Algorithm Final Sample Answer

1.

(a)
If Z is not a "longest" common sequence of Xm-1 and Y, then there were a common sequnce W of Xm-1 and Y longer than Z.
Therefore, W would also be a common sequence of X and Y with longer length than Z, which contradict to "Z is an LCS of X and Y".

(b)

	j	0	1	2	3	4	5	6	7	8
i		y	0	1	0	1	0	0	0	1
0	x	0	0	0	0	0	0	0	0	0
1	0	0	↖ 1	← 1	↖ 1	← 1	↖ 1	↖ 1	↖ 1	← 1
2	0	0	↖ 1	↑ 1	↖ 2	← 2	↖ 2	↖ 2	↖ 2	← 2
3	1	0	↑ 1	↖ 2	↑ 2	↖ 3	← 3	← 3	← 3	↖ 3
4	0	0	↖ 1	↑ 2	↖ 3	↑ 3	↖ 4	↖ 4	↖ 4	← 4
5	1	0	↑ 1	↖ 2	↑ 3	↖ 4	↑ 4	↑ 4	↑ 4	↖ 5
6	1	0	↑ 1	↖ 2	↑ 3	↖ 4	↑ 4	↑ 4	↑ 4	↖ 5
7	1	0	↑ 1	↖ 2	↑ 3	↖ 4	↑ 4	↑ 4	↑ 4	↖ 5

2.

Assume the amortized cost of each insertion is $\$3$.

When a table of size \(T\) expands, it means \(T/2\) values have been inserted since the last expansion.

Each of these \(T/2\) insertions is charged $\(3\), resulting in a total cost of \((T/2) \times 3 = 1.5T\). The actual cost of each insertion is $\(1\), so the actual insertion costs are \((T/2) \times 1 = 0.5T\).

Therefore, the stored credit is \(1.5T - 0.5T = T\). This credit is used to pay for the expansion, which costs \(T\) units.

Thus, the accumulated credits from the previous insertions are enough to cover the cost of copying all \(T\) elements during the expansion.

3.

(a)

1. First, we need to prove that the edge selected by the algorithm each time is the ***light edge*** of some cut. (you will receive full score if you have proven this part)

- At the moment when the algorithm select e = (u, v), let S be the set of nodes reachable from u.
- Claim v will be in V-S.
  because if it were in S, it would form a cycle according to the definition of algorithm, leading to a contradiction.
- Then, by the definition of S, no picked edge cross (S, V-S) before e .
- and by the definition of the alorithm, no edge cheaper than e that cross from (S, V-S).
- So, e is the light edge cross (S, V-S)

2. With the theorem (or the corollary) in the textbook, since the light edge e is a safe edge for MST, this concludes that the algorithm can correctly find the MST. (you would get some bonus score if you have proven or mentioned this theorem when you don't get full score in previous part)

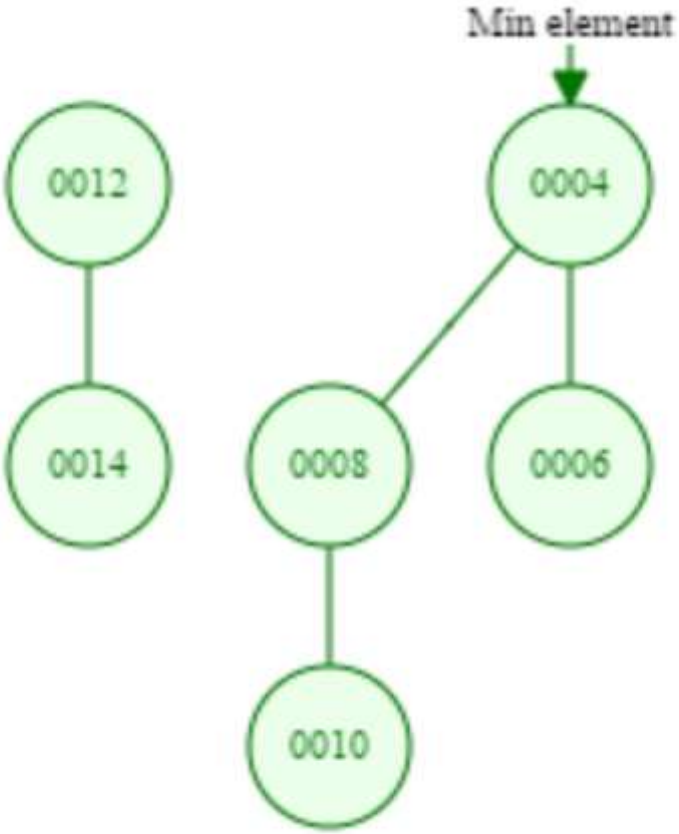
(b)

```
O(mlogm + m*α(n)) = O(mlogm) = O(mlogn)
```

4.

(a)

(b)



5.

The time spent on union is primarily due to the number of times pointers are updated for all the nodes.
Due to the weighting rule, the length after each merge is at least twice the length of the shorter list.
So each pointer is updated at most ceil(lgn) times over all the UNION operations.
Thus the total time spent updating object pointers over all UNION operations is O(nlgn).

6.

Prove by induction - 4%

7.

(a)

```
a -> c
b
c
d -> e
e -> c
f -> g
g
s -> x -> b -> a
x -> b
```

(b)
one of possible examples:

```
s, x, b, a, d, e, c, f, g
```

(c)
Yes, since G is a directed acyclic graph, we can find the longest (not necessary simple) path of any reachable pair of vetices.

8.

No, it's probably not easy. If the problem is easy, we can check each \(k\) for the optimal solution for the vertex covering problem, which will make it easy to solve. However, there's no known polynomial time solution for NP-complete problems.

9.

(a) A to B
If we want to prove that B is NP-Hard, we need to show for every NP problem, there is a polynomial-time reduction to B. That is we need to reduce an NP-Complete problem A to problem B.

(b) A to B
For problem A, there is no algorithm can solve it with upper bound lower than O(nlogn).
If we can reduce problem A to B, it means we can solve A using an algorithm that solves B.
That implies there is no algorithm can solve B with upper bound lower than O(nlogn) which means B has lower bound Ω(nlogn).

10.

Theorem (Zero-One Principle)

If a comparison network with n inputs sort all 2^n possible sequence of 0's and 1's, then it sorts all sequences of arbitrary number correctly.