Ozobot Editor—**Lesson: Two-Dimensional Arrays**—Teacher Guide

by Richard Born, Associate Professor Emeritus, Northern Illinois University

If you haven't already done so, you should read the *Two-Dimensional Array Student Guide*. Doing so will familiarize you with the many activities that will engage your students as they learn how to work with two-dimensional arrays. The *Student Guide* includes seven programs that take students from the basics to a very challenging program dealing with two-dimensional arrays. Some comments regarding each of these programs along with answers to questions posed for students in each of the investigations are covered in this Teacher's Guide.

Computer Science Programming Concepts in this Lesson

- The use of variables to hold the values of program constants
- Subroutines to break up code into logical segments
- The use of loops to handle tasks that must be repeated several times
- The use of conditionals, as needed, to control the flow of logic
- Subroutine and variable names that support self-documentation
- Nesting of structures (*e.g., if* inside a *loop*)

Program 1: Determining the Default Values for an Array Element (Share Code u2uf8n8}

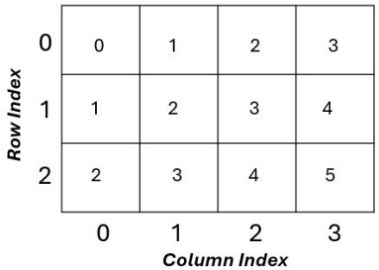
The students will determine that the default value for each element in the two-dimensional array is 0.

Program 2: Filling the Elements of the Array with the Number 5 (Share Code epmcn3)

This program shows students how they can change the default value of the elements in a two-dimensional array to any desired value. They learn how to accomplish this by the use of a double nested loop that cycles through the column elements within each row.

Program 3: Filling the Elements of the Array with a Mathematical Calculation (Share Code rhm58rm)

In this program, the student fills the array with elements representing the *sum* of the row and column index for each element.



Program 4: Filling the Elements of the Array with a Sequence of Values (Share Code yg7en74)

This program shows that the elements of an array can be filled with values that are based on a mathematical sequence of values such as the squares of integers from 1 through 12. A new variable is introduced that is initialized at the value 0 and is incremented by 1 for each element in the array.

Program 5: Rolling a Pair of Dice and Searching a Two-Dimensional Array (Share Code t9bqhim)

Here are typical results for 5 runs of the program, counting the number of 7's in each run:

Run #	1	2	3	4	5	Average
Number of 7's	2	3	2	1	3	2.2

Answer to Question 1:

2	3	4	5	6	7	8	9	10	11	12
(1, 1)	(1, 2)	(2, 2)	(1, 4)	(3, 3)	(3, 4)	(4, 4)	(4, 5)	(5, 5)	(5, 6)	(6, 6)
	(2, 1)	(1, 3)	(4, 1)	(4, 2)	(4, 3)	(5, 3)	(5, 4)	(4, 6)	(6, 5)	
		(3, 1)	(3, 2)	(2, 4)	(5, 2)	(3, 5)	(6, 3)	(6, 4)		
			(2, 3)	(5, 1)	(2, 5)	(6, 2)	(3, 6)			
				(1, 5)	(6, 1)	(2, 6)				
					(1, 6)					

There are a total 1+2+3+4+5+6+5+4+3+2+1 = 36 different outcomes, all equally likely. Of these, 6 outcomes result in a sum of 7. Therefore, the probability of getting a 7 is 6/36 or 1/6.

Answer to Question 2:

1/6 is the same as 2/12, and since our program rolls the dice 12 times, we would expect, on average, 2 of those times to result in 7.

Answer to Question 3:

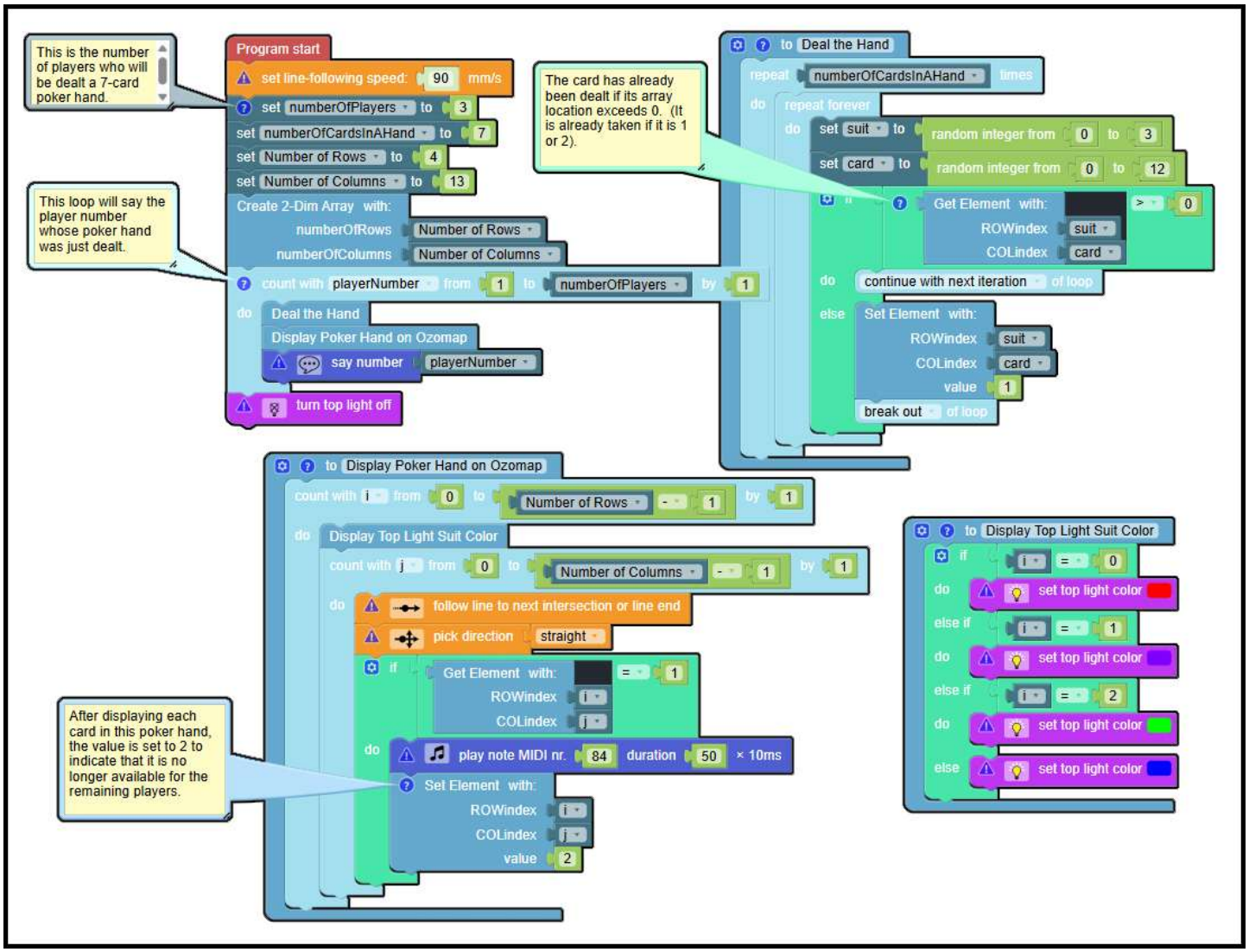
Our experimental average of 2.1 is not far off the theoretical average of 2. *[It might be interesting for the students to compute the average of the averages for all student groups. This **may** be closer to the theoretical value of 2.]*

Program 6: Creating a 3x3 Identity Matrix (4ssg898)

Here is the reason for stipulating that students are allowed to have **just one** "Set Element" block in their program. We want to avoid setting the main diagonal to 1's using a separate "Set Element" block for each element on the diagonal. Suppose that the square array was a 10 x 10 array. This would require ten "Set Element" blocks and would therefore be rather inefficient coding.

Program 7: Dealing a 7-Card Poker Hand (One Player Program Share Code audrxv9)

The solution to the challenge of dealing poker hands to more than one player is shown in the Blockly program below. Comments are included to document the changes made from the original program. ***(Multiple Player Program Challenge Share Code 7eoyih8)***



Answer to Question 1: The frequency of occurrence of the "continue with next iteration" block increase as the number of players increases.

Answer to Question 2: The program would run forever, or until you stopped it manually. Eight players implies that Ozobot would need to deal 8 x 7 = 56 cards, but the poker deck only contains 52 cards.

Applicable CSTA (Computer Science Teachers Association) Standards

2-CS-02	Design projects that combine hardware and software components to collect and exchange data.
2-AP-11	Create clearly named variables that represent different data types and perform operations on their values.
2-AP-12	Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.
2-AP-13	Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.
2-AP-19	Document programs in order to make them easier to follow, test, and debug.
3A-DA-11	Create interactive data visualizations using software tools to help others better understand real-world phenomena.
3A-AP-17	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3B-CS-02	Illustrate ways computing systems implement logic, input, and output through hardware components.
3B-AP-14	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.