CS 251 – Data Structures
Midterm Exam #2 - Prof. Reed
Fall 2014

You may take this test with you afterwards, but you must turn in your “bubble form” answer sheet.

This test has the following sections:
I. True/False ....................... 50 points; (25 questions, 2 points each)
II. Multiple Choice ............... 50 points; (12 questions, various points each)
---------------
100 points total

This test is worth 10% of your final grade. You must put your answers on the bubble form. You are allowed to have resources with you printed on paper, but no computers. For the multiple choice problems, select the best answer for each one and select the appropriate letter on your answer sheet. Be careful - more than one answer may seem to be correct. Some questions are tricky.

True/False: (2 points each) On your bubble form fill out A for true and B for false.

T F 1. An algorithm that runs in $O(2n)$ runs faster than another version that runs in $O(4n)$.

T F 2. $n$ unique values in the range 0 to $n-1$ could be sorted in $O(n)$ time by using an additional array of size $n^2$.

T F 3. Given an unsorted binary tree of $n$ randomly distributed integer values, it is possible to use only the minimum(...), successor(...) and the insert(...) functions of a binary search tree to print out those $n$ values in sorted order in $O(n \log n)$.

T F 4. Given an unsorted binary tree of $n$ randomly distributed integer values, it is possible to use only the minimum(...), delete(...) and the insert(...) functions of a binary search tree to print out those $n$ values in sorted order in $O(n \log n)$.

T F 5. Assume a binary search tree has been built using $n$ integer values. Regardless of the distribution of the input values used to build the tree, any value can be found in $O(n)$ time or better.

A Eulerian walk is a trail in a graph which visits each edge exactly once.

T F 6. There is a Eulerian walk for the following graph:

![Graph Image]
7. There is a Eulerian walk for the following graph:

8. For program #2 we inserted and deleted random numbers into a large array, doing the same thing using a linked list. For values of n < 10000 the linked list was faster.

9. A greedy algorithm by definition is always guaranteed to find a lowest-cost solution.

10. To calculate the shortest path from one node to all others in an un-weighted graph we can use a breadth-first search.

11. We can find the two smallest values in a min heap in constant time.

12. Given the total number of elements in a heap, we can compute the number of levels in the heap in constant time.

13. A heap stored using an array representation has the advantage of not needing to store the child pointers.

14. A graph of Chicago city streets would likely be better represented by a directed graph as compared to an undirected graph.

15. Road networks tend to be sparse graphs.

16. In a weighted graph a longer path might still be a lower-cost path.

17. The prerequisites diagram for UIC CS courses is a cyclic un-weighted directed labeled graph.

18. Adjacency matrices (as opposed to adjacency lists) are better suited for dense graphs.

19. Compared to an adjacency matrix, an adjacency list for a sparse graph gives better performance at finding the successors to a vertex.

20. Compared to an adjacency matrix, an adjacency list for a sparse graph gives better performance at finding the predecessors to a vertex.

21. A complete graph with 3 vertices has 3 edges.

22. A complete directed graph with 3 vertices has 3 edges.

23. The sum of ones in a column for an adjacency matrix is the out-degree for that column’s vertex.

24. When using an adjacency matrix to represent a graph, an algorithm to find the number of edges in the graph can be done in $O(n \log n)$ time.

25. Finding the in values for a node in a graph represented using regular adjacency lists requires scanning all the values on all the lists.

Multiple Choice (50 points total, different numbers of points for different problems)
26) (2 points) Assume you have a stream of n integer values in ascending order that you want to insert into a binary search tree. What is the run-time complexity (Big-Oh) of running this program on this data?

a) \( O(c) \)

b) \( O(n^2) \)

c) \( O(n \log n) \)

d) \( O(\log n) \)

e) \( O(n) \)

27) (2 points) What is the input that would result in the creation of the binary search tree shown below?

```
     22
    /  \
   11   29
  /  \\  /  \
 8   13 26 30
```

a) 8 11 13 22 26 29 30
b) 22 11 8 13 29 26 30
c) 8 13 11 26 30 29 22
d) 22 11 29 8 13 26 30
e) None of the above

28) (4 points) The expression: \( A \times (B + C) - 3 \times 2 + \frac{6}{(D - 1)} \) converted to postfix is:

a) \( AB*C+3-26+/D1- \)

b) \( ABC*+32*-6+D1-/ \)

\( \text{c) } ABC+*32*-6D1-/+ \)

d) \( A*BC+3-2*6+/D1- \)

e) None of the above

29) (4 points) Consider a weighted graph represented using adjacency lists, with the same weights on several different edges. Two different programs give minimum spanning trees with different sets of edges in the solution. From our experience with this in class, what is the most plausible reason?

a) One of the programs is not giving the correct answer

b) One of the programs started at a different point

c) The adjacency lists in one version inserts in order, but the other doesn’t

d) One uses adjacency lists, while the other uses an adjacency matrix

e) None of the above
Consider a binary tree T in which all the levels are completely full and the nodes are numbered in a breadth-first fashion, where the root is 1, its immediate left and right children are 2 and 3, and their children respectively are 4, 5 and 6, 7, and so on. Which of the following are true about T, where it is implemented as an array?

A. It could be implemented using a static (not dynamic) array, but the total number of Nodes must be known.
B. It would require less storage than using a tree of Node structures.
C. The level a node is in can be determined by its node number.
D. Accessing any node can be done in constant time.

a) A, B  
b) A, C, D  
c) A, B, D  
d) B, C, D  
e) A, B, C, D

Consider the following max Heap:

Consider what this max heap would look like after the following instructions are executed, where q is the Heap:

1. cout << getMax( q, qSize);
2. cout << getMax( q, qSize);
3. cout << getMax( q, qSize);
4. qInsert( q, qSize, 10);
5. qInsert( q, qSize, 12);

After doing the above operations, the root node then its left child and then right child would be:

a) 10 12 11  
b) 10 11 12  
c) 8 9 11  
d) 8 11 9  
e) None of the above
For the following 3 problems if it makes a difference assume the graph is stored using adjacency lists, where vertices are stored in ascending alphabetical order.

32) (4 points) Starting at vertex A in the graph at right, what is the list of vertices in the order in which they would be visited using Prim’s algorithm to find a Minimum Spanning Tree?

a) A D B C F G E H
b) A D C B F E G H
c) A D G H F E C B
d) A D C F E H G B
e) None of the above

33) (4 points) Select the edges in the order in which they would be added for Kruskal’s algorithm to find a Minimum Spanning Tree, just as we did in class. (The above graph is given again here for your convenience so you can draw on it.)

What is the order of edges added?

a) E, F, F, H, C, D, D, G, D, F, A, D, A, B
b) E, F, F, H, D, F, C, D, D, G, A, B, A, D
c) E, F, F, H, C, D, D, F, A, D, D, G, A, B
d) E, F, F, H, C, D, D, F, D, G, A, B, A, D
e) None of the above
Consider the graph shown at right. Starting from **vertex C** fill in the table below using Dijkstra’s algorithm to show each step from vertex C to all other points, similar to what was done in class and in lab.

The first row has been done for you.

<table>
<thead>
<tr>
<th>S Selected</th>
<th>Vertex</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>{}</td>
<td>C</td>
<td>maxint</td>
</tr>
<tr>
<td>{C}</td>
<td>D</td>
<td>10</td>
</tr>
<tr>
<td>C,D</td>
<td>F</td>
<td>10</td>
</tr>
<tr>
<td>C,D,F</td>
<td>A</td>
<td>✓10</td>
</tr>
<tr>
<td>C,D,F,A</td>
<td>G</td>
<td>✓10</td>
</tr>
</tbody>
</table>

34) *(5 points)* After filling out the table, what are the first 4 values you wrote down in the “**Vertex Selected**” column (highlighted with a rectangle)?

- a) D G H F
- b) D A F G
- c) D A G B
- d) D F A G
- e) None of the above

35) *(5 points)* After filling out the table, what are the values, left-to-right, in the circled cells of the table?

- a) 12 12 11 12
- b) 12 12 12 12
- c) 12 12 11 13
- d) 13 13 11 13
- e) None of the above
Given the following directed graph, fill in the values for the packed array representation of an adjacency multi-list in the table at right below. Handle all elements in ascending vertex order.

36) *(5 points)* What are the 6 numbers in the highlighting rectangle at left in the chart above?

a) 0 3  
   1 0  
   2 1

b) 3 0  
   1 0  
   1 2

c) 0 3  
   1 0  
   3 2

d) 0 3  
   1 0  
   1 2

e) None of these

37) *(5 points)* What are the 6 numbers shown in the highlighting circle at right in the chart above? (Note that the slash character ‘/ ’ represents NULL.)

a) 8 4  
   / 7  
   / 6

b) 8 4  
   7 /  
   6 /

c) 4 8  
   / 7  
   / 6

d) 8 4  
   / 7  
   6 /

e) None of these