I&C SCI 46 Diagnostic Exam 2, Winter 2023

DO NOT OPEN EXAM UNTIL INSTRUCTED TO DO SO

SILENCE AND STORE ALL ELECTRONICS

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

The following rules apply to you, whether you think they do or not. Read and understand them; 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.
- You may not open the exam or begin writing until the instructor has explicitly given you permission to do so.
- 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.
- You must take the exam in your assigned seat unless the professor (not a TA) tells you otherwise. The instructor will call to cease writing at X:48 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.
- We will only grade responses marked in the space provided for each question.
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. Suppose we have a Cuckoo Hash Table with each table having room for \( m = 11 \) entries each. Our hash functions are \( h_0(x) = x \% 11 \) and \( h_1(x) = (x/11) \% 11 \), where the / is integer division (floor of division; discard remainder). For example, \( h_0(1289) = 2 \) and \( h_1(1289) = 7 \). We insert the keys 46, 51, 84, 200, 134, 138, 52, 184, 179, 25, 28, 36, 72, 85, in that order, into the Hash Table, in the manner described in lecture.

For your convenience, here are the hash values:

<table>
<thead>
<tr>
<th>( x )</th>
<th>46</th>
<th>51</th>
<th>84</th>
<th>200</th>
<th>134</th>
<th>138</th>
<th>52</th>
<th>184</th>
<th>179</th>
<th>25</th>
<th>28</th>
<th>36</th>
<th>72</th>
<th>85</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h_0(x) )</td>
<td>2</td>
<td>7</td>
<td>7</td>
<td>2</td>
<td>6</td>
<td>8</td>
<td>8</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>( h_1(x) )</td>
<td>4</td>
<td>4</td>
<td>7</td>
<td>7</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
</tbody>
</table>

(a) (1 point) After the first five keys are inserted, are they in the upper array (indexed by \( h_0 \)) or the lower array (indexed by \( h_1 \)) after all of first five keys have been inserted? Circle your choice on each clearly.

(b) (1 point) Which value will be the last one inserted successfully into the table prior to the rehash/resize? Recall that we are rehashing / resizing only after an element cannot be inserted; there is no \textbf{a priori} maximum eviction length as there was in project three.

2. (1 point) I have a binary tree (\textbf{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 “YXBJSWTHGMR”
- A post-order traversal of the tree reads “YXSJTWPBGMRH”

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).

3. (3 points) Describe how to perform a removal from a hash table that uses linear probing to resolve collisions where we do not use a special marker to represent deleted elements. That is, we must rearrange the contents so it appears that the removed entry was never inserted in the first place. Do not rehash/reinsert any more elements than is necessary.
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. Circle clearly your answers for 1(a) here. Note that they are listed in numerical order, not the order in which they were inserted. Answers that depend on an insertion order other than that which we listed will not earn credit.

<table>
<thead>
<tr>
<th>46</th>
<th>upper</th>
<th>lower</th>
</tr>
</thead>
<tbody>
<tr>
<td>51</td>
<td>upper</td>
<td>lower</td>
</tr>
<tr>
<td>84</td>
<td>upper</td>
<td>lower</td>
</tr>
<tr>
<td>134</td>
<td>upper</td>
<td>lower</td>
</tr>
<tr>
<td>200</td>
<td>upper</td>
<td>lower</td>
</tr>
</tbody>
</table>

Write your answers for 1(b) and for 2 in these boxes. For question 2, write only the letters.

<table>
<thead>
<tr>
<th>1(b)</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Write your answer to question 3 here. Note that your answer might not use the entire space provided.
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.

4. (1 point) Consider the following Level-balanced binary search Tree.
   (a) Select a leaf node whose deletion will cause no other nodes to change level.
   (b) Select a leaf node whose deletion will cause exactly one other node to change level.

5. (1 point) Consider the following Level-balanced binary search tree.
   (a) Give an integer key I could insert into this tree that would cause exactly zero keys that were already in the tree to change level when the insert completes.
   (b) Give an integer key I could insert into the tree that would cause exactly one key that was already in the tree to change level when the insert completes.
Write your answers for questions 4 and 5 in these boxes. Write only the integers.

<table>
<thead>
<tr>
<th>4a</th>
<th>4b</th>
<th>5a</th>
<th>5b</th>
<th>5c</th>
</tr>
</thead>
</table>

You may use this area for scratch work, but nothing written underneath this will be graded.
6. (1 point)

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

\[
\begin{array}{c}
\text{1} \\
\text{4} & \text{9} \\
\text{14} & \text{21} & \text{18} & \text{15} & \text{19}
\end{array}
\]

(b) What is the result of a \text{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?

7. (3 points) Suppose we have a binary tree, which may or may not be a Binary Search Tree, with the following declaration:

```cpp
struct Node {
    Node(unsigned k) : key(k), left(nullptr), right(nullptr) {}
    unsigned key;
    Node *left;
    Node *right;
};
```

Finish writing the following function, in your choice of C++, pseudo-code, or English, that determines if the given tree is a valid \textit{binary search tree} or not.

```cpp
bool validBST(Node *r) {
```

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.
Question 6(a). Please write only the vector representation of the heap.

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
</table>

Question 6(b). Refer to each node by the number contained within it.

What is the root of the tree?

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
</table>

For each node, list the character contained in the left and right children nodes. If there is no such child, leave the box blank.

<table>
<thead>
<tr>
<th>Node</th>
<th>Left Child</th>
<th>Right Child</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Question 6(c). Please write only the integers in the vector representation.

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
</table>

Write your code for question 7 here. Note that your answer might not use the entire space provided.

```cpp
bool validBST(Node * r)
```