CSC 4500, Fall 2018

HW4: Pyramid Visualizations

All of this first part is unchanged from HW1:

**Part I.** Shown below is a pyramid of integers. Think of this pyramid painted on a chalk board. You start a “game” by placing your finger at the 1. Then you roll a fair dice with exactly 4 sides, labeled UL, UR, DL, and DR. (Imagine that you have such a thing.) UL means “up left.” UR means “up right.” DL means “down left.” And DR means “down right.” This particular pyramid has 6 levels and 21 nodes.

1

2 3

4 5 6

7 8 9 10

11 12 13 14 15

16 17 18 19 20 21

Sometimes, you can’t make all the possible moves. For example, if you are at 1, you can’t go up. If you are at 19, you can’t go down. If you roll a direction you can’t move, you stay where you are in the pyramid, but that does count as a “move.”

There is another aspect to this game. Whenever you make a move, and when you start the game on the number 1, you put a dot next to the number where you are. You put a new dot on the number even if the “move” forces you to stay on the same number. You keep playing this strange “game” until every number has at least one dot. At that point, the game is over.

After the game is over, you will record the total number of moves it took to finish the game, the average number of dots on the numbers, and the largest number of dots on any number. More than one number may have exactly that number of dots. I don’t care which nodes have this maximum number of dots. You will collect all of this information for each game you simulate.

**Part 2.** All the same rules apply as HW1 and HW2 for submitting your program. One textfile, ending in “.py,” should hold your entire program. Name the program HW4lastname.py, where “lastname” is replace by your last name. My file would be HW4miller.py.

**Part 3.** When your program starts, you should print out to the screen an explanation of the game. Then you should show on the screen a graphical representation of the pyramid shown above (with 6 levels and 21 nodes), and then simulate and visualize a game playing out. Use one color to show nodes that have no dots, a different color to show the current node (where the “finger” resides at the moment), and different colors to show nodes that have a few dots, more dots, and so on. As the interactive viewer watches, s/he should be able to follow the action reasonably well; don’t make it run too fast or too slow.

After the game finishes, print out beneath your picture of the pyramid (now with colors showing the last state of the game), the number of moves it took to finish the game, the average number of dots, and the maximum number of dots. Annotate the numbers appropriately.

**Part 4.** Next, open an external file called HWinfile.txt. Inside that file there will be four rows with two integers in each row, separated by a blank. You may assume that the file is as described. (That is, no bullet proofing required.) The two integers on each line represent:

L. The number of levels to simulate, an integer from 2 to 25 inclusive.

T. Another integer, this one between 10 and 50 inclusive, that tells how many times you should simulate the pyramid game with this sized pyramid.

When you are finished reading in the four rows, you should have four ordered pairs, (L1, T1), (L2, T2), (L3, T3) and (L4, T4).

**Part 5.** Next your program should run four sets of simulation, one for each ordered pair. For example, for the first ordered pair, you should run T1 simulations of a game with L1 levels. When you do those simulations, store the average number of moves it took to complete the games over all T1 simulations; call that number M1. Also store the maximum number of dots over all T1 simulations; call that number D1. Do this for all four ordered pairs. When you have completed Part 5, you should have four vectors, each of them with four elements:

(L1,T1,M1,D1), (L2,T2,M2,D2), (L3,T3,M3,D3) and (L4,T4,M4,D4)

Print out a table showing all of these numbers, appropriately labeled.

**Part 6. (OPTIONAL)** It would be cool (though this is not required) to figure out a way to display the data you generated (those numbers in the table) in a way that is interesting for an interactive user. You might try graphing, for example. For graphing, you might ignore one of the columns in the table, making it easier to graph in two dimensions. You could do multiple graphs. This part of the assignment is extra for experts. No extra points if you do it, and no points off if you don’t. But I’ll be impressed if you do something interesting.

**Conclusion:** If you do this program well, it could be used to experiment with data to try to understand the pyramid game better. For example, as the number of levels increases, it seems reasonable that the average number of moves needed to finish increases. But is that increase linear, or geometric, or something else? Similarly, does the maximum number of dots on a single node increase in the same way as the number of moves, or is it a different way? When your program is running, use different data (in the HW4infile.txt) to try to answer this sort of question.

As usual, if you have questions, please email me at [millerkei@umsl.edu](mailto:millerkei@umsl.edu).

Keith