COS 226	Algorithms and Data Structures	Fall 2009

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
0	
1	
2	
3	
4	
5	
Sub 1	

Problem	Score
6	
7	
8	
9	
10	
11	
Sub 2	

Total	

Name:

Login ID:

Precept:

P01	12:30	Anuradha
P02	3:30	Berk
P03	2:30	Corey

0. **Miscellaneous. (1 point)**

   Write your name and Princeton NetID in the space provided on the front of the exam, and circle your precept number.

1. **Analysis of algorithms. (15 points)**

   (a) Which of the following can be performed in *linear time* in the *worst case*?
   Write *P* (possible), *I* (impossible), or *U* (unknown).

   ___ Printing the keys in a binary search tree in ascending order.

   ___ Finding a minimum spanning tree in a weighted graph.

   ___ Finding all vertices reachable from a given source vertex in a graph.

   ___ Checking whether a digraph has a directed cycle.

   ___ Building the Knuth-Morris-Pratt DFA for a given string.

   ___ Sorting an array of strings, accessing the data solely via calls to `charAt()`.

   ___ Sorting an array of strings, accessing the data solely via calls to `compareTo()`.

   ___ Finding the closest pair of points among a set of points in the plane, accessing the data solely via calls to `distanceTo()`.

   (b) Match up each operation with the best description of its running time.

   ___ Insert into a red-black tree. A. $\log N$ worst case

   ___ Insert into a 2d-tree. B. $\log N$ amortized

   ___ Insert into a binary heap. C. $\log N$ average case on random inputs

(c) The order-of-growth of the running time of one algorithm is N^2; the order-of-growth of the running time of a second algorithm is N^3. List two compelling reasons why a programmer would prefer to use the N^3 algorithm instead of the N^2 one.

-

-

(d) How many bytes does each `Node` object consume? Include all memory allocated by a call to `new Node(x, y, z)`. As usual, assume the following values for memory in Java: `int` (4 bytes), `double` (8 bytes), reference (4 bytes), object overhead (8 bytes).

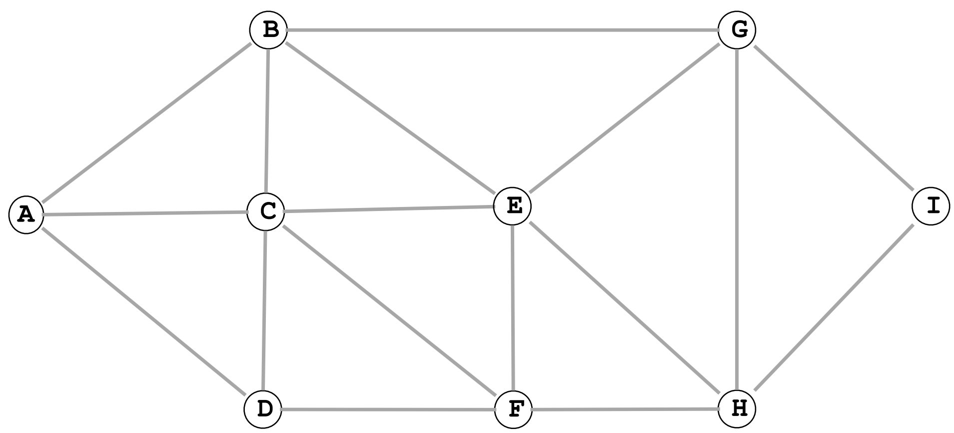
```
public class Node {
    private Node left, right;
    private int count;
    private Point3D point;

    public Node(double x, double y, double z) {
        left  = null;
        right = null;
        count = 0;
        point = new Point3D(x, y, z);
    }
    ...
}

public class Point3D {
    private double x, y, z;
    public Point3D(double x, double y, double z) {
        this.x = x;
        this.y = y;
        this.z = z;
    }
    ...
}
```

2. **Breadth-first search. (8 points)**

(a) Run *breadth-first search* on the graph below, starting at vertex A. As usual, assume the adjacency sets are in sorted order, e.g., when exploring vertex F, the algorithm considers the edge F-C before F-D, F-E, or F-H.



List the vertices in the order in which the vertices are enqueued on the FIFO queue.

A B --- --- --- --- --- --- ---

(b) Consider two vertices x and y that are simultaneously on the FIFO queue at some point during the execution of breadth-first search from s in an undirected graph. Which of the following are true?

I. The number of edges on the shortest path between s and x is at most one more than the number of edges on the shortest path between s and y.

II. The number of edges on the shortest path between s and x is at least one less than the number of edges on the shortest path between s and y.

III. There is a path between x and y.

(a) I only.

(b) I and II only.

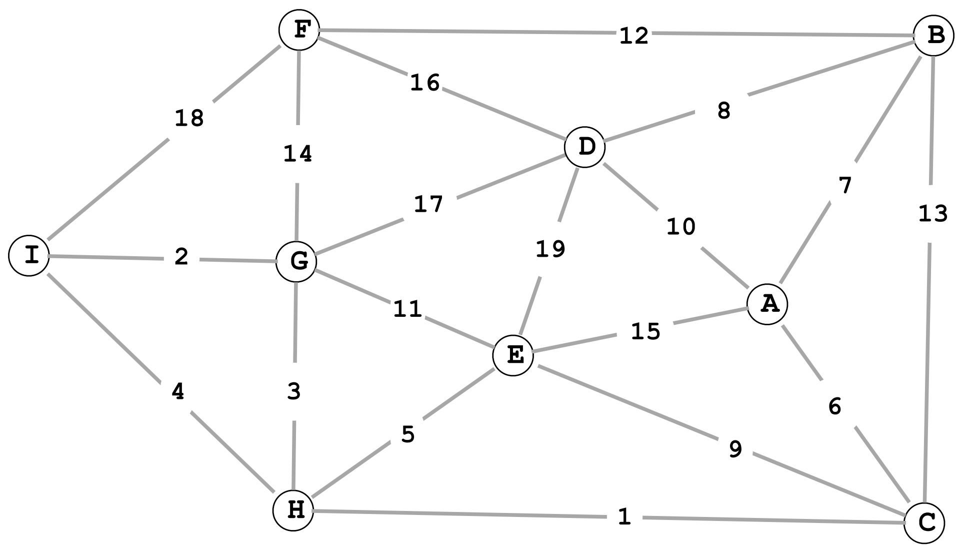
(c) I and III only.

(d) I, II and III.

(e) None.

3. **Minimum spanning tree. (10 points)**

   For parts (a), (b), and (c), consider the following weighted graph with 9 vertices and 19 edges. Note that the edge weights are distinct integers between 1 and 19.

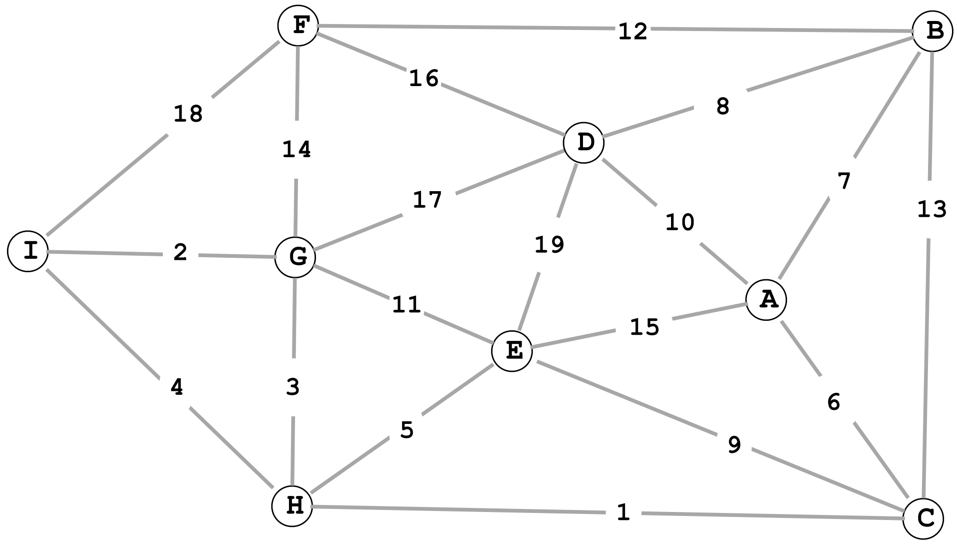


   (a) Complete the sequence of edges in the MST in the order that *Kruskal's algorithm* includes them.

   1 ---- ---- ---- ---- ---- ---- ----

   (b) Suppose that the edge *D-I* of weight *w* is added to the graph. For which values of *w* is the edge *D-I* in a MST?

The weighted graph from the previous page is repeated here for reference.



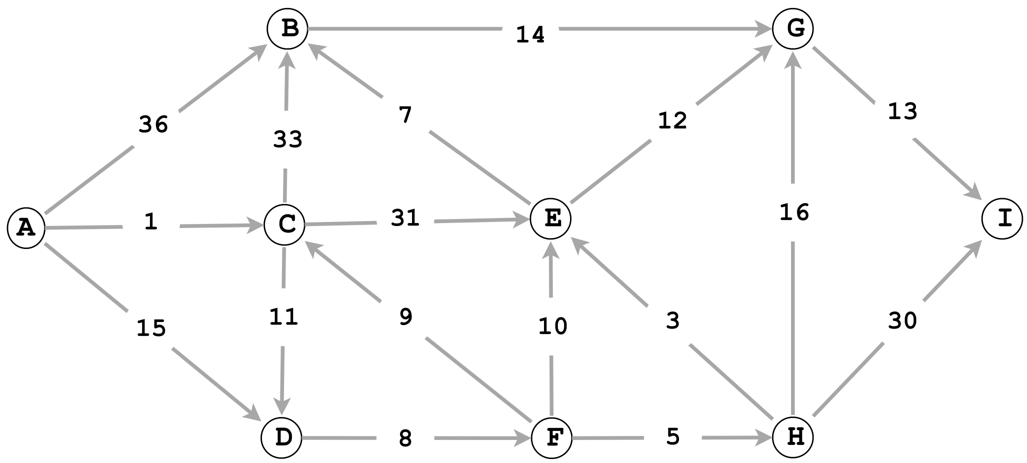
(c) Complete the sequence of edges in the MST in the order that *Prim's algorithm* includes them. Start Prim's algorithm from vertex A.

6 ---- ---- ---- ---- ---- ---- ----

(d) Given a minimum spanning tree T of a weighted graph G, describe an $O(V)$ algorithm for determining whether or not T remains a MST after an edge x-y of weight w is added.

4. **Shortest paths. (8 points)**

Run *Dijkstra's algorithm* on the weighted digraph below, starting at vertex *A*.



(a) List the vertices in the order in which the vertices are dequeued (for the first time) from the priority queue and give the length of the shortest path from *A*.

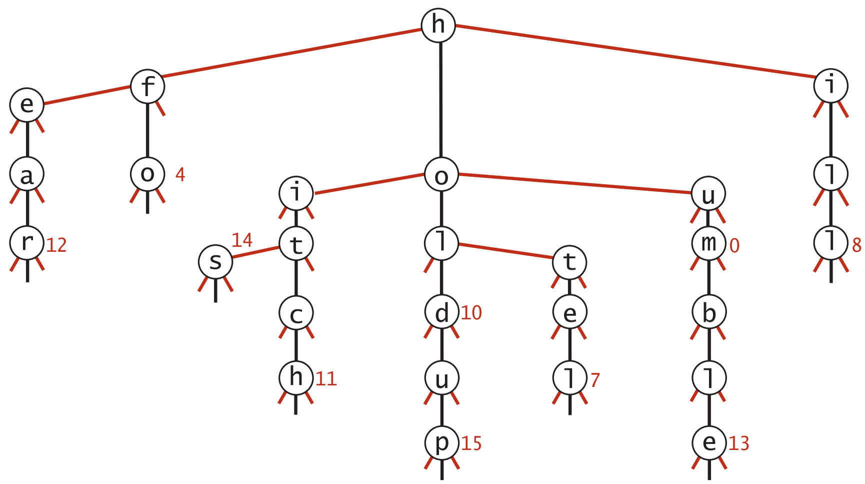
```
vertex:      A    C   ___   ___   ___   ___   ___   ___   ___

distance:    0    1   ___   ___   ___   ___   ___   ___   ___
```

(b) Draw the edges in the shortest path tree with thick lines in the figure above.

5. **Ternary search tries. (8 points)**

   Below is the result of inserting a set of strings (and associated integer values) into a ternary search trie.



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

   (b) Add the string `hoho` (with associated value 77) and then add the string `horse` (with the associated value 88) to the TST and draw the results in the figure above.

   (c) List two compelling reasons why a programmer would use a TST instead of a red-black tree.

   -

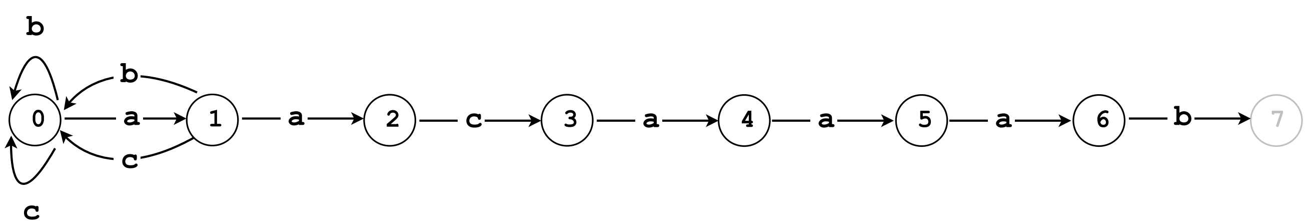
   -

6. **Substring search. (8 points)**

Create the Knuth-Morris-Pratt DFA for the string `aacaaab` over the alphabet { `a, b, c` } by completing the following table. As usual, state 0 is the start state and state 7 is the accept state.

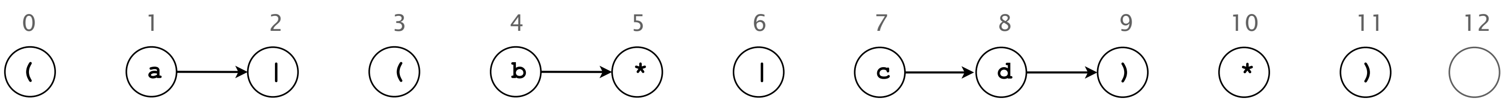
	0	1	2	3	4	5	6
a	1	2		4	5	6	
b	0	0					7
c	0	0	3				

You may use the following partially-completed graphical representation of the DFA for scratch work (but we will consider your solution to be the completed table above).



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

Convert the regular expression `( a | ( b * | c d ) * )` into an equivalent NFA (nondeterministic finite state automaton) using the algorithm described in lecture by adding ϵ-transition edges to the diagram below.



8. **Burrows-Wheeler transform. (8 points)**

   (a) What is the Burrows-Wheeler transform of

```
b a b a a b a c
```

   (b) What is the Burrows-Wheeler inverse transform of

```
7
b b b b a a a a a
```

9. **Circular suffixes. (6 points)**

Consider the following three Java code fragments for forming the circular suffixes of a string `s` of length `N`.

I.

```
s = s + s;
String[] circularSuffixes = new String[N];
for (int i = 0; i < N; i++)
   circularSuffixes[i] = s.substring(i, N+i);
```

II.

```
String[] circularSuffixes = new String[N];
for (int i = 0; i < N; i++)
   circularSuffixes[i] = s.substring(i, N) + s.substring(0, i);
```

III.

```
StringBuilder sb = new StringBuilder();
sb.append(s);
sb.append(s);
String[] circularSuffixes = new String[N];
for (int i = 0; i < N; i++)
   circularSuffixes[i] = sb.substring(i, N+i);
```

Which of the above code fragments consumes space proportional to N?

(a) I only.

(b) I and II only.

(c) I and III only.

(d) I, II and III.

(e) None.

10. **Tandem repeats. (10 points)**

A *tandem repeat* of a base string `b` within a string `s` is a substring of `s` consisting of at least one consecutive copy of the base string `b`. Given `b` and `s`, design an algorithm to find a tandem repeat of `b` within `s` of maximum length.

For example, if `s` is `"abcabcababcaba"` and `b` is `"abcab"`, then `"abcababcab"` is the tandem substring of maximum length (2 copies).

Your answer will be graded on correctness, efficiency, clarity, and succinctness. Let M denote the length of `b` and let N denote the length of `s`. For full credit, your algorithm should take time proportional to $M + N$.

(a) Describe your algorithm in the space below.

(b) What is the worst-case running time of your algorithm as a function of M and N? Circle the best answer.

N M $M + N$ MN N^2 M^2 other _____

11. **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 $x_i + x_j + x_k = 0$?

- 4SUM. Given N integers $x_1, x_2, \ldots, x_N$, are there four distinct indices i, j, k, and l such that $x_i + x_j + x_k + x_l = 0$?

(a) Show that 3SUM linear-time reduces to 4SUM. To demonstrate your reduction, give the 4SUM instance that you would construct to solve the following 3SUM instance: $x_1, x_2, \ldots, x_N$.

(b) Suppose that Alice discovers an $N^{1.9}$ algorithm for 3SUM and Bob discovers an $N^{1.9}$ lower bound for 4SUM. Which of the following can you infer from the fact that 3SUM linear-time reduces to 4SUM?

I. There does not exist an $N^{1.8}$ algorithm for 3SUM.

II. 3SUM and 4SUM have the same asymptotic complexity.

III. There exists an $N^{1.9}$ algorithm for 4SUM.

(a) I only.

(b) I and II only.

(c) I and III only.

(d) I, II and III.

(e) None.