CS1020E: DATA STRUCTURES AND ALGORITHMS I

Tutorial 7 – Recursion

(Week 9, starting 10 October 2016)

1. *Simple Recursion*

The greatest common divisor (GCD) of two positive integers is the largest positive integer that divides the numbers without a remainder. For example, the GCD of 18 and 42 is 6.

Write a **recursive** function `gcd(int a, int b)`, which prints how Euclid's algorithm is used to derive the answer. For example, `gcd(5, 8)` returns `1`, but prints out:

```
(5, 8) 5 = 0 * 8 + 5
(8, 5) 8 = 1 * 5 + 3
(5, 3) 5 = 1 * 3 + 2
(3, 2) 3 = 1 * 2 + 1
(2, 1) 2 = 2 * 1 + 0
```

2. *Application of Recursion*

You have a 2D int array, in which each element contains a colour - 0 for white, 1 for light gray, 2 for dark gray. You have to implement the paint bucket **fill** functionality. If you don't know what this is, open paint and play around with the paint bucket tool ☻.

If you fill one element in the 2D array with a colour **c**, the surrounding cells are coloured till it reaches the border of the original colour. The colour spreads from the target cell to the 4 neighbouring cells (top left right bottom), the neighbours of the neighbours, and so on.

Implement the `fill(int** arr, ...)` function below. The **main algorithm must be recursive**. The user will only call the second function. Your second function can make use of the first function if you need to add more parameters.

```
void fill(int** arr, int numRows, int numCols,
        int currRow, int currCol, int newColour /*,one more param? */) {
    // can use this method to recurse if necessary
}
void fill(int** arr, int numRows, int numCols,
        int startRow, int startCol, int newColour) {
    // client will use this method
}
```

As an example, the PaintBucket tool is here being applied to row index 2, column index 3:

1	1	1	1	2	2	2	1
1	1	1	0	0	0	0	2
2	1	0	0	0	0	2	2
1	0	0	0	0	0	2	2
2	0	0	2	2	2	1	1
2	0	0	2	2	0	0	0
1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	2

1	1	1	1	2	2	2	1
1	1	1	2	2	2	2	2
2	1	2	2	2	2	2	2
1	2	2	2	2	2	2	2
2	2	2	2	2	2	1	1
2	2	2	2	2	0	0	0
1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	2

If you are banned from using recursion, what data structure(s) can help you accomplish something similar?

Design algo before coding
Find repeatable patterns
Draw out the recursive tree

If time is still permitting, Lab TA will discuss some of the Midterm Test questions.