**Random Map Generation with 2D Arrays**

The week's lab will introduce you to the use of multi-dimensional arrays. A 2D array of doubles, for example, is created as follows:

double[][] A = new double[4][5];

The array consists of 4 rows and 5 columns. Arrays in Java (as well as C) are stored "row-major" order, which means that the row number or y-coordinate comes before the column number or x-coordinate. Please visualize the coordinates in the correct orientation. An array variable such as A above is a pointer to a structure stored on the heap.

One can also think of a 2D array as a one-dimensional array with elements that are also pointers to one-dimensional arrays. It is perfectly valid to treat a 2D array this way. For example, the expression **A.length** will return 4, the length of the "outer" array. **A[0].length** will give 5, which is the length of the "inner" array.

Operations on 2D arrays usually will involve a nested loop, which iterates through each element of the array. The following code sets the value of every element of an array to 1:

int rows = A.length;

int cols = A[0].length;

for(int i=0;i<rows;i++)

for(int j=0;j<cols;j++)

{

A[i][j] = 1;

}

# 

# Step 1

For this assignment we will use a 2D array of integers to represent a map. One finds such maps in many computer games. Although the game maps may look fancier, the basic technique is the same. Each cell on the map represents a type of terrain: 0 = desert, 1 = water, 2 = forest

We want different types of terrain on the map. Try the following:

for(int i=0;i<rows;i++)

for(int j=0;j<cols;j++)

{

double r = Math.random(); // r is a number 0 to .9999...

if (r<0.33)

map[i][j] = DESERT;

else if (r<0.66)

map[i][j] = WATER;

else

map[i][j] = FOREST;

}

What does it look like?

# Step 2

A simple loop doesn’t produce a very interesting map. We want a map with patches of water and patches of forest. That is, the different types of terrain should be bunched together. The solution to this is to "grow" the map gradually. Let's first consider how to add water patches (lakes) to the map, which is initially set to all desert.

First, start with a very low probability, say 0.005 (½ percent). Going through the map array once, each cell will only have a very small chance of becoming water. We then apply the following algorithm for several successive "generations":

For each cell **map[i][j]**, look at all the cells that surrounds it: up, down, left, and right.

|  |  |  |
| --- | --- | --- |
|  | **map[i-1][j]** |  |
| **map[i][j-1]** | **map[i][j]** | **map[i][j+1]** |
|  | **map[i+1][j]** |  |

For every one of these neighboring cells that also has value water, you increase the probability that **map[i][j]** is also going to be water. If all four surrounding cells are water cells, then the probability that **map[i][j]** will be water should be very high. If we apply this algorithm for several generations (your program will consist of triple-nested loops), the water cells will grow into large patches. You will need to experiment with how many generations to run, and what the exact probability values are.

Most of all, you must be careful of the boundary conditions. Not all cells have four neighbors. You cannot examine **map[i-1][j]** without first checking that **i>0**. Similarly, you cannot refer to **map[i+1][j]** without first checking that **i<rows-1**. If you do not think about the boundary cells carefully, your program will crash with an "ArrayIndexOutOfBounds" exception.

Try creating patches of water.

What does it look like?

Try generating patches of forest. You may experiment with different approaches. For example, you may want the forests to be close to water.

What does it look like?

# Step 3

Take your map and run with it. What kind of interesting maps can you generate? Will you add extra types of terrain besides water, forest, and desert? Would this map be useful for a game board?

# Step 4

Share your map with the class and explain your algorithm. Animate your map creation by pausing after each generation of terrain creation to better illustrate your map development. Why did you choose this process?

# Rubric

|  |  |  |
| --- | --- | --- |
| Top Notch | The map has terrain areas that are bunched in a creative way. More than 3 terrains/colors are used. The student can explain his algorithm. | 90-100 |
| Meets Expectations | Like terrain is bunched together in a meaningful way. Student can show the program. | 80-89 |
| Basics Only | Meets minimum requirements | 70-79 |
| Incomplete | Doesn’t meet requirements. | below 70 |