Name

This is a take home exam so all the work must be entirely your own. You may consult your own notes, text books, and other published material. Cite any sources you used (other than the class text) in the space below.

You may ask the instructor or TA to clarify anything you don’t understand. Any corrections to the exam will be posted to the class moodle discussions. This exam is based on a 3 hour closed book exam for CMPS 101. Graduate students should be able to get at least 90% on it as a take-home exam.

Answer the questions on the exam itself as much as possible. If you use additional paper, staple it to the exam. Clearly label your work and show enough steps to justify your answers.

10 questions, 80 points, 72 to pass. Good luck!

question 1: ______ ( out of 14 )
question 2: ______ ( out of 7 )
question 3: ______ ( out of 5 )
question 4: ______ ( out of 8 )
question 5: ______ ( out of 6 )
question 6: ______ ( out of 10 )
question 7: ______ ( out of 6 )
question 8: ______ ( out of 8 )
question 9: ______ ( out of 8 )
question 10: ______ ( out of 8 )

TOTAL: ______ ( out of 80 )
1. (14 pts) Quickies (2 pts each):
   a. What is the “partially ordered condition” associated with the binary-heap implementation of a (min) priority queue (assume smaller values have more priority)?
   
   b. Evaluate the sum: \( \sum_{i=0}^{n} (2^{i+2} - 2^i) \)
   
   c. What are the average case and worst case running times of the Quicksort algorithm when sorting \( n \) elements (use \( \Theta \) notation)?
   
   d. Why doesn’t the \( \Omega(n \log n) \) lower bound for sorting apply to radix sort?
   
   e. What are the two heuristics used in the “almost linear” \( (O(n \alpha(n)) \) or \( O(n \log^* n) \)) implementation of the Union-Find abstract data type (other than using the pointer representation)?
   
   f. Define the term “free tree” (or “unrooted tree”).
   
   g. Mark each of the following true or false.
      • \( n \log n \) is \( O(n \sqrt{n}) \)
      • \( n \log n \) is \( \Omega(n \sqrt{n}) \)
      • \( n \log n \) is \( \Theta(n \sqrt{n}) \)
2. (7 pts) Complete the following definition of “big-$O$” (2 pts):

\[ f(n) \text{ is in } O(g(n)) \text{ if and only if there exists } c > 0 \text{ and } n_0 > 0 \]

such that for all \( n > n_0 \):

\[ f(n) \leq c g(n). \]

(5 pts) Use the definition to formally prove that \((n + 2)^2\) is in \(O(n^3)\).

3. (5 pts) Find an asymptotic bound (\(\Theta\)-expression) for the function \(T(n)\) defined by the recurrence:

\[ T(n) = 9T\left(\left\lfloor n/3 \right\rfloor\right) + n\sqrt{n}; \text{ and } T(0) = 1. \]

Justify your answer.
4. (8 pts) Gary the grad student thinks he has an improved hash table using chaining. Instead of storing the elements hashing to the same place in a linked list, he uses a standard binary search tree. Gary claims that his modified hash table maintains its constant time expected performance for insertions and lookups, but now has a $O(\log n)$ worst case time for these operations. What is wrong with Gary’s scheme (4 pts) and how can it be fixed (4 pts)?

5. (6 pts) Draw the state of the tree-based union-find data structure (the one with the “almost linear” running time) after the following sequence of operations have been performed. (Assume that initially A, B, C, D, E, F, G, and H are each in their own set, and break any ties by unioning the first operand into the second).

union(find(A),find(B)), union(find(C),find(D)), union(find(D),find(A))

Continue with the following operations, again drawing the state at the end.
union(find(E),find(F)), union(find(G),find(H)), find(C)
6. (10 pts) Consider Depth First Search on the following directed graph:

```
A --- B --- C
   |   |   \
   D --- E --- F
```

a. (3 pts) List the vertices in the order they are discovered by a DFS starting at vertex “A”. Visit the lowest lettered vertex whenever you have a choice (assume the adjacency lists are in alphabetical order).

b. (4 pts) What are the 4 types of edges that result from a depth first search? Give one example of each kind from your DFS in part (a).

c. (3 pts) Identify the strongly connected components of the graph by listing the vertices in each strong component.

7. (6 pts) Use breadth first search to build a shortest path tree in the following undirected graph (draw the tree on the graph below). Start your search (and root your tree) at vertex A. When executing a “for each adjacent vertex” loop, examine the adjacent vertices in alphabetical order.

```
A --- B --- C --- D
   |   |   |
   E --- F --- G --- H
```
8. (8 pts) Run Prim’s MST algorithm (starting at vertex A) on the following graph. List the edges of the MST in the order that they are added to the MST by the algorithm.

\[
\begin{array}{c}
A & 4 & B & 9 & C & 3 & D \\
6 & 5 & 8 & 7 \\
E & 3 & F & 2 & G & 1 & H
\end{array}
\]

9. (8 pts) Run Kruskal’s MST algorithm on the following graph. List the edges of the MST in the order that they are added to the tree by the algorithm.

\[
\begin{array}{c}
A & 6 & B & 9 & C & 4 & D \\
8 & 7 & 6 & 5 \\
E & 3 & F & 2 & G & 1 & H
\end{array}
\]
10. (8 pts) Simulate Dijkstra’s shortest path algorithm on the following directed graph to find the cheapest paths from A to all other vertices. List the vertices in the order in which they are dequeued, and draw the resulting shortest path tree rooted at A on the graph immediately below.

<table>
<thead>
<tr>
<th>Order Dequeued</th>
</tr>
</thead>
<tbody>
<tr>
<td>A 3 B 9 C 4 D</td>
</tr>
</tbody>
</table>

Give a formal proof by induction on the number of vertices for the following theorem:

Any undirected graph having the same number of edges as nodes contains a cycle.

(Hint: you may use without proof the fact that any connected graph on $n$ vertices with $n$ edges has at least one vertex of degree $\leq 2$, and consider replacing a node of degree 2 with a single edge in the inductive step.)