

Second Midterm Exam

CSE 101

Name: _____

ID: _____

Be clear and concise. Write your answers in the space provided. Use the backs of pages for scratchwork.

1	
2	
3	
4	

1. **(10 points)** Write down the answer to each question in the given box. No justification is required.

(a) A tree T has 100 nodes. What is the minimum number of leaves it could possibly have?

Answer:

(b) An undirected graph G has n nodes and k connected components. What is the minimum number of edges it could possibly have, as a function of n and k ?

Answer:

(c) How many lines, as a function of n (in $\Theta(\cdot)$ form), does the following program print? You may assume n is a power of 2.

```
function f(n)
  if n > 1:
    print_line('still going')
    f(n/2)
    f(n/2)
```

Answer:

(d) What is the solution to the recurrence

$$T(n) = T(n/2) + n^2, \quad T(1) = 1,$$

in big- Θ notation?

Answer:

2. **(10 points)** Given a sorted array of distinct integers $A[1, \dots, n]$, you want to find out whether there is an index i for which $A[i] = i$. Give a divide-and-conquer algorithm that runs in time $O(\log n)$.

Algorithm:

Justification of correctness:

Running time analysis:

3. (10 points) For each of the following statements, say whether it is **true** or **false**. If **true**, give a brief proof. If **false**, give a counterexample.

(a) Suppose G is an undirected, connected graph. If its lightest edge is unique, then this edge must be part of *every* MST.

(b) Suppose undirected, connected graph G has *two* lightest edges, and the remaining edges have distinct weights. Then G must have exactly two distinct MSTs.

(c) Pick any nodes u, v in a connected, undirected graph G with positive edge weights. Then the shortest path between u and v must be part of some MST of G .

4. **(10 points)** Suppose that *instead of* having edge lengths $\{l_e : e \in E\}$, a graph has *vertex costs* $\{c_v : v \in V\}$. Now define the cost of a path to be the sum of all vertex costs on the path (including the endpoints). Give an efficient algorithm for the following problem.

Input: A directed graph $G = (V, E)$; *possibly negative* vertex costs c_v ; starting vertex $s \in V$.

Output: An array `cost[.]` such that for every vertex u , `cost[u]` is the least cost of any path from s to u (i.e., the cost of the cheapest path), under the definition above.

You may assume that there are no cycles with net negative cost. Notice that `cost[s] = cs`.

Algorithm:

Justification of correctness:

Running time analysis: