ECE GY 9343 Midterm EXAM (2022 Fall)

Name: **NetID:**

Session(Circle one): Session A (Prof. Yong Liu) Session B (Prof. Pei Liu) Online

---

Answer ALL questions. Exam is closed book. No electronic aids. However, you are permitted two cheat sheets, two sides each sheet. Any content on the cheat sheet is permitted.

Multiple choice questions may have **multiple** correct answers. You will get **partial credits** if you only select a subset of correct answers, and will get zero point if you select one or more wrong answers.

Requirements for in-person students ONLY

Please answer all questions on the question book. You should have enough space. Since we scan all the submissions in a batch, DON'T write on the back of the pages and don't tear off any page.

Make sure you write your NetID on the bottom of each page (not your N number).

If you need extra scrape papers, we can give to you. However, it will not be graded.

Requirements for remote students ONLY

Each student is required to open Zoom and turn on their video camera, making sure the camera captures your hands and your computer screen/keyboard. The whole exam will be recorded. To clearly capture the video of exam taking, it is recommended that: A student can use an external webcam connected to the computer, or use another device (smartphone/tablet/laptop with power plugged in). Adjust the position of the camera so that it clearly captures the keyboard, screen and both hands. During the exam, you should keep your video on all the time. If there is anything wrong with your Zoom connection, please reconnect ASAP. If you cannot, please email ASAP, or call your proctor.

During the exam, if you need to use the restroom, please send a message using the chat function in Zoom, so we know you left.

Please use a separate page for each question. Clearly write the question numbers on top of the pages. When you submit, please order your pages by the question numbers.

Once exam is finished, please submit a single PDF on Gradescope, under the assignment named Midterm Exam. DEADLINE for submission is 15 minutes after the exam ended. Please remember to parse your uploaded PDF file. Before you leave the exam, its your responsibility to make sure all your answers are uploaded. If you have technical difficulty and cannot upload 10 minutes after exam ended, email a copy to your proctor and the professor.

1. (**20 points**) **True or False**

   (a) **T or F:** Counting sort is NOT in-place;

   (b) **T or F:** There exists a comparison sort of 5 numbers that uses at most 6 comparisons in the worst case;

   (c) **T or F:** For any asymptotically positive function $f(n)$, $f(n) + o(f(n)) = \Theta(f(n))$;

   (d) **T or F:** For any array with n elements, one can always find the i-th smallest element within $O(n)$ time;

   (e) **T or F:** By using uniform hashing functions, the number of elements hashed into a slot is independent from the number of elements in any other slot;

   (f) **T or F:** If chaining is used to handle collisions, the worst case time to insert a key into a hash table is $\Theta(1)$.

   (g) **T or F:** Quicksort with Hoare's partition is stable;

   (h) **T or F:** The complexity of building a max-heap with n nodes is $\Theta(n \log n)$;

   (i) **T or F:** Radix sort follows the divide-and-conquer design;

   (j) **T or F:** The worst-case running time and average-case running time are equal asymptotically (same order) for any randomized algorithms;

2. (**4 points**) Which of the following sorting algorithms are in-place:

   (a) MergeSort;

   (b) QuickSort;

   (c) HeapSort;

   (d) InsertionSort;

   (e) None of the above

3. (**4 points**) If $f(n) = \Omega(g(n))$, and $z(n) = \omega(g(n))$, which of the following can be true?

   (a) $f(n) = O(g(n))$;

   (b) $f(n) = \omega(z(n))$;

   (c) $g(n) = \Theta(z(n))$;

   (d) $f(n) = o(g(n))$;

   (e) $f(n) = \Theta(z(n))$;

4. (**4 points**) Which of the following statements regarding Divide-and-Conquer algorithms are true?

   (a) In quick-sort, dividing the problem into sub-problems is more complex than combining solutions to the sub-problems;

   (b) Divide-and-Conquer algorithms always run faster than iterative algorithms;

   (c) Heapsort is a divide-and-conquer algorithm;

   (d) In the maximum-subarray problem, combining solutions to the sub-problems is more complex than dividing the problem into sub-problems;

   (e) None of the above.

5. (**4 points**) Given a max-heap with height 10, if all the elements are distinct, the 4-th largest element can be at height of:

   (a) 6; (b) 1; (c) 7; (d) 9; (e) none.

6. (**4 points**) Which of the following statements about hash tables are true?

(a) When handling collisions using chaining, searching for a non-existing key always takes $\Theta(n)$;

(b) With universal hash functions, a random hash function is picked each time when inserting a new key into the table;

(c) Hash tables are preferred over direct-address tables due to possible reduction in storage requirement;

(d) Search for the maximum key always takes $\Theta(n)$;

(e) None of the above

7. (**6 points**) Prove the following properties of asymptotic notations:

(a) (**3 points**) $n^3 = \omega(n^2)$, where $\omega(\cdot)$ stands for little-ω;

(b) (**3 points**) If $f(n) = \Omega(g(n))$, and $h(n) = \Theta(g(n))$, then $f(n) = \Omega(h(n))$.

8. (**12 points**) Solve the following recurrences:

(a) (**4 points**) Use the iteration method to solve $T(n) = 2T(\frac{n}{4}) + T(\frac{n}{3}) + n$;

(b) (**4 points**) Use the substitution method to verify your solution for Question 8a;

(c) (**4 points**) Solve the recurrence: $T(n) = 3T(n^{\frac{1}{3}}) + \log^2 n$.

(additional space for problem 8)

9. (**6 points**) Illustrate the operation of insertion-sort on the array [71, 25, 36, 13, 41, 7, 20, 8]

10. (**7 points**) Given an array of $[6, 12, 16, 3, 9, 5, 8, 4, 11, 15, 7]$,

(a) (**3 points**) plot the initial min-heap built from the array;

(b) (**2 points**) plot the min-heap after key 16 is decreased to 2;

(c) (**2 points**) continue after step b), plot the min-heap after key 3 is increased to 13.

11. (**8 points**) In Randomized Quicksort (with Lomuto's partition) for a list of n distinct numbers,

(a) (**2 points**) what is the probability that the i-th ranked number stays in the same partition as the $(i+1)$-th ranked number after the first randomized-partition call?

(b) (**3 points**) if the rank of the pivot chosen in the first randomized-partition call is i, and the second partition call works on the left partition after the first partition call, what is the probability that the $(i-k_1)$-th ranked number stays in the same partition as the $(i-k_2)$-th ranked number after the second partition call? $1 \leq k_1 < k_2 \leq i-1$

(c) (**3 points**) is it true that the i-th ranked number will always be compared with the $(i+1)$-th ranked number exactly once throughout the sorting process? briefly justify your answer.

12. (**8 points**) The following is Hoare's partition algorithm for quick-sort:

(a) (**2 points**) analyze the complexity of the algorithm;

(b) (**6 points**) prove the correctness of the algorithm by applying Loop-Invariant to the `while` loop: clearly state the `Loop-Invariant` statement, and prove `initialization`, `maintenance`, and `termination` steps.

```
Alg. PARTITION (A, p, r)
x ← A[p]
i ← p – 1
j ← r + 1
while TRUE
        do repeat j ← j – 1
              until A[j] ≤ x
        do  repeat i ← i + 1
              until A[i] ≥ x
         if i < j
              then exchange A[i] ↔ A[j]
        else return j
```

13. (**13 points**) For an unordered array $A[1 \cdots n]$ with n distinct non-zero numbers,

(a) (**6 points**) construct an $O(n \log n)$ divide-and-conquer algorithm to find a pair of indexes (i^*, j^*), $1 \leq i^* < j^* \leq n$, so that their absolute ratio $|\frac{A[i^*]}{A[j^*]}|$ is the maximum among all the possible (i, j) pairs, $1 \leq i < j \leq n$, write down the pseudo-code, analyze its complexity;

(b) (**7 points**) now we want to find all pairs of numbers so that their absolute ratio equals to a given number $\rho > 0$, i.e., return all index pairs (i, j) such that $|\frac{A[i]}{A[j]}| = \rho$, construct an algorithm with the average-case complexity of $\Theta(n)$, write down the pseudo-code, analyze its complexity; what is the worst-case performance of your algorithm? (*Hint: use Hash Table, assuming simple universal hashing*).

If you use up the space under any particular problem, you can write your answer here. On the page of the problem, tell us part of your work is here and write down the page number of this page and. Don't tear off any pages.

If you use up the space under any particular problem, you can write your answer here. On the page of the problem, tell us part of your work is here and write down the page number of this page and. Don't tear off any pages.