I&C SCI 46 Diagnostic Exam 3, Spring 2022

DO NOT OPEN EXAM UNTIL INSTRUCTED TO DO SO

SILENCE MOBILE PHONE AND OTHER DEVICES

This is a diagnostic exam intended to help you evaluate your readiness for the real exam.

Write the following information **clearly**. You may write **this information only** before the instructor calls to begin the exam. You **may not** write this information after the instructor calls to stop writing.

Name : ____

UCI Email Address : ____________________________________@uci.edu

UCI Student ID # : ____

Read and understand the following rules; failure to abide by these rules, or directions given by course staff during the exam, may result in disciplinary action, including but not limited to a failing grade in the class.

- This exam is solely for students enrolled in this lecture. Anyone not enrolled in this lecture may not take an exam.
- Keep your UCI ID readily accessible during the test. Proctors may request to see it.
- This exam is closed book, closed notes, and is individual effort. Once course staff begin passing out exams, you may not communicate with anyone other than proctors for any reason, nor may you have electronics, including calculators watches and phones, available to you during the test for any reason. **YOU DO NOT NEED A CALCULATOR!**
- If you leave your seat during the test for any reason, your instructor may collect it and deem you to have turned it in. Do not ask proctors for an exemption to this, they are not authorized to grant such.
- If you are still seated at 9:35 AM at the real quiz, you may not leave your seat until explicitly dismissed by the instructor. Leaving after 9:35 AM and before being dismissed may result in a penalty.
- You must take the exam in your assigned seat unless the professor (not a TA) tells you otherwise. You may not open the exam until explicitly told to do so by the professor. The instructor will call to cease writing at 9:45 AM, at which point you must immediately cease writing and close the exam. You may not write any further at that point, including finishing one's current sentence.
- If you believe a question is ambiguous, write at least two reasonable interpretations and indicate clearly which one you will be using. Then answer your question with that assumption. Unless your interpretation makes the problem much more trivial than intended, we will grade your response as if one of us had made that clarification.
- The purpose of the real exam is to evaluate how well *you* understand the material presented in the course. It is an academic integrity violation to do anything that subverts the goals of this assessment including, but not limited to, not doing your own work or submitting that of anyone else.
- Write your answers in the space provided for each question.
- Write your UCI email at the top of each answer page. You may not do this until the exam has begun. There is one point for doing this.

Nothing you write on this page will be graded. The next page in this booklet contains a spot to answer these questions. You may use this page as scratch paper if you would like, and room to do so exists.

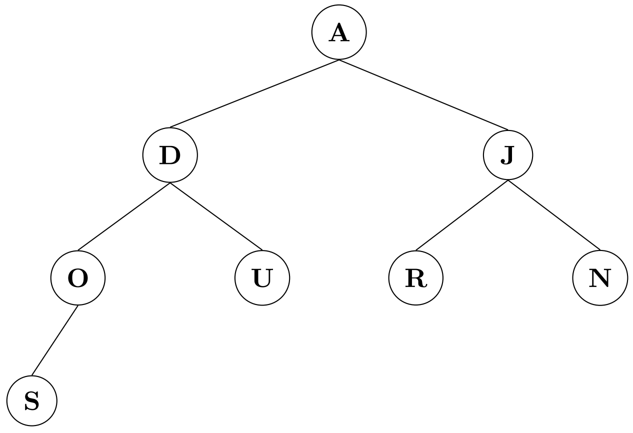
1. (2 points) I have a binary tree (**not** a binary search tree) with the following properties:
   - Each non-null node of T contains a single character
   - An in-order traversal of the tree reads "FBCDAHG"
   - A post-order traversal of the tree reads "BFDHGAC"
     This is a reminder that this is not a binary search tree.

   Give a pre-order traversal of the tree. It may help for you to draw the tree, although that is not required (and should not go on the answer sheet).

2. (1.5 point) In lecture, we saw that an array can be considered a complete tree and, if the heap property applies, it is also a heap. In previous classes, you saw that a string (data type) is an array of characters.

   Recall that alphabetical order is ABCDEFGHIJKLMNOPQRSTUVWXYZ (first to last).

   (a) Here is a binary heap, drawn as a tree. What is the character array/string representation of the heap? Write only the eight characters in the form.



   (b) What is the result of a `remove-min` operation on this heap? You may want to draw the resulting heap, and on the answer page, you will describe the result.

   (c) What is the string/vector representation of the resulting heap?

UCI Email: ______________________________@uci.edu

Question 1. Write only the relevant characters in this box.

Question 2(a). Please write only the string (no punctuation, etc) representing the heap.

Question 2(b). Refer to each node by the character contained within it.

What is the root of the tree?

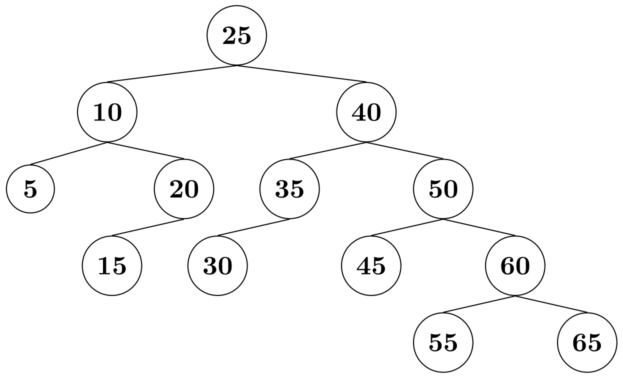
For each node, list the character contained in the left and right children nodes. For any `nullptr`, leave the relevant box **blank**.

Node	Left Child	Right Child
D		
J		
N		
O		
R		
S		
U		

Question 2(c). Please write only the string (no punctuation, etc) representing the heap.

Nothing you write on this page will be graded. The next page in this booklet contains a spot to answer these questions. You may use this page as scratch paper if you would like, and room to do so exists.

3. (1 points) Consider the following AVL Tree. We saw in lecture that an insert operation will cause at most one re-balance operation. We also saw that a delete operation could take more than that. Which key can I delete in the following tree to cause two separate re-balancing operations? *Do not select a node that has two children for your answer. You will get a zero for this question if you do so.*



4. (2 points) In project 4, you did not have to write a delete on your AVL tree. You're going to write it here.

   You may assume every node in the tree stores a Key,Value pair, along with left, right, and parent pointers. You also have the following helper functions already written and tested (you may call them and assume they work):

   - `Node * find(Key k)` returns a pointer to a Node that has the given key. It returns `nullptr` if the key is not in the tree.
   - `Node * rebalance(Node * z)`, which performs the local rebalancing, given that z is the lowest unbalanced node. This is the same definition of z we used in lecture.
   - `Node * localDelete(Node * n)`, which deletes the given node, freeing all relevant memory. It returns a pointer to the "replacement" (same child of same parent) if a non-leaf with one child is deleted, the parent if a leaf is deleted, or the actually-deleted node if the parameter had two children (after moving the key/values appropriately). This is the "normal BST" localDelete, and does not do any height re-balancing.
   - `int height(Node *n)`, which returns the height of the given node. That is, the number of hops to fall off the farthest leaf underneath it. If the parameter is `nullptr`, it returns -1.

   While we will not require strict C++, you should write your code in such a way as to make it very clear to the grader that you *could* implement it correctly from your ideas.

   `void delete(Key k)`

UCI Email: ______________________________@uci.edu

What is your answer for question 3?
Write the key in this box:

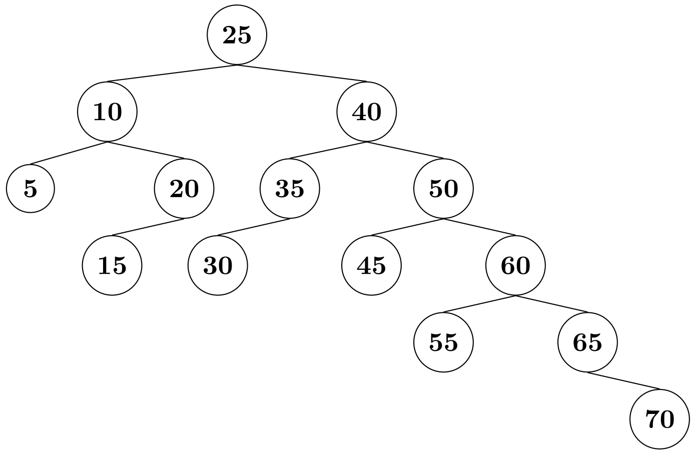
....

Write your code for question 4 here. Note that your answer might not use the entire space provided.

```
void MyAVLTree<Key,Value>::delete(Key k)
```

Nothing you write on this page will be graded. The next page in this booklet contains a spot to answer these questions. You may use this page as scratch paper if you would like, and room to do so exists.

5. (2 points) Consider the following Binary Search Tree, being balanced via the rules of AVL-Trees. We are examining it in the middle of the "insert" operation. A Key has been inserted into the tree, but we have not (yet) performed any rotations.
   (a) Which key did we just insert?
   (b) In setting up the update operation, what is the key stored at z, as described in lecture?
   (c) In setting up the update operation, what is the key stored at y, as described in lecture?
   (d) In setting up the update operation, what is the key stored at x, as described in lecture?



6. (2 points) Suppose we are writing code for a priority queue, implemented as a min heap (using a `std::vector`). We have `insert, min` and `extract-min` written and we want to add a function `where-is-key`(k). This function is designed to answer the following query: is k in the heap, and if so, which index in the vector holds k? I want this to run in $\mathcal{O}(\log n)$ time. The other operations' running times may increase by a factor of $\mathcal{O}(\log n)$ if needed. Explain how to do this. You may use $\mathcal{O}(n)$ additional space to support this operation.

   You will still get full credit if your running time is *in expectation* $\mathcal{O}(\log n)$ instead of guaranteed, and/or your $\mathcal{O}(n)$ additional space is in expectation, but if either or both is in expectation, you need to state that.

   After you take the diagnostic test, and before you read the answer to this question, discuss this question and your solutions with your study group for a few minutes. Of course, make sure they have taken the diagnostic test too!

I&C SCI 46- Diagnostic-MT3-S22 UCI Email: ______________________________@uci.edu

What is your answer for question 5?

5(a)	5(b)	5(c)	5(d)

. . . .

Check this box if your running time is in expectation: ☐

Check this box if your space usage is in expectation: ☐

Write your answer to question 6 below: