1. (10 Points) Use heap sort to sort the following input array into non-decreasing order. Ignore array element at index 0.

Answer:

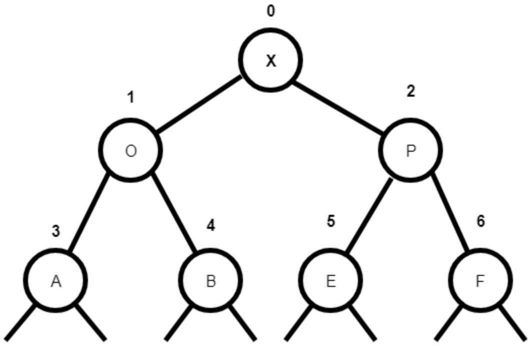
Index	0	1	2	3	4	5	6	7	8	9	10
Original Array	-1	533	342	714	130	541	10	346	931	57	217
After heap construction	-1	931	541	714	342	533	10	346	130	57	217
After Swap & Sink	-1	714	541	346	342	533	10	217	130	57	931
After Swap & Sink	-1	541	533	346	342	57	10	217	130	714	931
After Swap & Sink	-1	533	342	346	130	57	10	217	541	714	931
After Swap & Sink	-1	346	342	217	130	57	10	533	541	714	931
After Swap & Sink	-1	342	130	217	10	57	346	533	541	714	931
After Swap & Sink	-1	217	130	57	10	342	346	533	541	714	931
After Swap & Sink	-1	130	10	57	217	342	346	533	541	714	931
After Swap & Sink	-1	57	10	130	217	342	346	533	541	714	931
After Swap & Sink	-1	10	57	130	217	342	346	533	541	714	931

2. (10 Points) The input integer array has N entries, which are all non-negative and have equal keys. What is the running time and space complexity for the following sorting algorithms? Assume each array entry has at most D digits and each digit is within 0 and K.

Answer:

	Running Time	Space Complexity
Merge Sort	O(NlgN)	O(N)
Quick Sort	O(NlgN)	O(lgN)
Heap Sort	O(N)	O(1)
Count sort	Not enough information to answer this	Not enough information to answer this
Radix sort (uses count sort)	O(D*(N+K))	O(N+K)

3. (6 Points) A binary heap is 0-based, where root is at index 0 and its left child is at index 1 and its right child is at index 2, and so forth. The following figure shows an example with 7 nodes. If a node is at index k, provide the indexes for its parents and children.

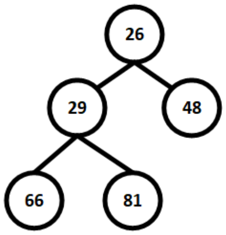


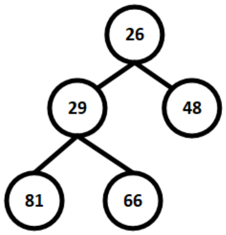
Answer:

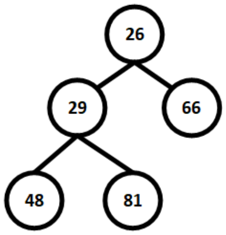
Node Index	k (k>0)
Parent Index	floor((k-1)/2))
Left Child Index	2k+1
Right Child Index	2k+2

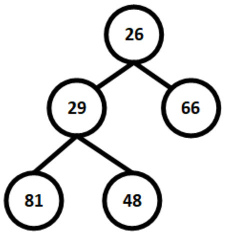
4. (4 Points) Draw all of the different binary min-heaps that can be made from the five keys 26, 48, 81, 29, 66.

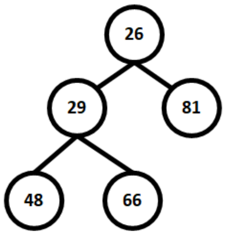
Answer:

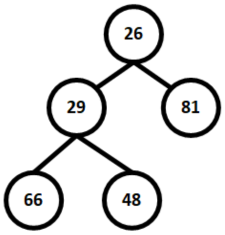


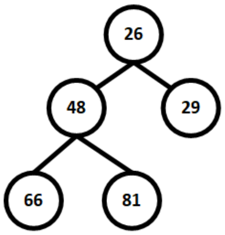


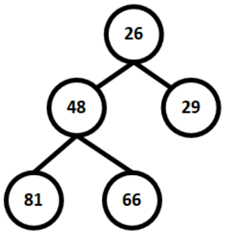












5. (10 Points) Indicate each of the following statements is true or false.

a) For a binary MinPQ, the maximum key must be at one of the leaves.

b) There are exactly ceiling(n/2) leaves in a binary priority queue with n nodes.

c) An array in decreasing order is a min-oriented heap.

d) In MaxPQ, suppose that a client calls insert() with an item that is larger than all items in the queue, and then immediately calls delMax(). The resulting heap might not be identical to the heap as it was before these operations.

e) If a node in a BST has two children, its successor has no left child and its predecessor has no right child.

Answer:

a) True, b) True, c) False, d) False, e) True

6. (5 Points) Suppose that a BST has keys that are integers between 1 and 10, and we search for 5. Which sequence(s) below cannot be the sequence of keys examined?

a) 9, 2, 4, 0, 5

b) 6, 1, 2, 4, 5

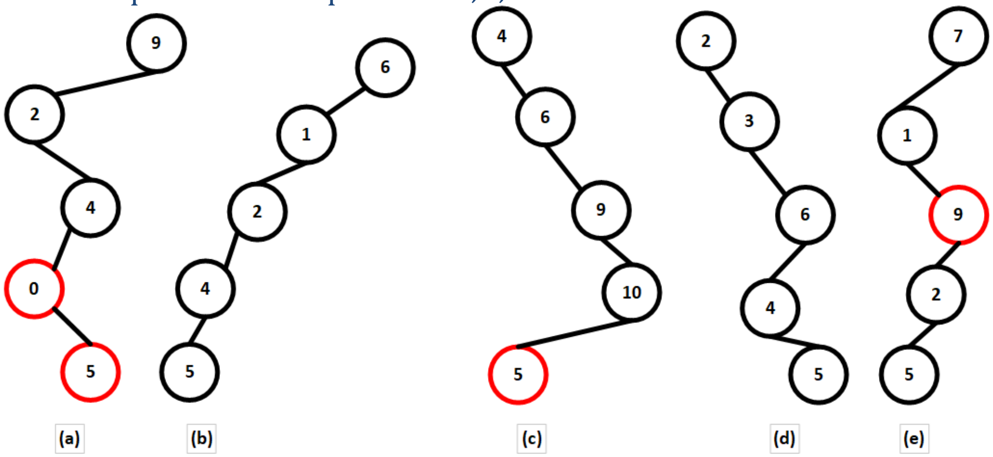
c) 4, 6, 9, 10, 5

d) 2, 3, 6, 4, 5

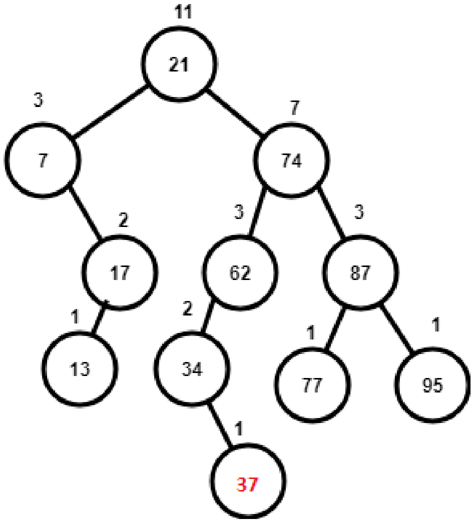
e) 7, 1, 9, 2, 5

Answer:

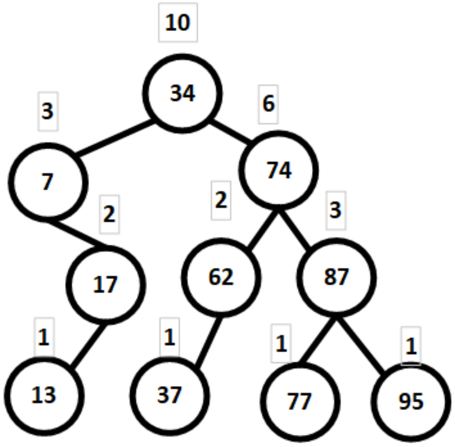
These sequences are impossible: a, c, and e.



7. (5 Points) The following figure shows a BST, where key is displayed inside the node and n value is shown on top of each node. Using the BST delete algorithm provided in class, show the result tree after deleting the root from the following tree. The result tree should show key and n values for each node.



Answer:



8. (10 Points) Write a recursive method isBinaryTree() that takes a Node as argument and returns true if the subtree count field n is consistent in the data structure rooted at that node, false otherwise. Provide your implementation inside **Node** class **isBinaryTree** method.

Answer:

```
public static <Key extends Comparable<Key>, Value> boolean isBinaryTree(Node<Key, Value> node) {
       if (node == null) {
              return true;
       } else if ( size(node) != size(node.left) + size(node.right) + 1) {
              return false;
       } else {
              return isBinaryTree(node.left) && isBinaryTree(node.right);
       }
}

private static <Key extends Comparable<Key>, Value> int size(Node<Key, Value> node) {
       return (node == null ? 0 : node.n);
}
```

Submission Note

1) For written part of the questions:
   a. Write your answers inside a text document (in plain text, MS Word, or PDF format)
   b. Name the file as firstname.lastname.assignment3.txt(doc, docx, or pdf) with proper file extension
2) For programming part of the questions
   a. Use JDK 1.8 and Junit5
   b. Put your full name at the beginning of every source file you created or modified. **2 points will be deducted if your names are not included in the source files.**
   c. Do not change the provided package, class, or method name. You can add extra classes or methods if they are needed.
   d. **If your code does not compile, you will get zero point**.
   e. Use the provided tests to verify your implementation. **Extra tests might be used for grading.**
   f. Zip all the source files into firstname.lastname.assignment3.zip
3) Submit both of your files (text document and zip file) via Canvas course web site.
2) Due Oct 14, 11:59 PM