# CS5463 Fundamentals of Software

## Homework 3

Due 10/30/20 before 11:59pm (Central Time)

### 1. Arrays and Heaps (6 points)

(1) (3 points) Suppose you wanted to create an algorithm to find the number of elements $\geq k$ in an array, $A$, of $n$ distinct elements.

Specifically, describe efficient algorithms to solve this problem when $A$ is the following data structures.

(a) Unsorted array
(b) Sorted array
(c) Max-heap

(2) (3 points) Analyze the runtime of your three algorithms from the previous part:

(a) Unsorted array
(b) Sorted array
(c) Max-heap

### 2. Huffman Encoding (2 points)

(1) Show the process by which a Huffman tree would be built for the following string of character: (i.e., redraw the tree after each new node is created).

"cannercancanacanra"

(2) Using your tree, find the encoding that would result for the above string.

### 3. Red-Black Trees (2 points)

(1) Company X has created a new variant on red-black trees which also uses blue as a color for the nodes. They call these "red-black-blue trees". Below are the new rules for these trees:

- Every node is red, blue, or black.
- The root is black.
- Every leaf (NIL) is black.
- If a node is red, then both its children are black.
- If a node is blue, then both its children are red or black.
- For each node, all simple paths from the node to descendant leaves contain the same number of black nodes.

(a) (2 points) In class we found that the height, $h$, of a red-black tree is $\leq 2\log_2(n+1)$ (where $n$ is the number of keys). Find and prove that a similar bound on height of the red-black-blue trees.

(**Hint:** You can use the same approach as we did to show

$h \leq 2\log_2(n+1)$).

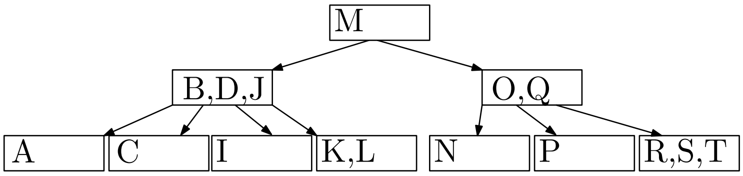
(b) (0 points - just for fun) Adding an additional color didn't seem to improve our bound on $h$ (i.e., 3 colors allows the tree to become more unbalanced than with 2 colors). What benefit might we get from the extra color?

## 4. B-trees (4 points)

(1) (2 points) Show the results of inserting the keys

$E, F, G, U, V, W, H$

in order into the B-tree shown below. Assume this B-tree has minimum degree $k = 2$. Draw only the configurations of the tree just before some node(s) must split, and also draw the final configuration.




(2) (2 points) Suppose you have a B-tree of height $h$ and minimum degree $k$. What is the largest number of keys that can be stored in such a B-tree? Prove that your answer is correct.

(**Hint:** Your answer should depend on $k$ and $h$. This is similar to theorem we proved in the B-tree notes).

## 5. Hash Table Probabilities (3 points)

(1) (1 point) Suppose 2 keys are inserted into an empty hash table with $m$ slots. Assuming simple uniform hashing, what is the probability of:
(a) exactly 0 collisions occurring
(b) exactly 1 collisions occurring

(2) (2 points) Suppose 3 keys are inserted into an empty hash table with $m$ slots. Assuming simple uniform hashing, what is the probability of:
(a) exactly 0 collisions occurring
(b) exactly 1 collisions occurring
(c) exactly 2 collisions occurring

## 6. Hash Table (7 points)

(1) Consider inserting the keys $2, 21, 3, 58, 11, 42, 34$ into a hash table of length $m = 10$ with the hash function $h(k) = k \bmod 10$.

(a) (2 points) Illustrate the result of inserting these keys using linear probing to resolve collisions.
(b) (2 points) Illustrate the result of inserting these keys using chaining to resolve collisions.

(2) Consider inserting the keys $8, 5, 14$ into a hash table of length $m = 8$ with the hash function $h(k) = \lfloor m(kA - \lfloor kA \rfloor) \rfloor$ where $A = 0.625$.
(a) (2 points) Illustrate the result of inserting these keys.
(b) (1 point) Now compute the hash function of the key 14 using the alternative algorithm described below.

You can assume we have a word size $w = 4$. Since $m = 8 = 2^3$, $p = 3$. Since $A = 0.625 = 10/2^4 = 10/2^w$, $s = 10$.

**Alternative Algorithm:** Compute $ks$ and convert it to a binary number. This number will consist of $\leq 2w$ bits. Look at the rightmost $w$ bits. Of those bits, convert the leftmost $p$ bits back to an integer. This integer is your hash table slot.