CSCI 4041, Spring 2019, Quiz 5 (30 minutes, 20 points)

Name:

x500:

Discussion Start Time (**circle one**):   3:35   4:40   5:45   6:50   7:55   other:

---

1. (1 points each) True/False - Circle one. Note that when asking about the properties of an algorithm, we specifically mean the version of that algorithm discussed in lecture.

   True   False   Huffman Codes assign longer encodings to high frequency characters.

   True   False   Dynamic Programming algorithms generally run faster than their brute-force, recursive counterparts.

   True   False   If a problem can be solved optimally by either Dynamic Programming or a Greedy Algorithm, the Dynamic Programming solution will generally use more memory.

   True   False   For the activity selection problem, choosing the activity with the shortest duration is an optimal greedy choice.

   True   False   An adjacency-matrix is preferable to an adjacency-list for representing a sparse graph.

2. (5 points) Define the imbalance factor of a Red-Black Tree or B-Tree with distinct keys to be the absolute difference between the number of keys that are less than any key in the root node and the number of keys greater than any key in the root node. Draw the following:
   a. An 8-key Red-Black Tree with the minimum imbalance factor
   b. An 8-key Red-Black Tree with the maximum imbalance factor
   c. An 8-key B-Tree with t = 3 that has the minimum imbalance factor
   d. An 8-key B-Tree with t = 3 that has the maximum imbalance factor

3. You are playing a board game on an infinitely long board in which each turn, you either move 2, 3, or 5 spaces forward, depending on a die roll. You want to know how many different ways there are to reach the n^{th} space on the board (here moving forward 3 and then 5 is counted a different ‘way’ to reach the 8th space than moving 5 and then 3). Here is a recursive function to solve the problem (assume indexing starting at 1):

```
Ways(n)
      baseCases = [0,1,1,1,3]
      if n <= 5
            return baseCases[n]
      else
            return Ways(n-2) + Ways(n-3) + Ways(n-5)
```

a. (2 points) Draw the recursion tree for Ways(10).

b. (2 point) What is the height and maximum branching factor of the recursion tree for Ways(n), in terms of n? Assume n > 5.

c. (3 points) Write a dynamic programming algorithm in pseudocode, Python, or Java to solve the above problem in Theta(n) time.

d. (3 points) Argue that this problem has both the overlapping subproblems and (optimal) substructure properties.