CS 218, Fall 18

Entrance Quiz

Name (first last) .................................................................

Student ID .................................................................

- This quiz is **closed book, closed notes** and 30 minutes long
- Read the questions carefully
- No electronic equipment allowed (cell phones, tablets, computers, ...)
- Write legibly. What can’t be read will not be graded
- Use pseudocode (or English) to describe your algorithms
- Always remember to analyze the time complexity of your solution
- If you have a question about the meaning of a question, raise your hand
Problem 1. (42 points: 6 points if correct, 3 if unanswered, 0 if wrong)
Mark by true or false each of the following (no need to prove).

\[ n \log_2 n \in O(n \log_2 \log_2 n^8) \]  
\[ 4^{\log_2 n} \sqrt{n} \in \Omega(n^2) \]  
\[ \log_3 3^n \in \Theta(\sqrt{n} \log_2 2^{\sqrt{n}}) \]  
When given an array of \( n \) number to sort, QUICKSORT runs in \( O(n \log n) \) time in the worst-case 

The following questions are on graphs; assume that \( n = |V| \) is the number of vertices, and \( m = |E| \) is the number of edges; DFS is “depth first search”; BFS is “breadth first search”; in DFS/BFS the the set of edges visited during the execution of these algorithms are called tree or discovery edges; non-tree edges are the others (also called back edges in DFS, cross edges in BFS)

- Given the spanning tree \( T \) formed by the discovery (tree) edges of a BFS traversal of a connected undirected graph \( G \) started from node \( s \), for each vertex \( v \), the path on tree \( T \) is the shortest path between \( s \) and \( v \) 

- For a connected undirected graph \( G \), the absence of back (non-tree) edges with respect to a DFS tree implies that \( G \) has at least a cycle 

- If one runs a DFS on a connected undirected graph, the number of back (non-tree) edges is exactly \( m - n + 1 \)
Problem 2. (24 points: 8 points each)

For each of the concepts listed below write a precise (possibly formal) definition. Do not explain or comment about the corresponding algorithm, if any.

1. binary heap

2. transitive closure of a directed graph $G = (V, E)$

3. spanning tree of an undirected graph $G = (V, E)$
Problem 3. (34 points)

Given a sorted (low to high) array of distinct integers \( \{a_1, a_2, \ldots, a_n\} \), describe an \( O(\log n) \) algorithm to determine whether there exists an index \( i \) such that \( a_i = i \). For example, in \( \{-10, -3, 3, 5, 7\} \), \( a_3 = 3 \). In \( \{2, 3, 4, 5, 6, 7\} \), there is no such \( i \). Explain briefly how the algorithm works, and why your solution takes \( O(\log n) \) time.