Study Questions for Midterm 2, Data Structures & Algorithms

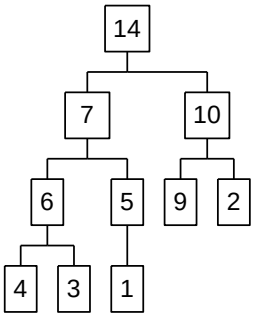
You are encouraged to discuss questions and solutions with your classmates or others. I have no solutions to provide but can answer questions you may have.

1. Heaps

Given the binary heap below:

a) Show the heap after inserting an element with the value of 8.

b) Show the heap after extracting the max value (use the original heap, not after inserting 8)



c) Give pseudocode for a recursive algorithm that determines whether or not a binary tree is a max-heap.

d) A d-ary heap is like a binary heap, but (with one possible exception) non-leaf nodes have d children instead of 2 children.

   a. Describe how would you represent a d-ary heap in an array in terms of organizing parents and children.

   b. What is the height of a d-ary heap of n elements in terms of n and d?

2. Python Heap Implementation

Here is code to implement a heap of Person objects using their age as the key value:

```
class Person:
    def __init__(self,name = "Bobo", age = 0):
        self.name = name
        self.age = age

class Heap:
    def __init__(self):
        self.data = []

    # Hard-coded to only work with Heaps of Person objects
    def heapify(self, i):
        largest = i    # Initialize largest as root
        left = 2*i + 1   # Formula for 0-based array
        right = 2*i + 2
        heapSize = len(self.data)

        if (left < heapSize and self.data[left].age >
self.data[largest].age):
            largest = left
        if (right < heapSize and self.data[right].age >
self.data[largest].age):
            largest = right

        if (largest != i):
            temp = self.data[i]
            self.data[i] = self.data[largest]
            self.data[largest] = temp
            self.heapify(largest)

    def buildHeap(self, personAry):
        self.data = copy.deepcopy(personAry)
        for i in range(len(self.data) - 1, -1, -1):  # i-1 down to 0
            self.heapify(i)
```

Here is pseudocode to insert a new key into a heap implemented as array A:

Insert(A, key)
    Add key to end of array A
    i = index of new key
    while i>0 and A[$\lfloor i/2 \rfloor$] < key
        A[i] = A[$\lfloor i/2 \rfloor$]
        i = $\lfloor i/2 \rfloor$
    A[i] = key

Write the python code to implement the insert function such that it works with the given heap implementation to insert a new Person object into the heap.

3. Tries

a. Suppose you want to store a dictionary of words. What are the pros and cons of using a trie vs. using a sorted list with binary search to look up a word? Be specific in terms of the runtime.

b. Consider a trie that holds a dictionary of words, like we discussed in the notes. Give pseudocode for an algorithm that given a prefix, prints out all words that start with that prefix.

4. Short Answer

a. What are the minimum and maximum number of elements in a heap of height h?

b. Where in a max-heap might the smallest element reside?

c. Under what conditions would it be more efficient to use a linked list to store all of the child nodes for a trie rather than store the child nodes in an array?

d. You have an unsorted array of n elements. You want to find the k smallest elements, where k is much smaller than n. Describe an efficient way to find these k elements.

5. Sorting

Below are the contents of an array as two algorithms are sorting it. Each line does not necessarily represent the next step in the algorithm, but only some later step during the course of the algorithm's execution. Key elements are depicted in **bold**. Identify each algorithm.

Algorithm 1

12	39	2	94	23	77	52	9
12	**39**	2	94	23	77	52	9
2	12	39	94	23	77	52	9
2	12	39	**94**	23	77	52	9
2	12	**23**	39	94	77	52	9
2	12	23	39	**77**	94	52	9
2	12	23	39	**52**	77	94	9
2	**9**	12	23	39	52	77	94

Algorithm 2

12	39	2	94	23	77	52	9
94	**39**	**77**	**12**	**23**	**2**	**52**	**9**
77	39	52	12	23	2	9	**94**
52	39	9	12	23	2	**77**	94
39	23	9	12	2	**52**	77	94
23	12	9	2	**39**	52	77	94
12	2	9	**23**	39	52	77	94
9	2	**12**	23	39	52	77	94
2	**9**	12	23	39	52	77	94