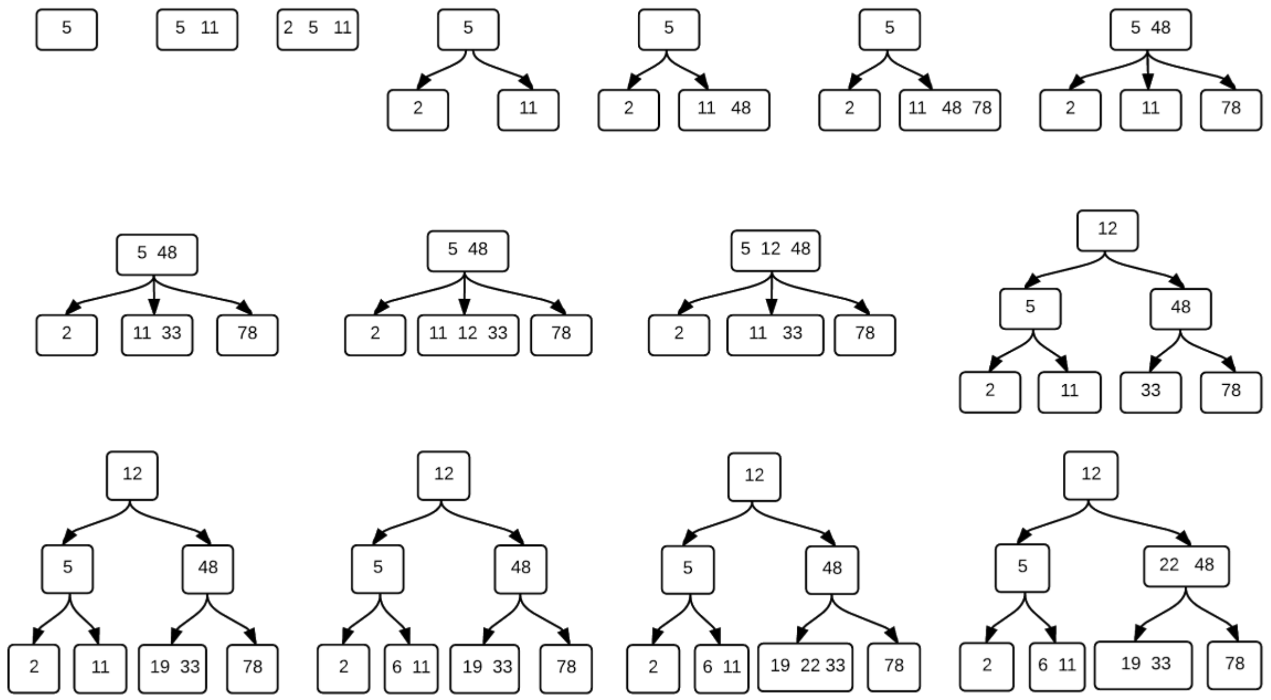
Shuni Li
Comp221 HW6
Professor Shilad Sen
April 11, 2016

1. (10 pts) For the list of numbers below, draw a series of diagrams that show the 2-3 trees you would get by inserting these values into a tree one after the other, starting with an empty tree. You must draw a new tree diagram every time a node splitting takes place.

5 11 2 48 78 33 12 19 6 22



2. (10 pts) Think about other ways of representing priority queues.

   a. Suppose that you were going to use an ordered array to implement a priority queue, where you would insert values in priority order. Discuss the worst-case efficiency class of insertion, deleting the largest, and accessing the largest, given this implementation.

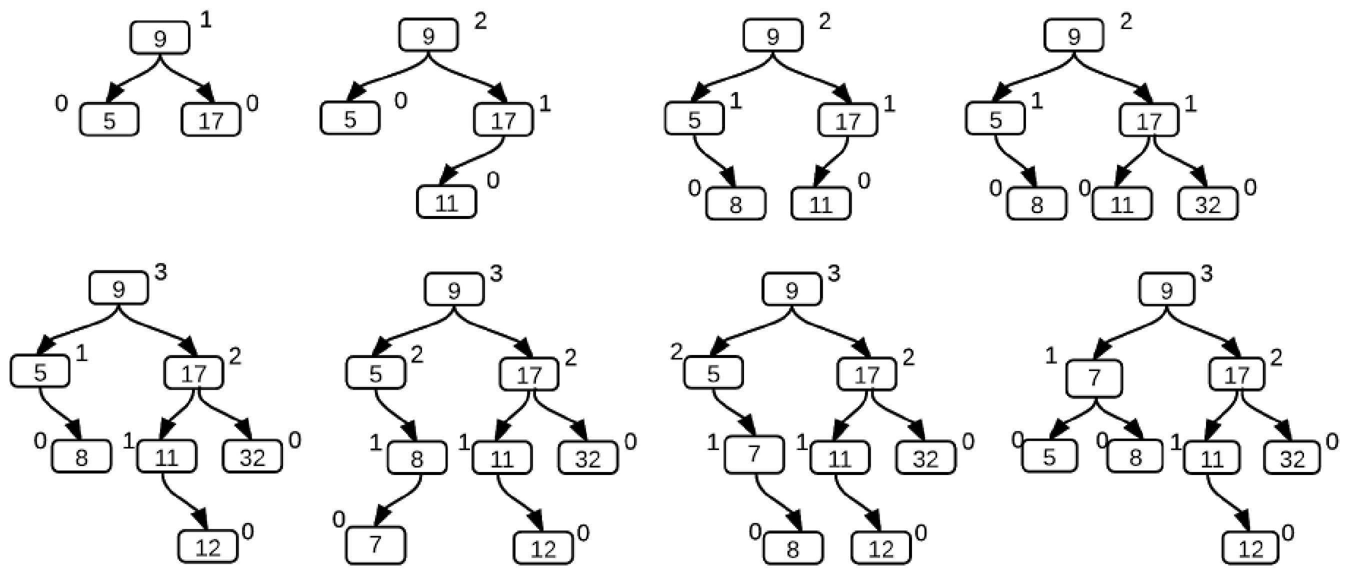
      The worst-case of insertion happens when the element we are inserting has the highest priority. In this case, we have $\Theta(n)$ time complexity because we are adding the element to the front of the array and we need to increase every element's index by 1.
      Deleting the largest is a $\Theta(n)$ operation because we are deleting from the front of the array and we need to move every element one position forward.
      Accessing the largest is just $\Theta(1)$ because we can access the element with index 0.

b. Suppose for some reason you decided to use an AVL tree to represent a priority queue. Discuss how you would implement operations to find the largest and delete the largest, and what the worst-case efficiency class of insertion, deleting the largest, and accessing the largest would be.

Find the largest: start from the root, keep going to the right child of the current node until we reach a leaf. That leaf is the largest element.
Delete the largest: use the method above to find the largest and set that node to null.

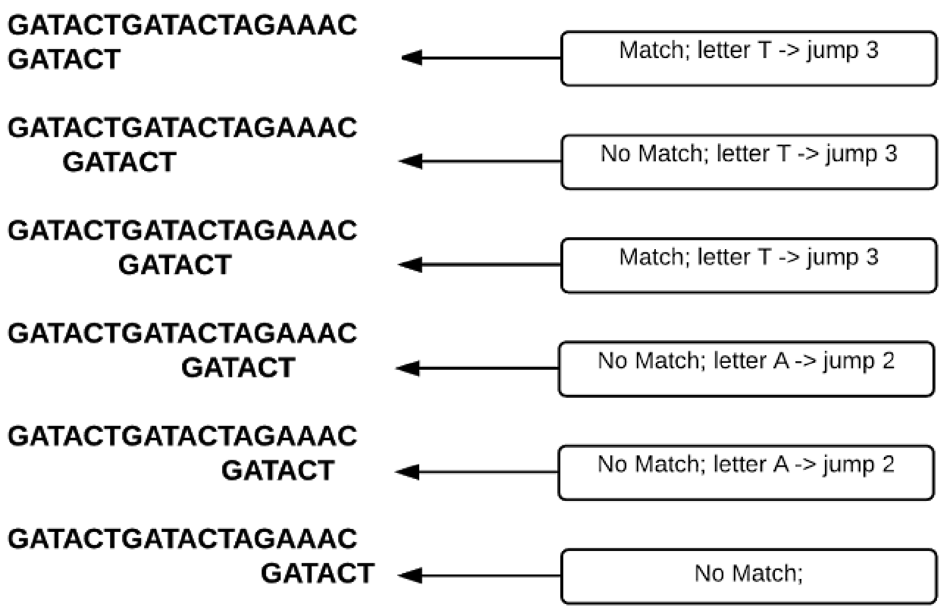
The worst-case efficiency class of insertion is $\Theta(log\ n)$. Deleting the largest is $\Theta(log\ n)$. Accessing the largest is also $\Theta(log\ n)$.

3. (10 points) Insert the following sequence of numbers into an AVL tree. Show your steps, and annotate each node with the height of its subtree. [9, 17, 5, 11, 8, 32, 12, 7]



4. (10 points) Given the text "GATACTGATACTAGAAAC" and a pattern "GATACT", show the Horspool "bad character shift" table for the pattern, and apply it to the text. Show your steps. Your work should calculate how many times the pattern applies; not just return true or false.

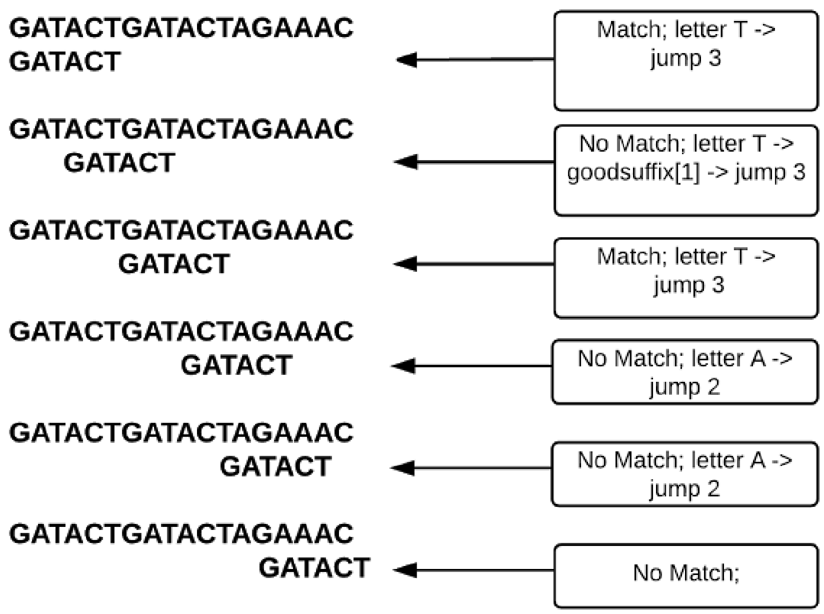
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
2	6	1	6	6	6	5	6	6	6	6	6	6	6	6	6	6	6	6	3	6	6	6	6	6	6



There are 2 matches!

5. (10 points) For the same pattern and text, above show the Boyer-Moore "good suffix" shift table. Show the results of applying the Boyer-Moore algorithm (which will use both tables) to the pattern and text.

k	pattern	d
1	GATACT	3
2	GATACT	6
3	GATACT	6
4	GATACT	6
5	GATACT	6


GATACTGATACTAGAAAC
GATACT
Match; letter T -> jump 3
GATACTGATACTAGAAAC
GATACT
No Match; letter T -> goodsuffix[1] -> jump 3
GATACTGATACTAGAAAC
GATACT
Match; letter T -> jump 3
GATACTGATACTAGAAAC
GATACT
No Match; letter A -> jump 2
GATACTGATACTAGAAAC
GATACT
No Match; letter A -> jump 2
GATACTGATACTAGAAAC
GATACT
No Match;