

CMPUT 204 Winter 2001: Section B2

Final Exam

Wednesday, Apr. 18

Time: 120 minutes

Last name:
First name:
UNIX ID:

- A handwritten $8\frac{1}{2} \times 11$ formula sheet is allowed.
- No books or other notes are allowed.
- No calculators or other mechanical devices are allowed.
- Unless otherwise specified the term "graph" in this exam refers to an undirected graph.
- This exam has 8 questions. You are responsible for checking that your exam booklet is complete.
- If you need extra space there is a blank page at the end of the exam. Additional paper is available from a proctor.

Q	Mark	
1		3
2		3
3		5
4		9
5		9
6		9
7		12
8		10
Σ		60

Question 1 [3 points]

Group the following functions into classes, so that all functions with the same asymptotic order are in the same class, and then arrange the classes into increasing order of complexity, from smallest to largest.

$$n^2 \cdot (\ln n)^5 \quad n^5 \cdot (\ln n)^2 \quad n^3 \quad (\ln n)^5 \quad \ln(n^5)$$

Question 2 [3 points]

$$\text{Let } T(n) = \begin{cases} 125 & n = 1 \\ 25 \cdot T(\frac{n}{5}) + n^3 & n > 1 \end{cases}$$

where you may assume n is a power of 5. Give the simplest big- Θ expression for $T(n)$.

Question 3 [5 points]

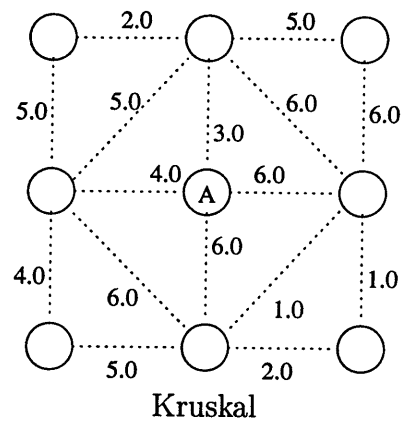
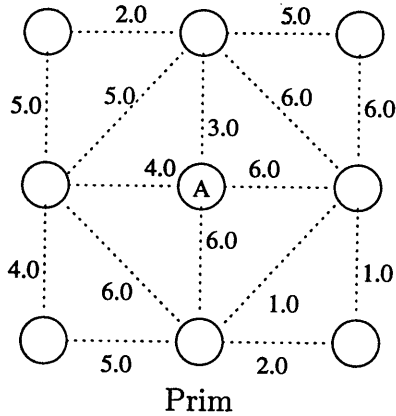
Consider the following code:

```
for i = 1 to n
  for j = i to n
    for k = j to n
      sum += f(i,j,k);
```

Show that the number of calls to function f is $\Theta(n^3)$.

Question 4 [9 points]

Two identical copies of a weighted graph appear below.



4.a [3 pts]: Apply Prim's algorithm to the graph on the right, starting at vertex A. Shade the edges that are incorporated into the MST, and number them in the order in which they are added to the tree. If more than one edge may be selected, arbitrarily choose one.

4.b [3 pts]: Repeat part a applying Kruskal's algorithm to the graph on the left.

4.c [3 pts]: If a priority queue used in Prim's algorithm which has the following characteristics:

- cost of inserting an element is $O(1)$
- cost of getting/deleting minimal element is $O(n)$
- cost of decreasing a key is $O(\lg n)$

then what is the worst-case complexity of this particular implementation of Prim?

Question 5 [9 points]

G is a graph such that a depth first search starting from vertex A yields the depth first search tree with root A, tree edges (AB) (BC) (CD) (CE) (BF) (FH) (AI), and traversal order ABCDEFHI.

5.a [2 pts]: Is it possible for (DI) to be an edge of G? Explain why or why not.

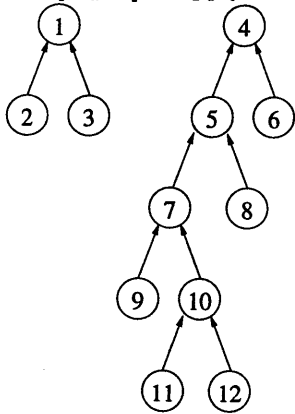
5.b [2 pts]: Assuming that (DB), (CA) and (HB) are the only back edges of the given DFS tree, draw this tree, showing back edges as dotted lines.

5.c [5 pts]: Again assuming that (DB), (CA) and (HB) are the only back edges of the DFS tree, give the output from the biconnected components algorithm. List the biconponents in order of discovery. Each line of output should list all edges in one bicomponent. (The order of the lines is important; the order of the edges within a line is not.)

Question 6 [9 points]

Recall how dynamic equivalence relations are implemented with Union-Find procedures on forests of in-trees.

6.a [2 pts]: Apply $wUnion(1,4)$ to the forest below. Draw the results.



6.b [2 pts]: Apply $cFind(10)$ to your output of part a. Draw the results.

6.c [5 pts]: Suppose that Union-Find is implemented with weighted union together with the basic find (without path compression).

Give a sequence of m union-find instructions which require $\Omega(m \log m)$ time to execute.

Briefly explain why your idea works.

Question 7 [12 points]

Consider the following nonsense algorithm. The input is an array, and the solution depends upon recursively applying the same algorithm to sub-arrays.

Arrays are indexed starting at 1.

```
float fuzzleWrapper(float[] data){
    return fuzzle(1, data.size(), data)
}

float fuzzle(int low, int high, float[] data) {
    if (low>high)
        best = 0.0;
    else {
        best = infinity;
        for i = 0 to high-low {
            mid = low+i;
            x = fuzzle(low,mid-1,data);
            y = fuzzle(mid+1,high,data);
            cost = 2*x + data[mid] + 2*y;
            best = min(best,cost);
        }
    }
    return best;
}
```

7.a [3 pts]: State (but do not solve) a recurrence relation for $T(n)$, the number of array accesses in fuzzle, as a function of, n , the size of the sub-array (n is the value $high-low+1$).

7.b [3 pts]: It is known that $T(n) \geq 2 \cdot T(n - 1)$. Use this inequality to show that $T(n) \in \Omega(2^n)$.

7.c [3 pts]: What is the main goal of converting a recursive algorithm to a dynamic programming algorithm?

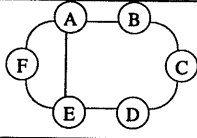
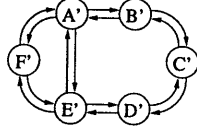
7.d [3 pts]: What is the complexity of the dynamic programming version of fuzzle? Briefly justify your answer.

Question 8 [10 points]

Suppose we know that the Hamiltonian Cycle problem on undirected graphs is NP-complete. Show that the Directed Hamiltonian Cycle problem is also NP-complete.

Hint: Start by stating all the steps required to show a problem is NP-complete. Many of the marks are for clearly indicating how to show a problem is NP-complete. To obtain full marks, you will need a simple reduction, which is strongly hinted at in the example.

A summary of the two decision problems is:

	Hamiltonian Cycle	Directed Hamiltonian Cycle
Input	An undirected graph $G = (V, E)$	A directed graph $G' = (V', E')$
Question	Does there exist a cycle in G which goes through every vertex exactly once?	Does there exist a <u>directed</u> cycle in G' which goes through every vertex exactly once?
Example input		
Answer for example input	Yes. (since $ABCDEF$ is a cycle in G)	Yes. (since $A'B'C'D'E'F'$ is a cycle in G')

