COMP 3270

Homework 3

100 points

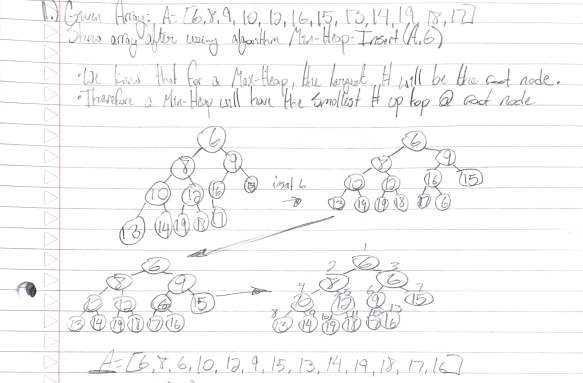
**Please submit using Canvas by 11:59PM on Thursday, October 20th, 2022**

Instructions:

1. This is an individual assignment. You should do your own work. Any evidence of copying will result in a zero grade and additional penalties/actions.
2. Late submissions **will not** be accepted unless prior permission has been granted or there is a valid and verifiable excuse.
3. Think carefully; formulate your answers, and then write them out concisely using English, logic, mathematics and pseudocode (no programming language syntax).
4. Type your final answers in this Word document.
5. Don’t turn in handwritten answers with scribbling, cross-outs, erasures, etc. If an answer is unreadable, it will earn zero points. **Neatly and cleanly handwritten submissions are acceptable**.

**1. (5 points)** Heapsort

Show the array A after the algorithm Min-Heap-Insert(A, 6) operates on the Min Heap implemented in array A=[6, 8, 9, 10, 12, 16, 15, 13, 14, 19, 18, 17]. In order to solve this problem you have to do some of the thinking assignment on the Ch.6 lecture slides. But you do not have to submit your solutions to those thinking assignments. Use your solutions to determine the answer to this question and provide the array A below.

A****

**2. (5 points)** Let A be a collection of objects. Describe an efficient O(nlgn) algorithm for converting A into a set. That is, remove all duplicates from A.

Text, letter

Description automatically generated

**3****. (5 points)** Given a sequence of numbers, S, the mode is the value that appears the most number of times in this sequence. Give an efficient O(nlgn) algorithm to compute the mode for a sequence of n numbers.

If we sort the array using mergesort, we have an algorithm that computes at O(nlogn) time. Once we have a sorted list, we can then go through the list and for each element, have a counter that keeps track of how many times a set number is duplicated as duplicate numbers will occur right after one another. We will then return the number that has the greatest value for its counter.

**4. (5 points)** Show that any comparison-based sorting algorithm can be made to be stable, without affecting the asymptotic running time of this algorithm. Hint: Change the way elements are compared with each other.

If we take our elements say x1 and x2 and place them in a pair system like (x1, 1) and (x2, 2) we now have a system that will be comparing two parameters rather than a single comparison that would permit x1 = x2 to be a valid input since this will allow us to compare the two pairs.

**5. (22 points)** Quicksort

(a) (6 points)

Quicksort can be modified to obtain an elegant and efficient linear (O(n)) algorithm QuickSelect for the selection problem.

Quickselect(A, p, r, k)

{p & r – starting and ending indexes; to find k-th smallest number in non-empty array A; 1≤k≤(r-p+1)}

1 if p=r then return A[p]

else

2 q=Partition(A,p,r) {Partition is the algorithm discussed in class}

3 pivotDistance=q-p+1

4 if k=pivotDistance then

5 return A[q]

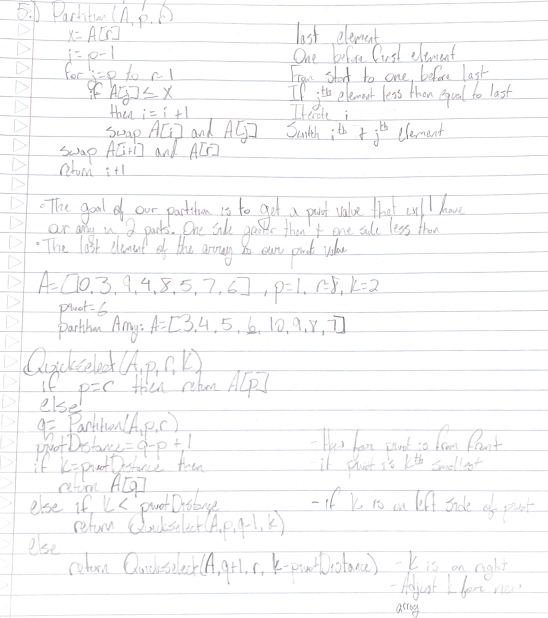
6 else if k<pivotDistance then

7 return Quickselect(A,p,q─1,k)

else

8 return Quickselect(A,q+1,r, k-pivotDistance)

Draw the recursion tree of this algorithm for inputs A=[10, 3, 9, 4, 8, 5, 7, 6], p=1, r=8, k=2. At each non-base case node show all of the following: (1) values of all parameters: input array A, p, r & k; (2) A after Partition. At each base case node show values of all parameters: input array A, p, r & k. Beside each downward arrow connecting a parent execution to a child recursive execution, show the value returned upwards by the child execution.



Text, letter

Description automatically generated

(b) (16 points). This algorithm has two base cases.

Explain what the first base case that the algorithm checks for is, in plain English:

The first base case checks to see if the input array that has been passed is a one element or empty array.

List the steps that the algorithm will execute if the input happens to be this base case:

Complete the recurrence relation using actual constants:

T(first base case) = 7

Explain what the second base case that the algorithm checks for is, in plain English:

The second base case will check if our kth smallest value is the pivot value that was selected in the partition part of the algorithm.

List the steps that the algorithm will execute if the input happens to be this base case:

2 q=Partition(A,p,r) {Partition is the algorithm discussed in class}

3 pivotDistance=q-p+1

4 if k=pivotDistance then

5 return A[q]

Complete the recurrence relation using actual constants (assume complexity of Partition to be 20n):

T(second base case) = 20n + 16

List the steps that the algorithm will execute if the input is not a base case:

2 q=Partition(A,p,r) {Partition is the algorithm discussed in class}

3 pivotDistance=q-p+1

6 else if k<pivotDistance then

7 return Quickselect(A,p,q─1,k)

else

8 return Quickselect(A,q+1,r, k-pivotDistance)

Text, letter

Description automatically generated

**6. (10 points)** Counting Sort

Show the B and C arrays after Counting Sort finishes on the array A [19, 6, 10, 7, 16, 17, 13, 14, 12, 9] if the input range is 0-19.

**Text, letter

Description automatically generated**

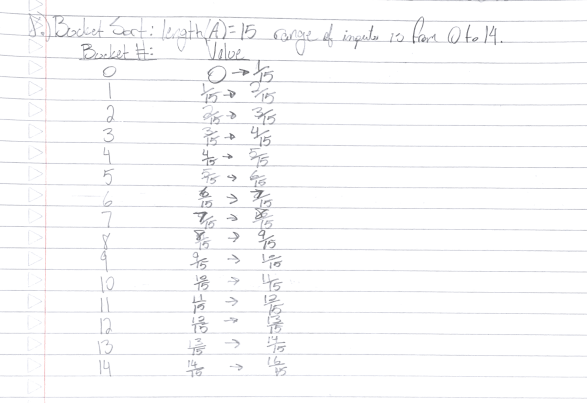
**7. (5 points)** Radix Sort

If Radix Sort is applied to the array of numbers [4567, 3210, 2345, 4321, 5678], show how these numbers will get rearranged after each of the four passes of the algorithm.

A picture containing text, document

Description automatically generated

**8. (12 points)** Bucket Sort

Consider the algorithm in the lecture slides. If length(A)=15 then list the range of input numbers that will go to each of the buckets 0…14.

Now generalize your answer. If length(A)=n then list the range of input numbers that will go to buckets 0,1,…(n-2), (n-1).

Bucket0: 0 -> 1/n

Bucket1: 1/n -> 2/n

Bucket(n-2): (n-2) -> (n-1)/n

Bucket(n-1): (n-1) -> 0

**9.** **(2 points)** Is the bucket-sort algorithm in-place? Why or why not?

It is not an in place algorithm due to the buckets requiring extra space outside of the array in order to store its values which are the array elements.

**10.** **(3 points)** Suppose we are given a sequence of n elements, each of which is an integer in the range [0,n2-1]. Describe a simple method for sorting in O(n) time.Hint: Think of alternate ways of viewing the elements.

**11. (10** points**)** Disjoint Set

Assume a Disjoint Set data structure has initially 20 data items with each in its own disjoint set (one-node tree). Show the final result (only show the array P for parts a, b & c below; no need to draw the trees) of the following sequence of unions (the parameters of the unions specified in this question are data elements; so assume that the find operation without path compression is applied to the parameters to determine the sets to be merged): union(16,17), union(18,16), union(19,18), union(20,19), union(3,4), union(3,5), union(3,6), union(3,10), union(3,11), union(3,12), union(3,13), union(14,15), union(14,3), union(1,2), union(1,7), union(8,9), union(1,8), union(1,3), union(1,20) when the unions are:

**A picture containing text, document, linedrawing

Description automatically generated**

**A picture containing text, document

Description automatically generated**

**Diagram

Description automatically generated**

**Letter

Description automatically generated**

**12. (16 points)** Binomial Queue

First show the Binomial Queue that results from merging the two BQs below. Then show the result of an Extract\_Max operation on the merged BQ. There may be more than one correct answer.

A piece of paper with writing

Description automatically generated with medium confidence