ICS 161 — Algorithms — Winter 1998 — Second Midterm

Name:	
ID:	
1:	out of 5
2:	out of 15
3:	out of 20
4:	out of 10
5:	out of 20
6:	out of 30
total:	out of 100

1. Multiple-choice: who of these researchers was the first to invent the Prim-Dijkstra minimum spanning tree algorithm.

(a) Prim?

(b) Dijkstra?

(c) Boruvka?

(d) Jarník?

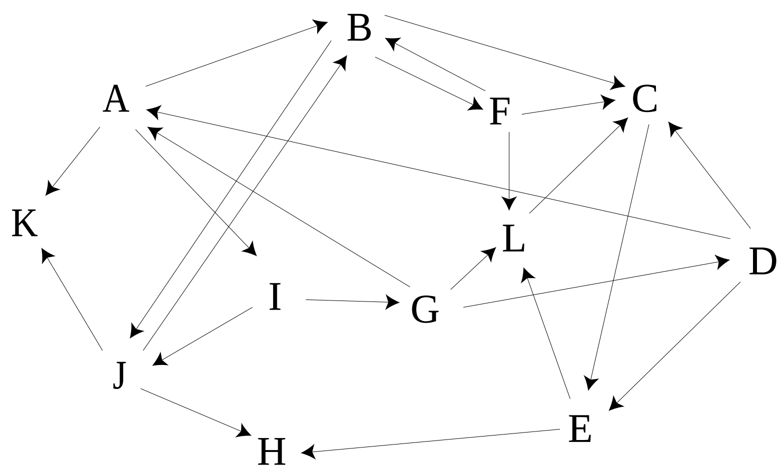
(5 points)

2. Give an example of a weighted undirected graph G and a starting vertex s in G such that the minimum spanning tree of G is not the same as the shortest path tree (starting from s).

Show both trees.

(15 points: 5 for a correct graph, 5 for each tree)

3. We say that a vertex in a directed graph is “looping” if it is part of a cycle of two or more vertices. For instance, in the following graph, B and J are part of a two-vertex cycle, and in fact all the vertices except for K and H are looping:



(a) How is the property of being looping related to strongly connected components?

(b) Describe a linear-time algorithm that lists all of the looping vertices of a graph.

(If you use as a subroutine an algorithm already described in class, just mention that algorithm by name, rather than repeating its description.)

(20 points: 10 for each part)

4. Recall that, in the version of the KMP string matching algorithm described in class, overlap[0] is always -1 and overlap[1] is always 0. But, overlap[2] can be either 0 or 1.

If overlap[2]=0, what does this imply about the characters of the pattern?

(10 points)

5. What values would the version of the KMP string matching algorithm described in class store in the overlap[] array, if the pattern string is “nanobanana”?

(20 points)

6. You don't need to read the next paragraph to solve this problem, but it may help you understand what the code does.

The following recursive algorithm takes as input an n-item array A, which we interpret as storing a complete binary tree in which the two children of A[i] are A[2i+1] and A[2i+2]. In this tree, we say that a set of vertices is "independent" if it does not contain any parent-child pair. The algorithm returns the largest weight (sum of values) of any independent subset of the descendants of A[i]. When called with i=0, it returns the maximum weight of any independent set.

```
int max_independent_set(int i)
{
    if (i >= n) return 0;
    return max( max_independent_set(2*i+1)
                 + max_independent_set(2*i+2),
                A[i] + max_independent_set(4*i+3)
                 + max_independent_set(4*i+4),
                 + max_independent_set(4*i+5),
                 + max_independent_set(4*i+6) );
}
```

(a) Convert this code to a top-down (memoizing) dynamic programming algorithm.

(continued on next page)

6. (continued)

(b) Convert this code into a bottom-up (iterative) dynamic programming algorithm.

(c) How much time does your dynamic programming algorithm take, as a function of n? (The answer should be the same for parts (a) and (b), so only answer once.)

(30 points: 10 for each part)