COM S 311 Exam 1

Name	Recitation Time

Problem	Max Points	Score
1 (big-O)	10	
2 (Runtime for Loops)	15	
3 (Binary Search)	20	
4 (Heap)	15	
5 (Hash)	15	
6 (Graph 1)	10	
7 (Graph 2)	15	
Extra Credit	15	
Total	100	

- **When asked to design a [illegible] uage and not library operati[illegible]**
- Proof of correctness [illegible]

- For all algorithm design problems, part of the grade depends on the runtime. Level of points for solutions related to design and analysis algorithm

```
if designed algorithm is correct and efficient as expected then
   if proof of correctness is unambiguous and justifiable then
      if run-time analysis shows derivation steps and is correct then
           points = 100%
      else
           points = 75%
   else if run-time analysis shows derivation steps and is correct then
           points = 75%
         else
           points = 50%
else if designed algorithm is correct and brute force then
   points = 30%
else if designed algorithm is incorrect then
   points = 0--20% (at the discretion of grader)
else if answer is "DO NOT GRADE"
   points = 15%
else
   points = 0%
```

You may use following algorithms, runtimes and their correctness proofs as blackbox

1. Binary search in sorted arrays, sorting algorithms.
2. Construction of Heaps/Hash tables and operations associated with them.
3. For the purpose of this exam, you may assume that time taken for hash table operations (add, remove, search) is $O(1)$ in worst-case.
4. Merge two sorted arrays.
5. DFS and BFS.
6. Reversing a directed graph.
7. Determining whether a graph is bi-partite
8. Computing connected components of an undirected graph.

Your application/use of these algorithms must match input/output behaviors. If you modify any of the algorithms you must state the modifications unambiguously and write the pseudocode.

Some Useful equalities

- $\sum_{i=1}^{n} i = n(n+1)/2$
- $\sum_{i=1}^{n} i^2 = n(n+1)(2n+1)/6$
- $2^{\log_2 n} = n, \quad a^{\log_b n} \quad \log_b a \quad n/2 \quad n \quad a$
- $1 + 1/r + 1/r^2 + 1$
- $a + ar + ar^2 + \cdots + ar \quad = \frac{}{(r-1)}$
- $1 + 2 + 4 + \cdots + 2^n = 2^{n+1} - 1$

1. Prove or disprove the following

   (a) $6 + 49x - 1001x^2 + x^3 \ \in \ O(x^3)$

   (b) $(log(n))^3 \ \in \ O(log(n^3))$

2. Derive the runtime for the following.

   (a)
```
for (i = 1; i < n; i++) {
 for (j = n; j >= i+1; j--) {
    for (k = 1; k < j + 1; k++) {
      <do some constant time operations>
    }
 }
}
```

   (b)
```
i = log(n)
while (i > 0) {
 for (j = 1; j <= exp(2, i); j++) {
   <do some constant time operations>
 }
 i--;
}
```

   In the above, `log(n)` denotes $log_2(n)$ and `exp(2, i)` denotes 2^i. You can assume that these operations are computable in constant time (though not correct).

3. We know that we can search for an element in a sorted array of n elements in O(log n). Now consider the situation where input (ascending order) sorted array is rotated by someone without your knowledge. For instance, a sorted a

   one time to f

   two times to f

   three times to form 51 53 0 17 33

   four times to form 53 0 17 33 51

   five times to get the original ordering (if there are rotating n times will result in the original array).

   You are asked to design an efficient algorithm to search an element in such a (ascending order) sorted and rotated array, where you do not know how far the array has been rotated. You can assume that the array given to you does not contain any duplicates. State the run-time of your algorithm.

4. This question has multiple parts.

   (a) What is the time to find the smallest element in a min heap?

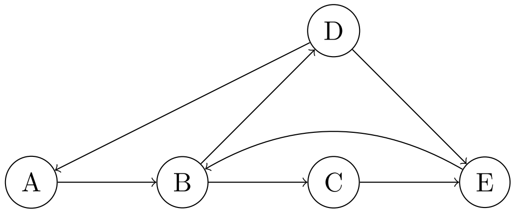
   (b) True or False: The last element in a min-heap is the largest element.

   (c) True or False: The first element in a min-heap is the smallest element.

   (d) Given an array A of integers (indexed from 1), give an algorithm that checks if the array is a binary min-heap or not. Derive the run-time of your algorithm.

5. Given two integers arrays A and B, we say that A is a permutation of array B, if the we can obtain array A by re-arranging elements of array B. Write an algorithm that takes two integer arrays as input and returns true if one is a permutation of the other. You may assume that every element of A differs from every other element of A, and similarly every element of B differs from every other element of B. Derive the run-time of your algorithm.

6. Consider the following graph:



Suppose we start doing DFS starting at vertex A. Write the start and end times for each of the vertices. If at any point of your exploration, you need to choose between multiple unexplored/undiscovered vertices, proceed by choosing the unexplored vertices in alphabetical order.

7. Design an algorithm that takes as input a directed graph G, and three pair-wise distinct vertices x, y and v ($x \neq y$, $x \neq v$ and $y \neq v$) and returns true if v is present in some shortest path from x to y. Prove the correctness of your algorithm. Derive the run-time of your algorithm.

* **Extra Credit** The city council is deciding the location of a state-of-art emergency response center. By location, the council considers cross-sections in the city. The objective is to locate the center at a cross-section from where almost all locations in the city can be reached *quickly*. Fortunately, one of the council members was a computer scientist. He modeled the major location/cross-sections of the city as vertices and the connections between cross-sections as edges recording the allowed direction of traffic from one cross-section to the other. Then each location is associated with an integer that quantifies the measure of suitability of the location for the emergency response center:

d

to y|

Higher suitability value for a cross-section indicates higher chanc -

gency response center. Design an algorithm to compute suitabilit me

of your algorithm.