CMSC351 Summer 2022 Exam 2 Solutions

NAME:

UID:

Instructions: Please do all problems on the pages and in the spaces provided. This exam will be scanned into Gradescope and if your answers are not in the correct locations they will not be found or graded!

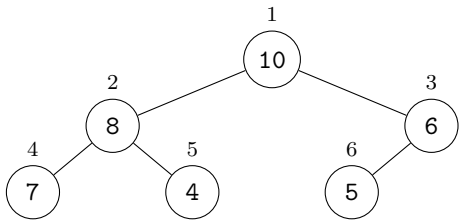
Extra Credit (Up to 2 Points): Write a haiku related to algorithms. A haiku is a short poem with three lines. The first line has 5 syllables, the second line has 7 syllables, the third line has 5 syllables.

THIS PAGE INTENTIONALLY LEFT BLANK! DO NOT USE!

1. Here is the critical code needed for Heap Sort: [10 pts]

```
function heapsort(A,n)
    converttomaxheap(A,n)
    for i = n down to 2
        swap(A[1],A[i])
        maxheapify(TREE,1,i-1) aka floatkeydown(TREE,1,i-1)
    end
end
```

Consider the following max heap:

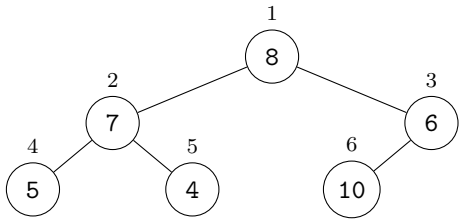


The above code is run on this heap. Note that `converttomaxheap` doesn't do anything because we already have a max heap.

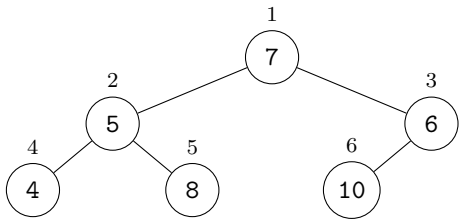
Draw the resulting heap after each `for` loop iteration ends. You can continue on the next page if needed.

Solution (Continue on Next Page if Needed):

After $i = 6$:

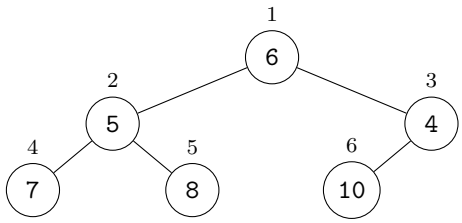


After $i = 5$:

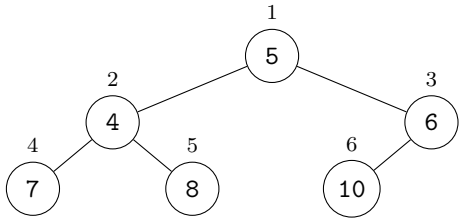


Previous Solution Continued if Needed. Otherwise Leave Blank:

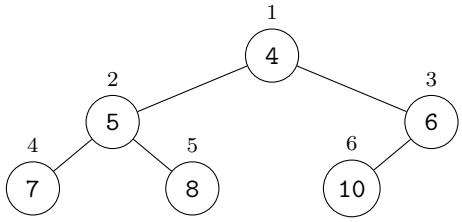
After $i = 4$:



After $i = 3$:



After $i = 2$:



2. Prove that: [10 pts]

$$\frac{1}{2}n(\lg n - 1) = \mathcal{O}(n \lg n)$$

Solution:

This is easiest with the limit theorem. Observe:

$$\begin{aligned}
\lim_{n\to\infty} \frac{\frac{1}{2}n(\lg n - 1)}{n \lg n} &= \lim_{n\to\infty} \frac{n \lg n - n}{2n \lg n} \\
&= \lim_{n\to\infty} \left[\frac{1}{2} - \frac{1}{s \lg n}\right] \\
&= \frac{1}{2} - 0 \\
&= \frac{1}{2}
\end{aligned}$$

Since this is not ∞, we have our conclusion.

3. Explain why it is impossible to construct a list `A` such that one single run of the partition routine of Quick Sort would result in the following: [10 pts]

```
A = [6,4,2,1,3,5]
```

Solution:

There is no element which has the property that every element to the left is smaller than it and every element to the right is larger than it.

4. Here is the high-level code for Radix Sort: [5 pts]

```
for i = 1 to d
    stable sort A using digit i
end
```

Radix Sort is applied to the following list:

A=[472,501,537,311,532,417,900]

Show the resulting list after each iteration of the `for` loop.

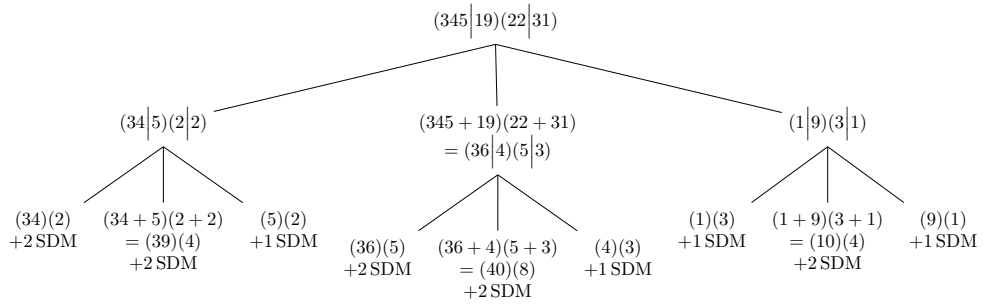
IMPORTANT: You may not need all the rows below, use only those you need!

Solution:

Iteration	A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]
1	900	501	311	472	532	537	417
2	900	501	311	417	532	537	472
3	311	417	472	501	532	537	900
4							
5							
6							
7							

5. Draw a tree diagram for Karatsuba's Algorithm applied to (34519)(2231) and use it to count the number of single-digit multiplications. [10 pts]

Solution:



This is a total of 14 SDMs.

6. The list A contains n integers between 0 and $10^{(3n^2)} - 1$ inclusive, each represented in decimal. What is the time complexity of RadixSort+CountingSort applied to such a list? [10 pts]

Solution:

We have $max = k = 1$. To represent numbers up to $10^{(3n^2)} - 1$ in decimal requires $d = 3n^2$ digits. Thus we have $\Theta(3n^2(n+9)) = \Theta(n^3)$.

7. Prove that if any list with n elements can be sorted with $f(n)$ comparisons then any list with $n+1$ elements can be sorted with $f(n)+n$ comparisons. [15 pts]

Solution:

First sort the first n using $f(n)$ comparisons.

Suppose the sorted list is $[b_1, ..., b_n]$ and suppose b is the final element that needs to be integrated into the list.

Ask the following series of questions:

- Is $b < b_1$? If so, $[b, b_1, ..., b_n]$.
- If not, is $b < b_2$? If so, $[b_1, b, b_2, ..., b_n]$.
- Continue up to:
- If not, is $b < b_n$? If so, $[b_1, b_2, ..., b, b_n]$.
- If not, $[b_1, , ..., b_n, b]$.

This is n more comparisons.

8. Merge Sort is applied to a list of length $n = 2^k$ for $k \geq 1$. For any i it takes $6i + 1$ seconds to merge two lists of length i and nothing else takes any time at all. Write down and simplify an expression for the amount of time it will take to sort such a list. [15 pts]

Solution:

Observe that:

- To get the final 1 list we merge 2 lists of length $n/2$. So that's $1[6(n/2) + 1] = 3n + 1$ seconds.
- For each of those 2 we merge 2 lists of length $n/4$. So that's $2[6(n/4) + 1] = 3n + 2$ seconds.
- For each of those 4 we merge 2 lists of length $n/8$. So that's $4[6(n/8) + 1] = 3n + 4$ seconds.
- ... this continues until ...
- For each of those 2^{k-1} we merge 2 lists of length $n/2^k$. So that's $2^{k-1}[6(n/2^k) + 1] = 3n + 2^{k-1}$ seconds.

The total is then the sum:

$$\begin{aligned}\sum_{i=0}^{k-1}(3n + 2^i) &= \sum_{i=0}^{k-1} 3n + \sum_{i=0}^{k-1} 2^i \\ &= 3n(k) + 2^k - 1 \\ &= 3n \lg n + n - 1\end{aligned}$$

9. In determining the MOM of a list containing n distinct elements with $n = 10k + 3$ for some positive integer k. We proceed as usual, arranging the list into columns of 5 elements and one column of 3 elements. Suppose that the MOM is taken from one of the columns of 5 elements. [15 pts]

   Show that the MOM is less than or equal to $0.3n$ elements. Note that you will need two cases.

   Solution:

   Note: This was awkwardly phrased as it should have really said "less than or equal to *at least* $0.3n$ elements" but it looks like everyone understood that.

   Observe that there are $2k$ columns with 5 elements and 1 column with 3 elements. Also note that $n = 10k + 3$ so $10k = n - 3$ so $k = 0.1(n - 3) = 0.1n - 0.3$.

   Either the MOM is less than the median of the 3-element column or it is not.

   - If it is, then MOM is less than or equal to at least $3k + 2$ elements and: $3k + 2 = 3(0.1n - 0.3) + 2 = 0.3n - 0.9 + 2 = 0.3n + 1.9$ so it's certainly less than $0.3n$ elements.
   - If it is not, then MOM is less than or equal to at least $3(k+1)$ elemenets and $3(k+1) = 3(0.1n - 0.3 + 1) = 0.3n - 0.9 + 3 = 0.3n + 2.1$ so it's certainly less than $0.3n$ elements.