shown on the right:

		2	4		5	8		
	4	1	8				2	
6				7			3	9
2				3			9	6
		9	6		7	1		
1	7			5				3
9	6			8				1
	2				9	5	6	
		8	3		6	9		

3	9	2	4	6	5	8	1	7
7	4	1	8	9	3	6	2	5
6	8	5	2	7	1	4	3	9
2	5	4	1	3	8	7	9	6
8	3	9	6	2	7	1	5	4
1	7	6	9	5	4	2	8	3
9	6	7	5	8	2	3	4	1
4	2	3	7	1	9	5	6	8
5	1	8	3	4	6	9	7	2

Although the algorithmic strategies necessary to generate or solve Sudoku puzzles are beyond the scope of CS 106A, you can easily write a method that checks to see whether a proposed solution follows the Sudoku rules against duplicating values in a row, column, or outlined 3 x 3 square. Write a method

```
private boolean checkSudokuSolution(int[][] grid)
```

that performs this check and returns **`true`** if the grid is a valid solution.

2. Image processing (Chapter 11, exercise 13, page 459)

Write a method **`rotateLeft`** that takes a **`GImage`** and produces a new **`GImage`** in which the original has been rotated 90 degrees to the left. For example, if you started with the image of a candle stored in a file **`Candle.gif`** as shown on the left, calling **`rotateLeft`** should transform the image as follows: