COMP 3270 Assignment 4 5 problems 50 points 10% Credit

Due before 11:59 PM Wednesday November 14

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. Enter your answers in this Word file. Submissions must be uploaded **as a single file** (Word or PDF preferred, but other formats acceptable as long as your work is LEGIBLE) to Canvas before the due date and time. Don't turn in photos of illegible sheets. **If an answer is unreadable, it will earn zero points.** Cleanly handwritten submissions (print out this assignment and write answers in the space provided, with additional sheets used if needed) scanned in as PDF and uploaded to Canvas are acceptable.
3. **Submissions by email or late submissions (even by minutes) will receive a zero grade.** No makeup will be offered unless prior permission to skip the assignment has been granted, or there is a valid and verifiable excuse.
4. Think carefully; formulate your answers, and then write them out concisely using English, logic, mathematics and pseudocode (no programming language syntax).

1. (15 points) **Binary Heap**

Max-Heap-Increase-Key(A[1...n]: array of number, i: int 1≤i≤n, key)

```
1  if key < A[i]
2    then print "new key is smaller than current key"
3  A[i] = key
4  parent = floor(i/2)
5  while i > 1 and A[parent] < A[i]
6         temp = A[i]
7         A[i]= A[parent]
8         A[parent] = temp
9         i = parent
10        parent = floor(i/2)
```

Show that the complexity of this algorithm is $O(\log_2 n)=O(\lg n)$ by developing and stating its T(n) in which the largest n-term is a lgn term. Do this by filling in the table and blanks below. Some entries are pre-filled. Cost of the floor operation = 1

Step#	Cost of single execution	Exact # of times executed	Total cost of this step = column 1 * column 2
1	5	1	5
2	1	1	1
3	4	1	4
4	4	1	4
5	10	At most _lgn+1 (or $\log_2 n$+1)_ times	10lgn+10
6	4	At most __lgn___ times	4lgn
7	6	At most __ lgn ___ times	6lgn
8	4	At most ___ lgn __ times	4lgn

9	2	At most __ lgn ___ times	2lgn
10	4	At most __ lgn ___ times	4lgn

0.5 point for each table cell (except the highlighted cells). Total 26 table cells to fill = 13 points. The values in column 2 (single execution cost) should be the same as or close to the provided answers to earn points. The values in column 3 (# of times executed) should be exactly the same as the provided answers to earn points. The values in column 4 (total cost) will earn points if those are correct products of the corresponding table cells in columns 2 & 3 even if the values in those cells were incorrect and did not earn points.

Sum the last column and simplify to obtain T(n) < ______30lgn+24___________

1 point for a "lgn" term and 1 point for a constant term in T(n). The actual coefficients do not matter.

2. (14 points) **Quick Sort**

Come up with an input of size 7 that will:

(a) produce the best case partitions in every recursive call of Quick Sort based on the Quick Sort and Partition algorithms that are given in the lecture slides.

A=

Smallest and **third smallest** numbers among the 7 numbers must be in the first two array cells in any order	**Second smallest** number must be in the array cell with index 3	**Sixth smallest** number must be in the array cell with index 4	**Fifth** and **seventh smallest** (i.e. largest) numbers among the 7 numbers must be in array cells with index 5 & 6 in any order	**Fourth smallest** number must be in the array cell with index 7

The best case partitions are those that divide the array equally. This means that the median should be picked as the pivot in every recursive execution. Since the Partition algorithm selects the last number in the array as the pivot and does a swap in its very last step, with a seven sized input array, the overall median should be in array cell 7, the median of the smallest three numbers in the input should be in array cell 3 and the median of the largest three numbers in the input should be in array cell 4 to guarantee that every recursive execution will pick the median of the input array/subarray as the pivot. To see this clearly, draw the recursion tree when A=[1, 3, 2, 6, 5, 7, 4]

1 point for each correct number in the array. To be correct the number must satisfy the above stated constraints

(b) produce the worst case partitions in every recursive call of Quick Sort based on the Quick Sort and Partition algorithms that are given in the lecture slides.

Worst case partitions are those in which either the left or the right partition is empty. This means that in every recursive execution, the pivot picked should be the largest/smallest number in the input array/subarray. There are several correct answers. The simplest ones are an input array that is already sorted in the ascending order, e.g., A=[1, 2, 3, 4, 5, 6, 7], or one in which all numbers are the same, A=[1, 1, 1, 1, 1, 1, 1]. This will make the right partition an empty array in every recursive execution in the recursion tree. Another example is A=[2, 3, 4, 5, 6, 7, 1]. This will make the left partition an empty array in every recursive execution in the recursion tree.

1 point for each correct number in the array. Verify that the answer is an array that is already sorted, or with all numbers identical, or such that each of the five non-base case recursive calls in the recursion tree will have one partition empty and the other subarrays of sizes 6, 5, 4, 3, and 2 respectively.

3. (5 points) **Counting Sort**

The Counting Sort algorithm can be used to sort integers in the range i-j, i<j and i>0 by pre-processing the input array A so that the algorithm can be applied to it as is with no modifications and then post-process the output array B to recover the original input in the sorted order. Explain in English what this will entail:

(a) What is the pre-processing on A that can be done so that the algorithm can work with no modifications?

Subtract i from each number in A

2 points

(b) What is the value of k in this case (the algorithm requires prior knowledge of the input range 0-k)?

k=(j-i)

2 points

(c) What is the post-processing on B that can be done so that the algorithm can work with no modifications?

Add i to each number in array B

1 point

4. (7 points) **Radix Sort**

If Radix Sort is used to sort an array of words alphabetically, and the input array is A=

CATS	BATS	BITS	PINE	DIG< >	BORE	DIM< >

show the array after each pass of the outer loop of the algorithm completes. < > is a single blank character that is used to pad words with less than 4 characters and it appears before the letter A in alphabetic ordering.

A after the first execution of the loop=

DIG< >	DIM< >	PINE	BORE	CATS	BATS	BITS

A after the second execution of the loop=

DIG< >	DIM< >	PINE	BORE	CATS	BATS	BITS

A after the third execution of the loop=

CATS	BATS	DIG< >	DIM< >	PINE	BITS	BORE

A after the fourth execution of the loop=

BATS	BITS	BORE	CATS	DIG< >	DIM< >	PINE

Each correct word in the four arrays gets 0.25 point = 28 entries, 7 points

5. (9 points) **Bucket Sort**

If length(A)=10 then numbers in the input array in the range [0,0.1) will all go to bucket 0, numbers in the input array in the range [0.1,0.2) will all go to bucket 1, numbers in the input array in the range [0.2,0.3) will all go to bucket 2, numbers in the input array in the range [0.3,0.4) will all go to bucket 3, numbers in the input array in the range [0.4,0.5) will all go to bucket 4, numbers in the input array in the range [0.5,0.6) will all go to bucket 5, numbers in the input array in the range [0.6,0.7) will all go to bucket 6, numbers in the input array in the range [0.7,0.8) will all go to bucket 7, numbers in the input array in the range [0.8,0.9) will all go to bucket 8, and numbers in the input array in the range [0.9,1.0) will all go to bucket 9. If length(A)=9 then list the range of input numbers that will go to buckets 0…8. State your answers with two decimal digit precision. 1 point for each

Numbers in the input array in the range [____0,0.11___) will all go to bucket 0
Numbers in the input array in the range [____0.11,0.22___) will all go to bucket 1
Numbers in the input array in the range [____0.22,0.33___) will all go to bucket 2
Numbers in the input array in the range [____0.33,0.44___) will all go to bucket 3
Numbers in the input array in the range [____0.44,0.55___) will all go to bucket 4
Numbers in the input array in the range [____0.55,0.66___) will all go to bucket 5
Numbers in the input array in the range [____0.66,0.77___) will all go to bucket 6
Numbers in the input array in the range [____0.77,0.88___) will all go to bucket 7
Numbers in the input array in the range [____0.88,1.0___) will all go to bucket 8