CS 101 *Data Structure and Algorithm*

Fall 2020

Sample Midterm

INSTRUCTOR: Dengji Zhao, Yuyao Zhang, Zhice Yang

TIME: Nov 5th 8:15-9:55

INSTRUCTIONS

- You have 100 mins.
- You are not allowed to bring any papers, books or electronic devices including regular calculators.
- You are not allowed to discuss or share anything with others during the exam.
- You should write the answer of every problem in the dedicated box.
- You should write **your name and your student ID** as indicated on the top of each page of the exam sheet.
- You should write your answers **clearly**.

Name	
Student ID	
Room Number	
Seat Number	
All the work on this exam is my own. **(please copy this and sign)**	

THIS PAGE INTENTIONALLY LEFT BLANK

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

   You should judge whether the following statements are true or false, and fill in your answer (T/F) in front of the statement.

   (a) (2 pt) ______ Reversing a linked list of length n could be done with space complexity $O(n)$ and reversing an array of length n could be done with space complexity $O(1)$.

   (b) (2 pt) ______ We can use an array to simulate a double-linked list.

   (c) (2 pt) ______ Searching for an element in a binary search tree with N nodes will always take $O(\log N)$ time.

   (d) (2 pt) ______ For any heap with n nodes, during `Push()` function, the number of nodes moved is $O(\log n)$, and during `POP()` function, the number of nodes moved is $O(1)$.

   (e) (2 pt) ______ `POP()` operation may cause a heap to become a non-complete binary tree.

   (f) (2 pt) ______ The data structure required for Breadth First Traversal on a tree is a queue.

   (g) (2 pt) ______ $f(n) = \Omega(g(n))$ if and only if $\lim_{n\to\infty} \left|\frac{f(n)}{g(n)}\right| \geq 0$

   (h) (2 pt) ______ Inserting an element into an open-address hash table with load factor α requires at most $1/(1-\alpha)$ probes on average, assuming uniform hashing.

   (i) (2 pt) ______ For a random ordering array with n elements, we would expect there are approximately $n(n-1)/4$ pairs of inversions.

   (j) (2 pt) ______ The preorder sequence of a binary tree will never be same as its BFS sequence.

2. **(20 points) Single Choice (Choose the ONLY ONE correct answer)**

   You should write your answers in the box below

Question (a)	Question (b)	Question (c)	Question (d)	Question (e)

   (a) (4 pt) How many times a node is visited in preorder DFS, postorder DFS and BFS respectively?

   (A) Once; Once; Once
   (B) Twice; Twice; Once
   (C) Once; Once; Twice
   (D) Equivalent to number of indegree of the node

   (b) (4 pt) Which one of the following is the recurrence equation for the worst case time complexity of the Quicksort algorithm for sorting n(≥2) numbers? In the recurrence equations given in the options below, c is a constant.

   (A) $T(n) = 2T(n/2) + cn$
   (B) $T(n) = T(n-1) + T(1) + cn$
   (C) $T(n) = 2T(n-1) + cn$
   (D) $T(n) = T(n/2) + cn$

(c) **(4 pt)** Which of the following statement is wrong for Reverse Polish Notation?

(A) The value of 3 7 $+$ 2 / 5 $*$ 6 8 $-$ 2 $*$ $-$ is 29.

(B) Suppose calculating 1 2 3 $+$ 4 5 6 $*$ $-$ 7 $*$ $+$ $-$ 8 9 $*$ $+$ using a stack. After processing the 11 elements $*$, the top element in the stack is -182.

(C) The RPN of $1 + (2 + 3)$ is $123 + +$

(D) None of the above.

(d) **(4 pt)** Which of the following statements are true for an AVL-tree?

(A) Removing an item from an n-node AVL tree costs $\Theta(\log n)$

(B) One LR rotation has the same effect as a combination of one RR rotation and one LL rotation in order.

(C) The in-order traversal of an AVL tree with unbalance nodes will change after rebalancing.

(D) None of the above.

(e) **(4 pt)** A sorting algorithm is said to be **stable** if two objects with equal keys appear in the same order in sorted output as they appear in the input array.
For example, there is an array of key-value pairs(the sorting is according to the key of an object):

$$[(199, Wang), (177, Zhang), (167, Song), (177, Yuan)]$$

A stable sorting algorithm will output with 100%:

$$[(199, Wang), (177, Zhang), (177, Yuan), (167, Song))]$$

An instable sorting algorithm will output with probability > 0:

$$[(199, Wang), (177, Yuan), (177, Zhang), (167, Song))]$$

$(177, Zhang)$ is still sure to be ahead of $(177, Yuan)$ as they used to be after a stable sorting algorithm.
Please choose the correct statement below(according to the widely accepted implementation in lecture):

(A) bubble sort is stable and insertion sort is stable

(B) bubble sort is stable and insertion sort is not stable

(C) bubble sort is not stable and insertion sort is stable

(D) bubble sort is not stable and insertion sort is not stable

3. (30 points) Multiple Choice (Choose the ALL correct answer, you will get half points if you only choose a subset of correct answers)

You should write your answers in the box below

Question (a)	Question (b)	Question (c)	Question (d)	Question (e)

(a) **(6 pt)** In an AVL tree with 19 nodes, you want to search for 35, which of the following sequences could be the sequence of nodes examined?

(A) 46.36.18.20.23.35

(B) 47.37.18.27.29.35

(C) 27.48.39.43.41.35

(D) 15.25.55.40.35

(b) **(6 pt)** What is the non-decreasing growth order of the following functions? (Give a increasing order of these four options)

(A) $f_1(n) = 2019(\sqrt{n^{1.5}})$
(B) $f_2(n) = 2^n + n! + e^n$
(C) $f_3(n) = 2^{n \log n} + \log^n n$
(D) $f_4(n) = n^2 \log n + n^{\log_2 3}$

(c) **(6 pt)** Please find all correct statements in the following about binary heap?

A Suppose you have an array $A[1, \cdots, n]$ of n elements in arbitrary order, where the root is stored at $A[1]$. If we use the modified Floyd's method which starts orderly percolation from index 1 to $\frac{n}{2}$ to build a min(max)-heap, then the heap we construct satifies the property of min(max)-heaps.
B In any min-heap with more than 3 nodes, the third smallest element can be found in $\Theta(1)$ time.
C Any max-heap with size n can be converted into a min-heap in $\Theta(n)$ time.
D The statements above are all false.

(d) **(6 pt)**Please find all correct statements in the following about sorting:

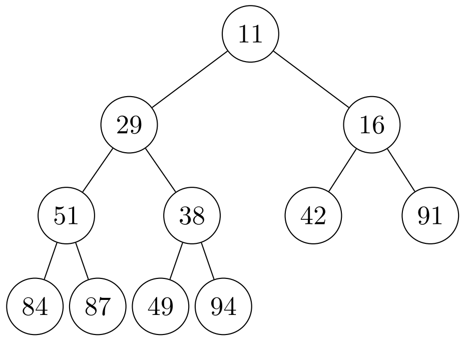
(A) The run time cost of bubble sorting algorithm without any optimization is $\Theta(n^2)$.
(B) The random list(with all permutations with the same probability) has $d = \Theta(n)$ inversions in average.
(C) Flagged bubble sort halts when no swaps occur.
(D) The insertion sort is in-place.

(e) **(6 pt)** Which of the following statements is true?

(A) For any node in an AVL tree, the left child tree of it is an AVL tree
(B) For any node in a binary heap, the left child tree of it is a binary heap
(C) A subtree of a binary heap is a binary heap
(D) A subtree of a binary search tree is a binary search tree

4. (10 points) Heap

The figure shows a binary heap. Each sub-problem are **independent** on each other.



(a) Is it a max-heap or min-heap?

(b) If we call `POP()` on this heap, what will the heap be like? Draw it as a binary tree.

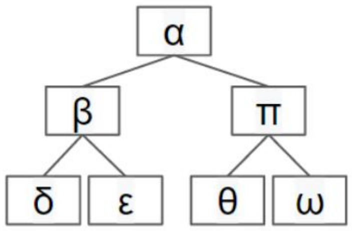
(c) We call `PUSH(x)` function on this heap. If we want `x` to fall at the position of 16, what is the value range of x?

(d) If we use Floyd's method to convert this heap to a max-heap, what will the heap be like? Draw it as a binary tree.

5. (10 points) BST

Consider the tree on the left where greek letters represent numerical values. In the boxes to the right, shade all values that might match the text. Assume all values are unique. For BSTs, assume left items are less than. When treating the tree like a graph, assume nothing about the order of adjacency lists.

Fill in the boxes completely.

	MinHeap, largest item	☐α	☐β	☐π	☐δ	☐ε	☐θ	☐ω	
	MinHeap, smallest item	☐α	☐β	☐π	☐δ	☐ε	☐θ	☐ω	
	BST, largest item	☐α	☐β	☐π	☐δ	☐ε	☐θ	☐ω	
	BST, smallest item	☐α	☐β	☐π	☐δ	☐ε	☐θ	☐ω	
	MinHeap, median item	☐α	☐β	☐π	☐δ	☐ε	☐θ	☐ω	
	BST, median item	☐α	☐β	☐π	☐δ	☐ε	☐θ	☐ω	
	MinHeap, new root after deleteMin	☐α	☐β	☐π	☐δ	☐ε	☐θ	☐ω	
	BST, new root after deletion of α	☐α	☐β	☐π	☐δ	☐ε	☐θ	☐ω	
	MinHeap, root after inserting new item φ	☐α	☐β	☐π	☐δ	☐ε	☐θ	☐ω	☐φ
	BST, root after inserting new item φ	☐α	☐β	☐π	☐δ	☐ε	☐θ	☐ω	☐φ
	Last item dequeued running BFS from ω	☐α	☐β	☐π	☐δ	☐ε	☐θ	☐ω	

6. (10 points) You cannot beat me

Given two points $A(x_1, y_1), B(x_2, y_2)$ in a two-dimensional space, we say that A is dominated by B if $x_1 < x_2$ and $y_1 < y_2$.

Given a set of points $S = \{(x_1, y_1), \cdots, (x_n, y_n)\}$, we say a point $p \in S$ is undominated if it is not dominated by any other point in S. We need to design an algorithm to find all undominated points in S.

(a) (1 pt) Give a straightforward $O(n^2)$ time algorithm for finding all undominated points. (Just describe the idea here, it should be one or two sentences.)

(b) (3 pt) Let m_x be the median value of the x-values of the points in S and define $S_0 = \{(x, y) \in S : x \geq m_x\}$ and $S_1 = \{(x, y) \in S : x < m_x\}$. That is, S_0 contains all points with x value at least the median x-value, and S_1 contains all points with x value less than the median x-value. Can any point in S_0 be dominated by a point in S_1? Give a brief explanation.

(c) (6 pt) Briefly describe an $O(n \log n)$ time algorithm for finding the set of undominated points. Briefly justify why your algorithm is correct and prove that it runs in the required time. (Hint: order all points by x-value first.)