CMPSC 465
Data Structures and Algorithms

Spring 2013

Exam IV: Graph Algorithms

April 11, 2013

Name: ____ Last 4 digits of ID: ____ ____ ____ ____

Honor Code Statement

I certify that I have not discussed the contents of the exam with any other student and I will not discuss the contents of this exam with any student in this class before it is returned to me with a grade. I have not cheated on this exam in any way, nor do I know of anyone else who has cheated.

Signature: ____

Directions/Notes:

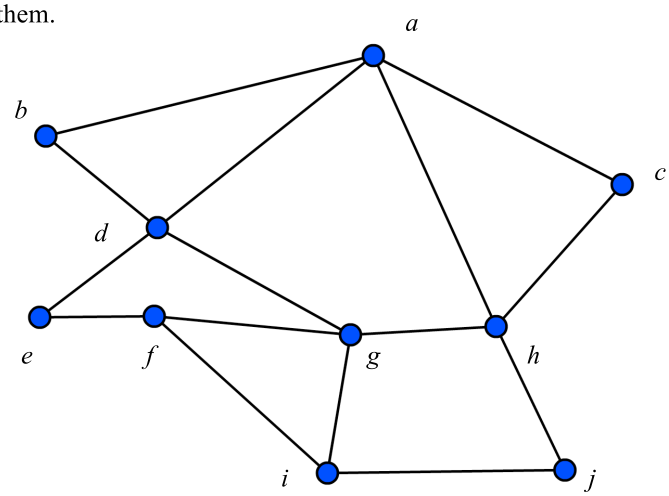
- Write your ID on every page of this exam. Write your name just on this cover page.
- No outside notes, books, or calculators are permitted.
- Be sure to sign the honor code statement when you are finished.
- All questions on this exam are implicitly prefaced with "As taught in CMPSC 465 lectures this term."
- Always justify your work and present solutions using the conventions taught in class.
- The problem that is a take-off on a challenge problem is on the last sheet of the exam. You must solve that problem first and remove and turn in that page of your exam within the first 30 minutes of the exam period. Due to the logistics of timing the challenge problem, if you arrive late without having made prior arrangements with the instructor, you will either forfeit missed time or be permitted to earn only up to half credit on the problem.
- Use pencil to complete this exam. Use of pen will result in an automatic 10-point deduction and we reserve the right not to read any problem with cross-outs.

Score Breakdown:

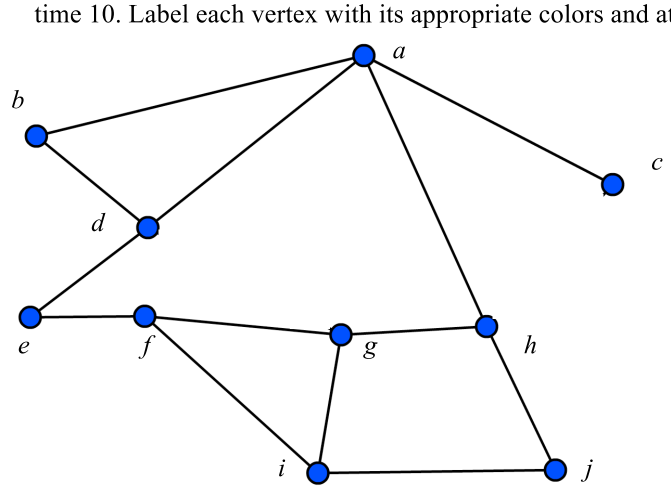
#	1	2	3	4	5	6	7	A.P.	Total
Score									
Value	25	15	10	11	10	10	20	--	**100**

1. **Tracing Execution**. [25 pts.]

a. Using the graph below[1], illustrate the action of a breadth-first search with *a* as the source. Label each vertex with its appropriate colors and attributes. Stop after you have 3 black vertices. You may cross out attributes as you change them. [5]

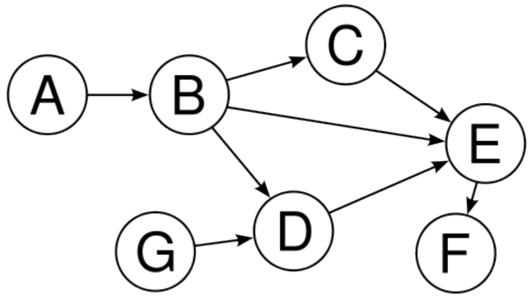


b. Using the graph below, illustrate a depth-first search with *a* as the source. Trace the algorithm up to and including time 10. Label each vertex with its appropriate colors and attributes. [5]

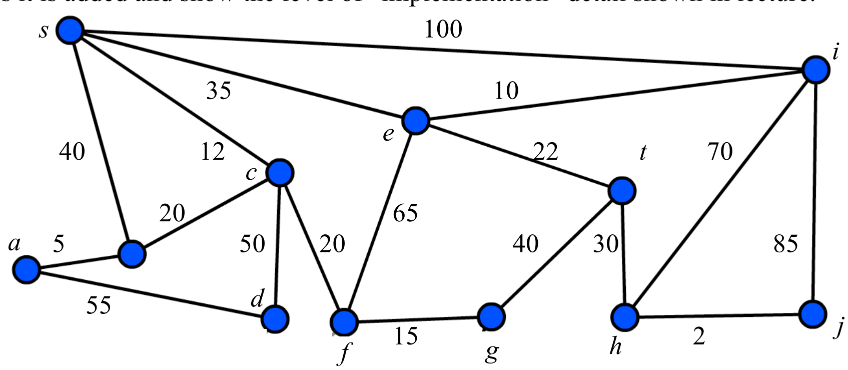


1 Taken from that game at Planarity.net I sent you a while back. It proved so very helpful for this.

c. Trace topological sort (to the level of detail done in class) on this DAG[2]: [5]

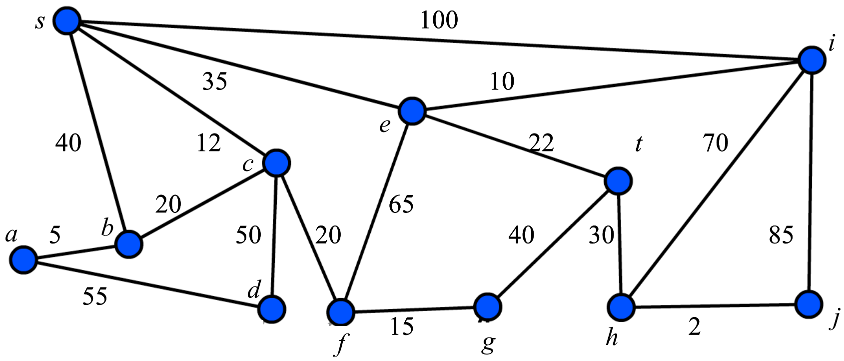


d. Using the graph shown, trace Prim's algorithm using vertex *s* as a source. Label each vertex with a circled number as it is added and show the level of "implementation" detail shown in lecture. [5]



2 This comes from Wikimedia Commons.

e. Determine a shortest path between *s* and *t* using the algorithm we studied. Show work in as much detail as we did in class; if it is possible to stop early, you may. [5]



2. **Short Answer Questions – Traversals** [15 pts.]

   a. Concisely contrast BFS and DFS. (Your response must comfortably fit in the space given.) [3]

   b. An elementary data structure is used in BFS. Which one, how is it used, and why does it make sense? [3]

   c. Throughout the entire execution of a breadth-first search, how much time is spent adding and removing data to the data structure from the last question? Why? [2]

   d. Draw an directed graph on 4 or 5 vertices and illustrate a depth-first tree within it. Point out an example of a tree edge, a back edge, and a forward edge and explain. [4]

   e. In a depth-first search of G, vertex x has a discovery time of 10 and a finishing time of 19. What can you say about discovery and finishing times of its ancestors? [3]

3. **Short Answer Questions – Topological Sort** [10 pts.]

   a. In recitation, we explored an alternate topological sort algorithm. Explain how this algorithm gets around using finishing times as in our algorithm, yet still produces a correct order. (You don't need to explain the whole algorithm; you only need to talk about finishing times.) [4]

   b. Explain why the running time of our topological sort algorithm is what it is. [3]

   c. Let *u* and *v* be vertices of a DAG *G* and let (*u*, *v*) be an edge in *G* we're exploring during topological sort. Suppose *v* is BLACK. Why does topological sort produce correct results in this case? [3]

4. **Short Answer Questions – MSTs** [11 pts.]

   a. Write an invariant for the loop in Kruskal's algorithm involving a cut. You may assume you're working with a graph *G*, but otherwise must define any necessary quantities. [4]

   b. Rigorously derive the running time of Prim's algorithm, implemented as we did in lecture. [4]

   c. Why is Prim a greedy algorithm? [3]

5. **Short Answer Questions – Dijkstra's Algorithm** [10 pts.]

   a. Suppose you run Dijsktra's algorithm on an expanded graph model of our entire campus and it suggests optimal travel time between IST and Thomas is 14 minutes. What does this say about other routes between IST and Thomas? [2]

   b. Suppose you have a working implementation of Prim's algorithm. What changes would you have to make to turn this into an implementation of Dijsktra's algorithm? [4]

   c. Explain, at a high level of abstraction in three or fewer sentences (or, better, bullet points), the outline of the proof that Dijsktra's algorithm is correct [4]

6. **Uniqueness of MSTs.** First, you may use the following theorem as a known fact you will likely need:

   If G is a graph with spanning tree T and $e \in E(G)$ but $e \notin E(T)$, then the graph obtained by adding e to T contains one and only one set of edges that forms a circuit.

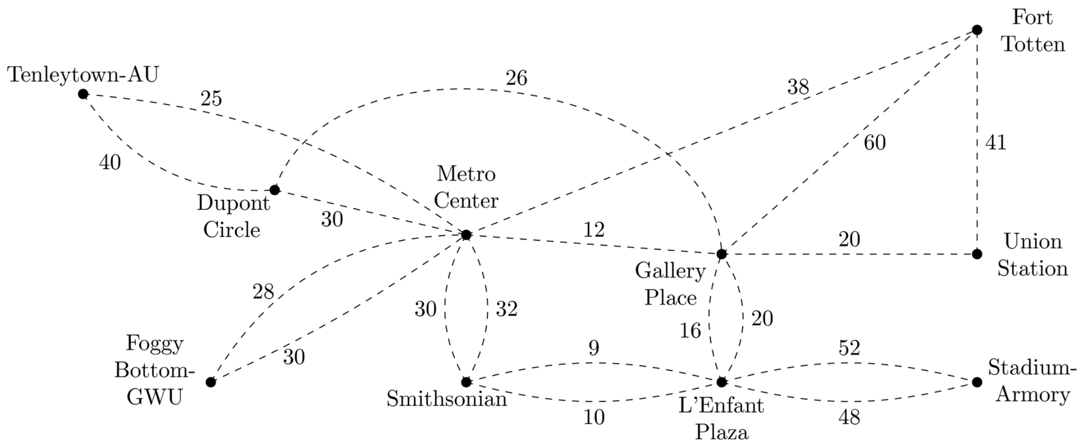
   Prove the following claim:

   If G is a connected, weighted graph and no two edges of G have the same weight, there exists a unique minimum spanning tree for G. [10 pts.]

7. **Challenge Problem Follow-Up**. [20 pts.]

a. **Spanning Trees.** In one of the challenge problems, you were presented with three "maybe" MST algorithms.

i. Pick one that *worked* and trace it on the graph below[3]. Define a numbering system for your steps, explain what you are doing and, either in pseudocode or formal language, summarize the loop in the algorithm. [6]



ii. Recall that we can write a generic MST algorithm with a loop invariant that an edge set *A* is a subset of some MST's edge set. Does that invariant apply to your algorithm or not? Explain. [2]

iii. Suppose you are proving the correctness algorithm you selected. Outline the termination part of your proof **only**. In other words, what can you take as fact for this subproof, what can you use to reason through it, and what are you trying to prove? [4]

iv. Suppose the input to an MST algorithm is *G* and that algorithm removes edges that form cycles. How many edges does such an algorithm remove? Explain. [3]

---

[3] This is a model of the DC Metro we drew up last semester for a Kruskal's algorithm problem on 360's Exam 3. Forgive me for recycling it. An interesting consequence: you can look up the solution to that problem and compare the result to this. What should happen?

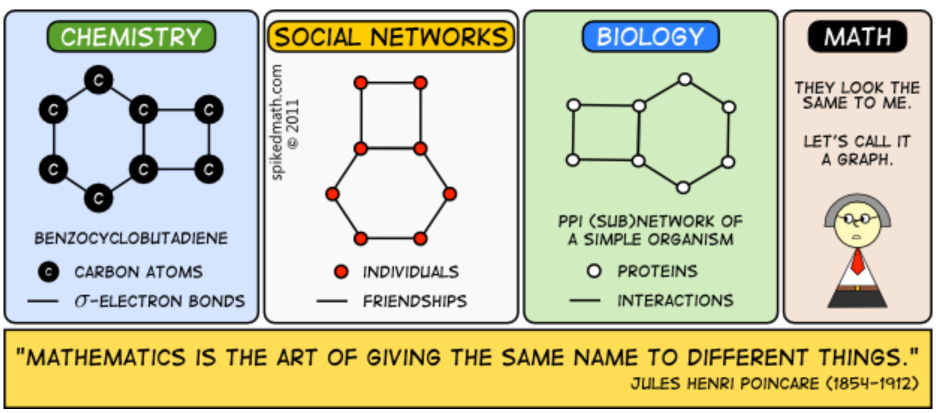
 Last 4 Digits of ID: ___ ___ ___ ___

b. **Classifying Edges in a BFS**. A problem had you consider classification of edges in a BFS as we did in DFS. Prove or disprove each claim.

i. In a BFS of a directed graph, $\forall$ tree edges (u, v), $v.dist = u.dist + 1$. [2]

ii. In a BFS on a simple undirected graph, there are no back edges and no forward edges. [4]

You don't need that much room. Here's something about graphs from Spiked Math to fill the space:



IMPORTANT NOTES:

- This last problem must be solved in the first 30 minutes of the exam period.
- Remove this sheet. Listen for time to be called to pass it in.
- Make sure the last 4-digits of your ID number on this sheet match the rest of your exam or you will not get credit for this problem.
- Write the first letter (only) of your last name in the box to the right to assist us with sorting exams while relatively maintaining anonymity during grading.

First Letter of Last Name: