COS 226	Algorithms and Data Structures	Fall 2008

Final

This test has 12 questions worth a total of 100 points. You have 180 minutes. The exam is closed book, except that you are allowed to use a one page cheatsheet (8.5-by-11, both sides, in your own handwriting). No calculators or other electronic devices are permitted. Give your answers and show your work in the space provided. **Write out and sign the Honor Code pledge before turning in the test.**

"I pledge my honor that I have not violated the Honor Code during this examination."

Problem	Score
1	
2	
3	
4	
5	
6	
Sub 1	

Problem	Score
7	
8	
9	
10	
11	
12	
Sub 2	

Total	

Name:

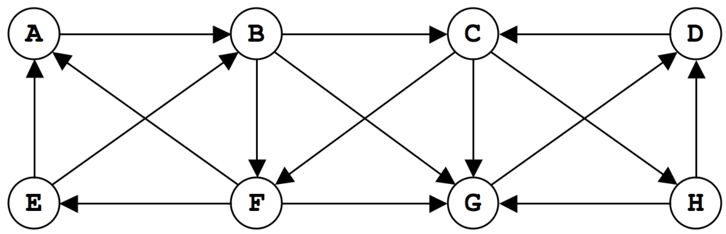
Login ID:

Precept:

P01	12:30	Boaz
P02	3:30	Boaz

1. **Depth-first search. (10 points)**

   Run *depth-first search* on the digraph below, starting at vertex A. As usual, assume the adjacency sets are in lexicographic order, e.g., when exploring vertex F, the algorithm considers the edge $F \to A$ before $F \to E$ or $F \to G$.



(a) Complete the *preorder* of the vertices (the order in which the vertices are first visited).

`A   B   C   ___   ___   ___   ___   ___`

(b) Complete the *postorder* of the vertices (the order in which the vertices are last visited).

`___   ___   ___   ___   ___   C   B   A`

(c) Identify each of the following statements as *true* or *false*. A vertex s is a *source* if there is a directed path from s to every vertex in the graph. An edge $v \to w$ is *postorder decreasing* if w appears before v in the postorder.

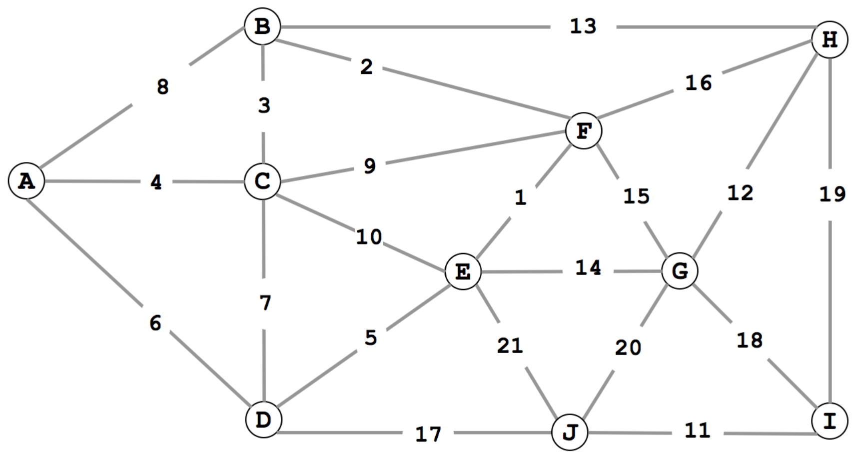
___ For every digraph G and source s, if we execute DFS from s then all edges in G are postorder decreasing.

___ For every digraph G and source s, if we execute DFS from s and all edges in G are postorder decreasing, then G is a DAG.

___ For every *DAG* G and source s, if we execute DFS from s then all edges in G are postorder decreasing.

2. **Minimum spanning tree. (8 points)**

   Consider the following weighted graph with 10 vertices and 21 edges. Note that the edge weights are distinct integers between 1 and 21.



   (a) Complete the sequence of edges in the MST in the order that *Kruskal's algorithm* includes them.

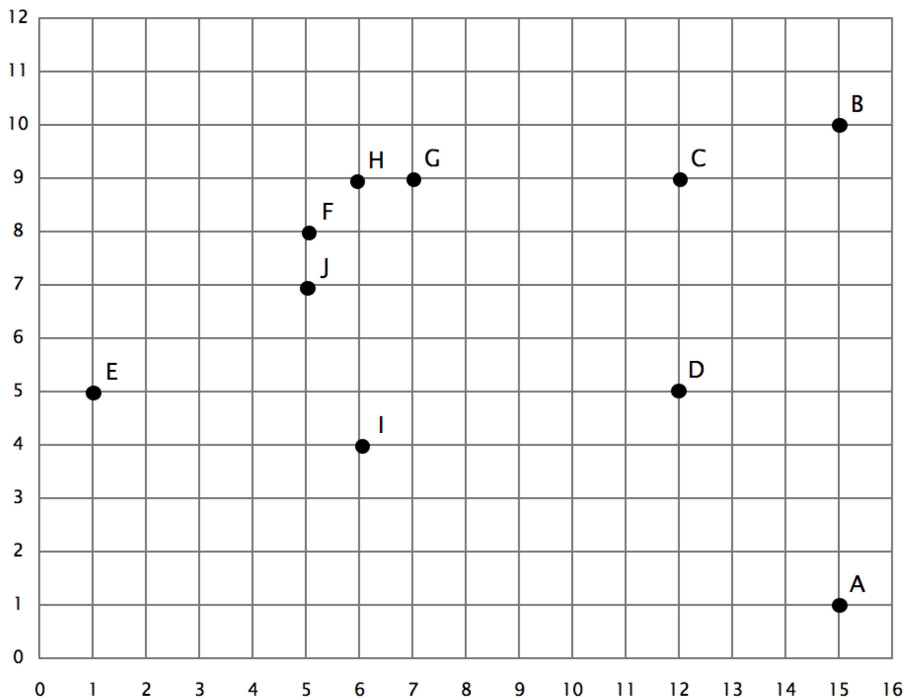
   1 2 3 ____ ____ ____ ____ ____ ____

   (b) Complete the sequence of edges in the MST in the order that *Prim's algorithm* includes them. Start Prim's algorithm from vertex *A*.

   4 3 2 ____ ____ ____ ____ ____ ____

3. **Convex hull. (8 points)**

Run the Graham scan algorithm to compute the convex hull of the 10 points below.



Give the points in counterclockwise order relative to the base point `A`.

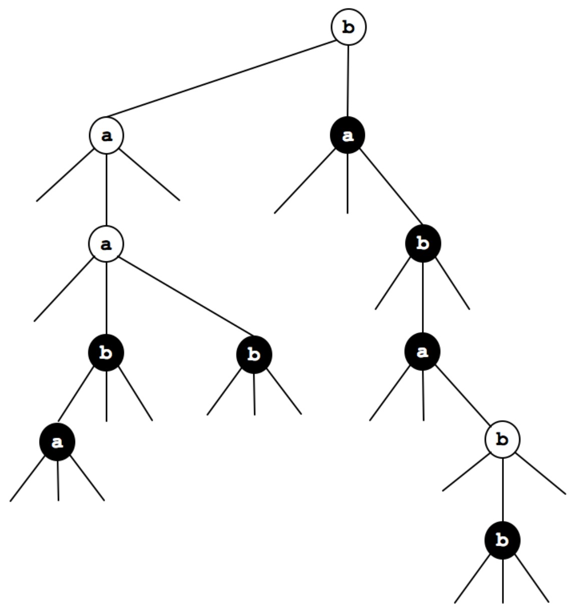
```
B   C   D   ----   ----   ----   ----   ----   E
```

Give the points that appear on the trial hull (after each of the remaining iterations).

```
1.   A -> B -> C

2.   A -> B -> C -> D

3.

4.

5.

6.

7.

8.
```

4. **TST. (8 points)**

   Below is the result of inserting a set of strings into a ternary search trie.
   A black node indicates the termination of a string.

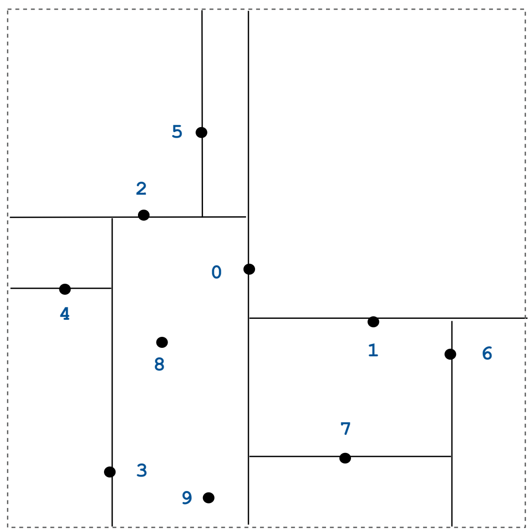


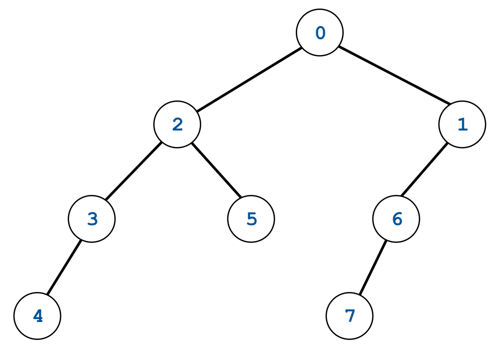
(a) List (in alphabetical order) the set of strings that were inserted.

(b) Add the string `aaca` to the TST and draw it in the figure above.

5. **2D tree. (8 points)**

Below is the results of inserting the points 0 through 7 in that order into a 2D tree.





Inserting the points 8 and 9 (in that order) into the 2D tree above and draw the resulting tree.

6. **Radix sorting. (8 points)**

   Put an `X` in each box if the string sorting algorithm (the standard version considered in class) has the corresponding property.

	mergesort	LSD radix sort	MSD radix sort	3-way radix quicksort
stable				
in-place				
sublinear time (in best case)				
fixed-length strings only				

7. **Data compression. (8 points)**

You receive the following message encoded using LZW compression.

```
 97
 98
128
129
131
132
130
```

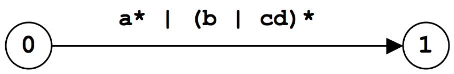
Finish decoding the message.

`a b a b b a ___ ___ ___ ___ ___ ___ ___ ___ ___ ___`

8. **Regular expressions. (8 points)**

   Convert the RE `a* | (b | c d)*` into an equivalent NFA using the algorithm described in lecture, showing the result after applying each transformation.

   1.

   2.

   3.

   4.

   5.

   6.

9. **1D nearest neighbor. (8 points)**

The *1D nearest neighbor* data structure has the following API.

- *constructor:* create an empty data structure.
- *insert(x):* insert the real number x into the data structure.
- *query(y):* return the real number in the data structure that is closest to y (or null if no such number).

Design a data structure that performs each operation in logarithmic time in the worst-case.

Your answer will be graded on correctness, efficiency, clarity, and succinctness. You may use any of the data structures discussed in this course provided you clearly specify it.

constructor:

insert(x):

query(y):

10. **Prefix-free codes. (8 points)**

In data compression, a set of binary code words is *prefix-free* if no code word is a prefix of another. For example, $\{01, 10, 0010, 1111\}$ is prefix free, but $\{01, 10, 0010, 10100\}$ is not because 10 is a prefix of 10100.

(a) Design an efficient algorithm to determine if a set of binary code words is prefix-free.

Your answer will be graded on correctness, efficiency, clarity, and succinctness.

(b) What is the order of growth of the worst-case running time of your algorithm as a function of N and W, where N is the number of binary code words and W is the total number of bits in the input?

N $\qquad$ W $\qquad$ $N \log N$ $\qquad$ NW $\qquad$ $NW \log N$ $\qquad$ N^2

(c) What is the order of growth of the memory usage of your algorithm?

N $\qquad$ W $\qquad$ $N \log N$ $\qquad$ NW $\qquad$ $NW \log N$ $\qquad$ N^2

11. **Shortest directed cycle. (8 points)**

Given a directed graph with V vertices and E edges, design an efficient algorithm to find a directed cycle with the minimum number of edges (or report that the graph is acyclic).

Your answer will be graded on correctness, efficiency, clarity, and succinctness. For full credit, your algorithm should run in $O(EV)$ time and use $O(E+V)$ space. Assume $V \leq E \leq V^2$.

(a) Describe your algorithm in the space below.

(b) What is the order of growth of the worst-case running time of your algorithm?

$E+V$ $\quad V^2$ $\quad EV$ $\quad E^2$ $\quad EV^2$

(c) What is the order of growth of the memory usage of your algorithm?

$E+V$ $\quad V^2$ $\quad EV$ $\quad E^2$ $\quad EV^2$

12. **Reductions. (10 points)**

Consider the following two problems:

- 3SUM. Given N integers $x_1, x_2, \ldots, x_N$, are there three distinct indices i, j, and k such that $x_i + x_j + x_k = 0$?

- 3SUMPLUS. Given N integers $x_1, x_2, \ldots, x_N$ and an integer b, are there three distinct indices i, j, and k such that $x_i + x_j + x_k = b$?

(a) Show that 3SUM linear-time reduces to 3SUMPLUS. To demonstrate your reduction, give the 3SUMPLUS instance that you would construct to solve the following 3SUM instance: $x_1, x_2, \ldots, x_N$.

(b) Show that 3SUMPLUS linear-time reduces to 3SUM. To demonstrate your reduction, give the 3SUM instance that you would construct to solve the following 3SUMPLUS instance: $b, x_1, x_2, \ldots, x_N$.

(c) Suppose that Professor Stewart discovers a $N^{1.9}$ lower bound for 3SumPlus and Professor Colbert discovers a $N^{1.9}$ algorithm for 3Sum. Which of the following can you infer from the fact that 3Sum linear-time reduces to 3SumPlus and vice versa?

I. There does not exist an $N^{1.5}$ algorithm for 3SumPlus.

II. There does not exist an $N^{1.5}$ algorithm for 3Sum.

III. There exists an $N^{1.9}$ algorithm for 3SumPlus.

(a) I only.

(b) I and II only.

(c) I and III only.

(d) I, II and III.

(e) None.