Tutorial

Instructor: Meng-Fen Chiang

COMPCSI220: WEEK 9

https://ankechiang.github.io/cs220_swu.html#week9

OUTLINE

- Question 1: Complexity
- Question 2: Complexity
- Question 3: Mergesort
- Question 4: Quicksort
- Question 5: Heapsort
- Question 6: Binary Search Tree
- Question 7: Graph/Digraph
- Question 8: Graph Data Structure

Question 1

- Prove that $T(n) = 3n^3 + 2n^2 + n + 90$ is both $O(n^3)$ and $O(n^4)$

$\lim_{n\to\infty} \frac{3n^3+2n^2+n+90}{n^3} = 3$. This means $T(n)$ is $\Theta(n^3)$, and is also $O(n^3)$.

$\lim_{n\to\infty} \frac{3n^3+2n^2+n+90}{n^4} = 0$. This means $T(n)$ is $O(n^4)$.

Question 1 (Contd.)

- Determine the asymptotic relationship between $f(n)$ and $g(n)$ using limit rule.

$$f(n) = n\ln(3n) \text{ and } g(n) = 3n^2$$

$\lim_{n\to\infty} \frac{n\ln(3n)}{3n^2} = 0$. This means $f \in O(g)$ and $g \in \Omega(f)$.

Question 2

- Let $T(n) = f(n)g(n) + h(n)$ be a function defined on three running time functions, where $f(n) \in \Theta(\sqrt{n}); g(n) \in \Omega(\log n^n); h(n) \in O(n^3)$. Show that $T(n) \in \Omega(n^{\frac{3}{2}} \log n)$

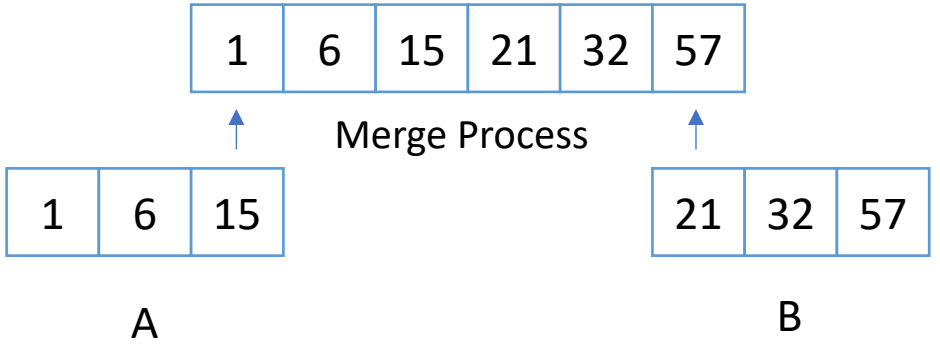
Proof. Because $f(n)$ is $\Theta(\sqrt{n})$, then $f(n)$ has to be $\Omega(\sqrt{n})$. Apply the product rule of Ω, we have $f(n)g(n)$ is $\Omega(n^{3/2} \log n)$. More specifically, we will find constants $n_0 \in \mathbb{N}$ and $C > 0$ such that $f(n)g(n) \geq C \cdot n^{3/2} \log n$ for all $n > n_0$. For $h(n)$, we don't have a lower bound, but we know that any running time function is non negative. Therefore, $T(n) = f(n)g(n) + h(n) \geq C \cdot n^{3/2} \log n$ for all $n > n_0$. This gives us $T(n)$ is $\Omega(n^{3/2} \log n)$.

Product Rule: Assume we have two functions $f_1(n) \in \Omega(g_1(n))$ and $f_2(n) \in \Omega(g_2(n))$, then

- $f_1(n) \geq c_1 \cdot g_1(n)$ for all $n > n_1$, where $n_1 \in \mathbb{N}$ and $c_1 > 0$.
- $f_2(n) \geq c_2 \cdot g_2(n)$ for all $n > n_2$, where $n_2 \in \mathbb{N}$ and $c_2 > 0$.
- $f_1(n)f_2(n) \geq c_1 \cdot c_2 \cdot g_1(n) \cdot g_2(n)$ for all $n > \max\{n_1, n_2\}$. This gives us $f_1(n)f_2(n) \in \Omega(g_1(n)g_2(n))$.

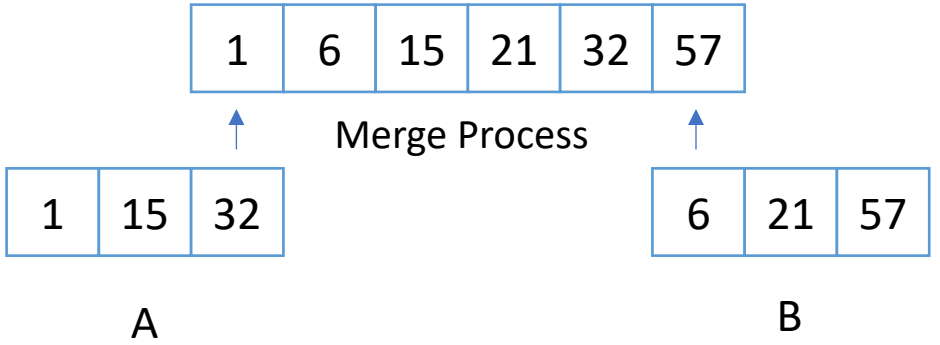
Question 3

- What is the minimum and maximum number of comparisons needed when merging two nonempty sorted lists of size n into a single list?
- **Minimum number of comparisons**: This happens when all elements in one list are smaller than all those in the other list. Denote the two lists as A and B and let's assume all elements in A are smaller than those in B. Then, we only need to compare all element in A with the first element in B. This requires n comparisons.



Question 3

- What is the minimum and maximum number of comparisons needed when merging two nonempty sorted lists of size n into a single list?
- **Maximum number of comparisons**: This happens when we need to zigzag between the two lists. The total number of comparisons is $2n - 1$.



Question 4

- Determine the order of the list after partitioning 23, 20, 6, 17, 13, 25, 14, assume you have a way to take the median of the list as the pivot.

[Note: The answer here will only include all the details for the first partition. We are assuming we can pick the median of the list, 17, in this case as the pivot]

- Step1: swap the pivot 17 with the first element in the list.

17, 20, 6, 23, 13, 25, 14

- Step2: two pointers L and R starting on each end of the list and looks for elements bigger than the pivot and smaller than the pivot respectively. L pointer will find 20 and R pointer will find 14 for the first time. Swap 20 and 14.

17, 14, 6, 23, 13, 25, **20**

Question 4 (Contd.)

- Step3: Continue to move L and R will lead to L finding 23 and R finding 13. Swap 23 and 13.

17, 14, 6, **13, 23**, 25, **20**

- Step4: when R moves to the left again, it will collide with L. This is when you swap this index with the pivot.

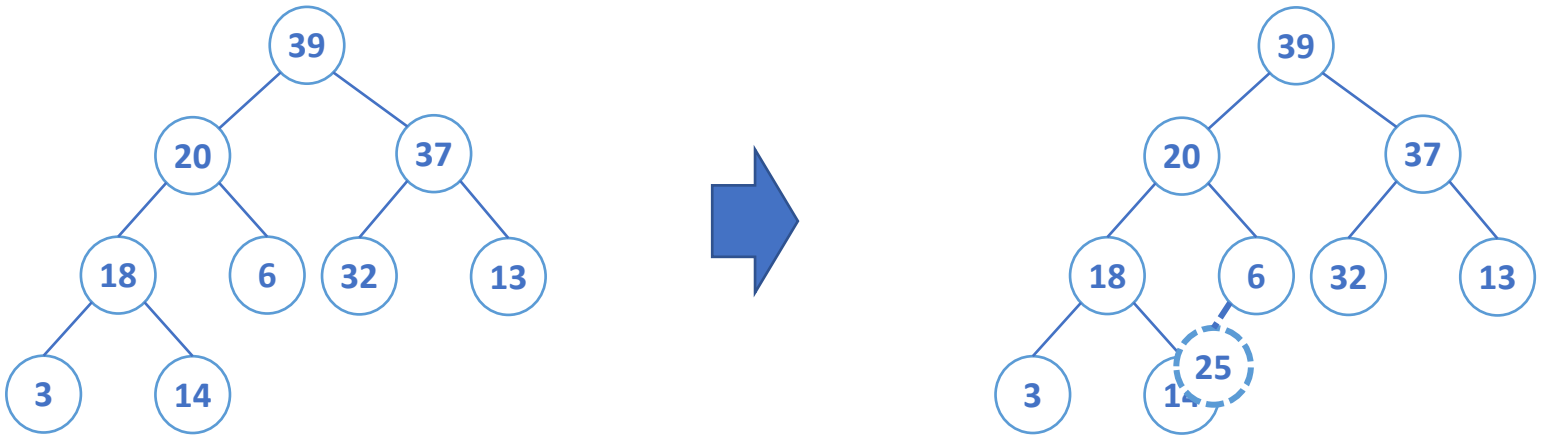
13, 14, 6, **17, 23**, 25, **20**

- At this point, all elements smaller than the pivot are on its left and all elements larger than the pivot are on its right. The first partition is done.

Question 5.1

- Consider the following maximum heap: 39, 20, 37, 18, 6, 32, 13, 3, 14. Insert 25 to the heap.
- Step1: We always insert new nodes into a heap at the next available index so the list will be as follows after the insertion.

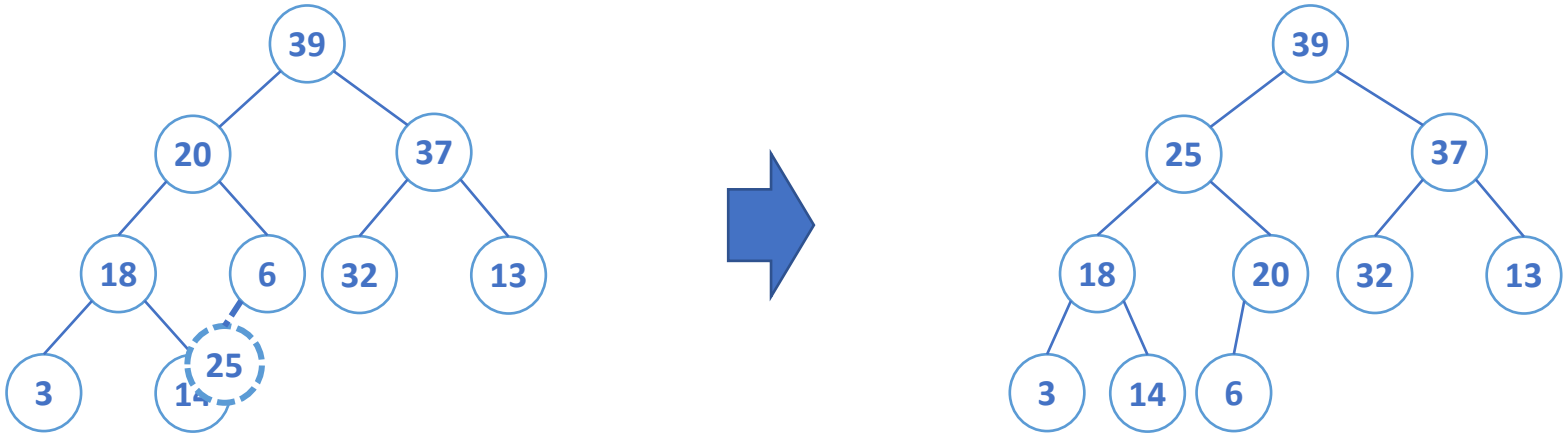
39, 20, 37, 18, 6, 32, 13, 3, 14, 25



Question 5.1 (Contd.)

- Consider the following maximum heap: 39, 20, 37, 18, 6, 32, 13, 3, 14. Insert 25 to the heap.
- Step2: We then bubble the new node up to the correct place repeatedly comparing with its parent node. We obtain

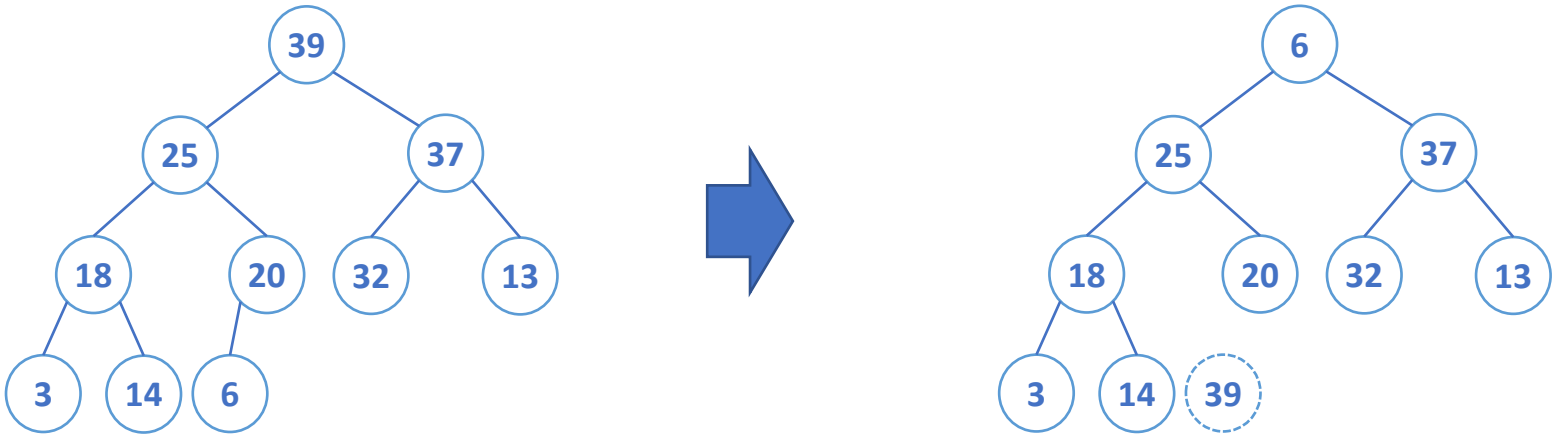
39, 25, 37, 18, 20, 32, 13, 3, 14, 6



Question 5.2

- Consider the following maximum heap: 39, 20, 37, 18, 6, 32, 13, 3, 14. Delete 39 from the heap.
- Step1: When removing a node, we always replace it with the last leaf. Thus we obtain the following list after removing 39.

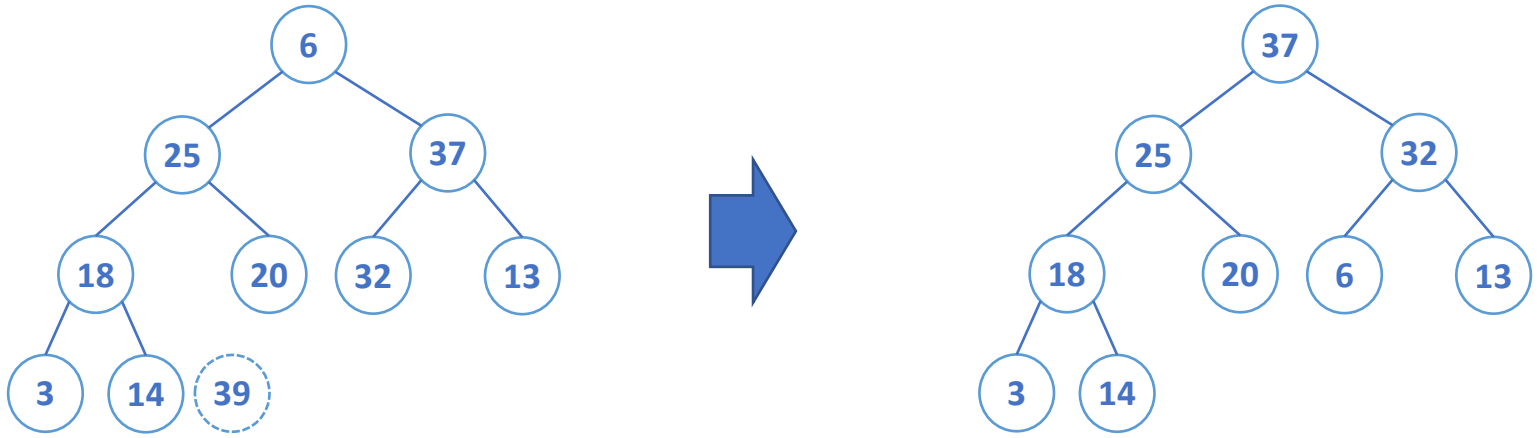
6, 25, 37, 18, 20, 32, 13, 3, 14



Question 5.2 (Contd.)

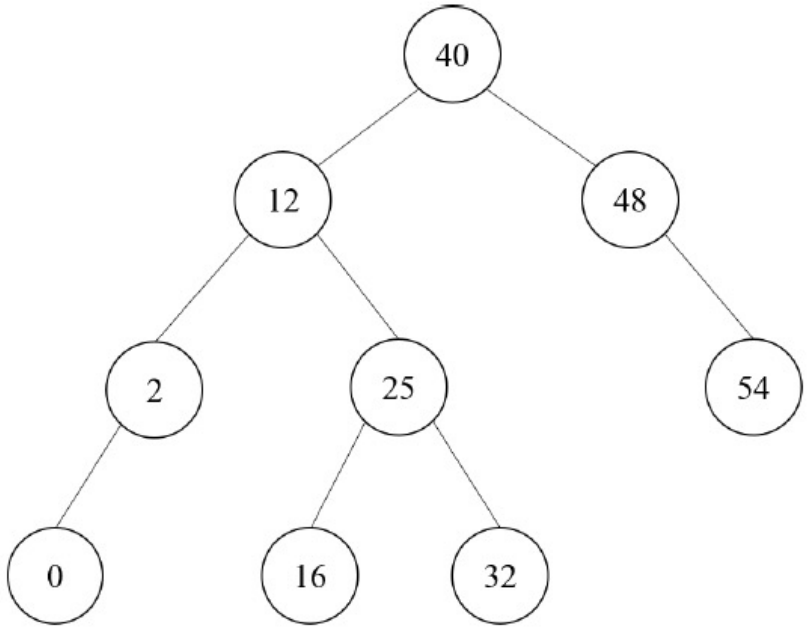
- Consider the following maximum heap: 39, 20, 37, 18, 6, 32, 13, 3, 14. Delete 39 from the heap.
- Step2: We then push the new root down to the correct place it should be by swapping it with its largest child until it is larger than any of its child node. We then obtain

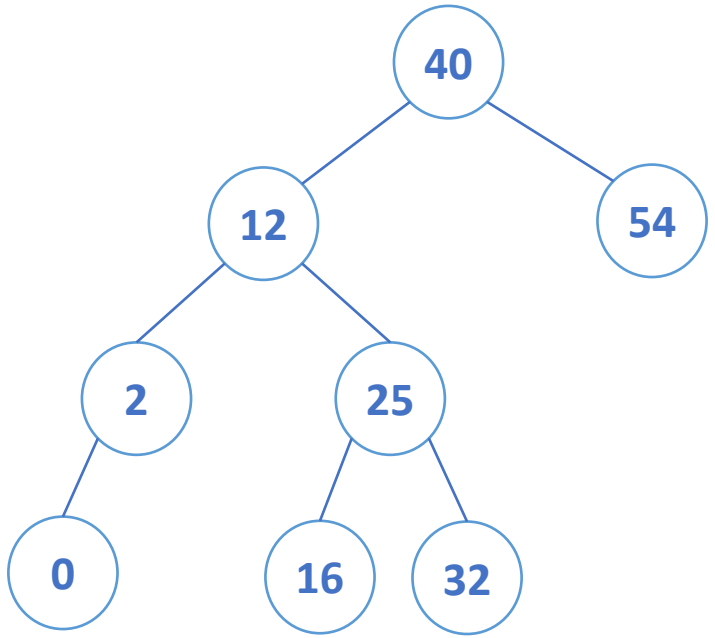
37, 25, 32, 18, 20, 6, 13, 3, 14



Question 6.1

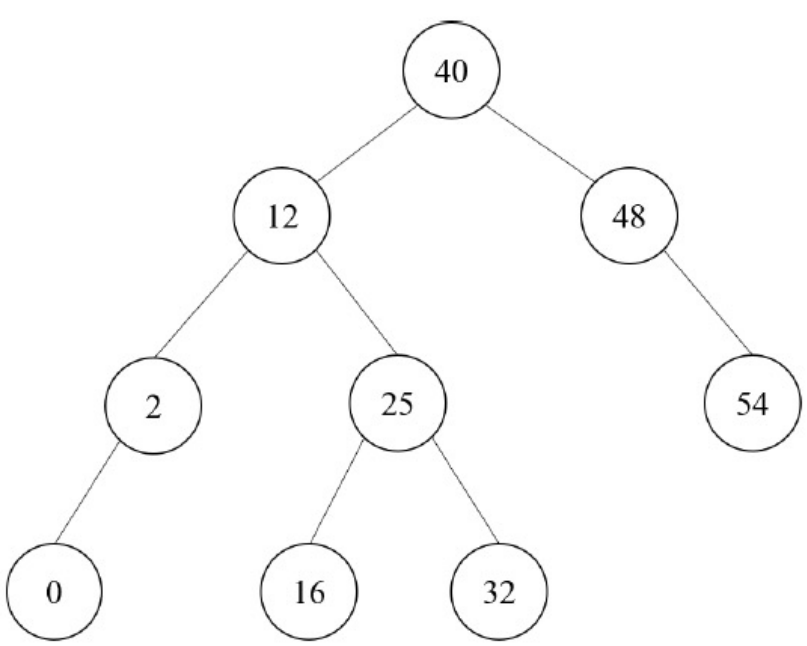
- Delete node 48 in the tree. Node 48 has only one child: delete the node, connect its child to its parent.

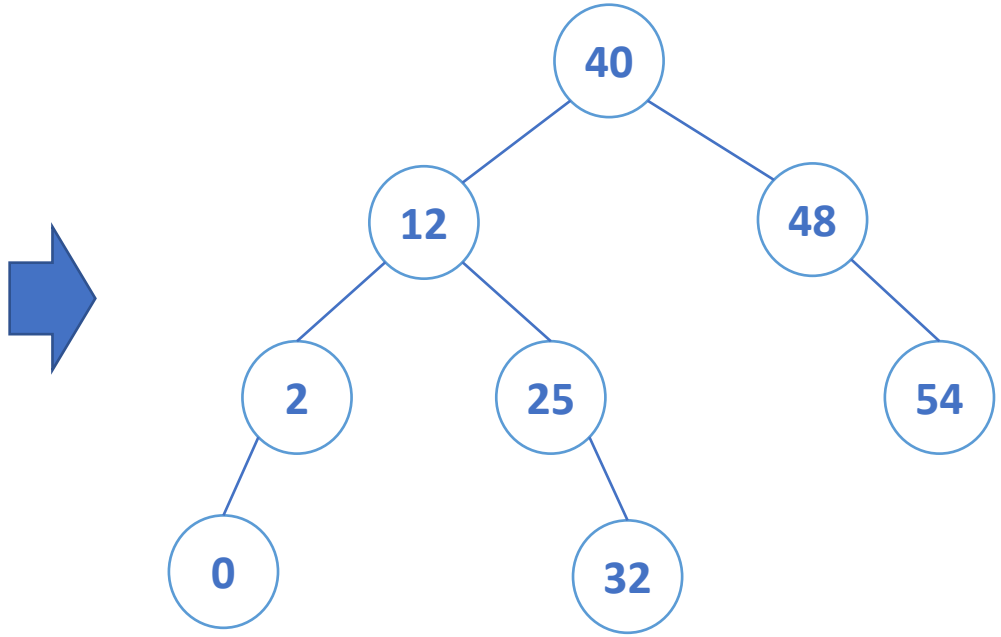




Question 6.2

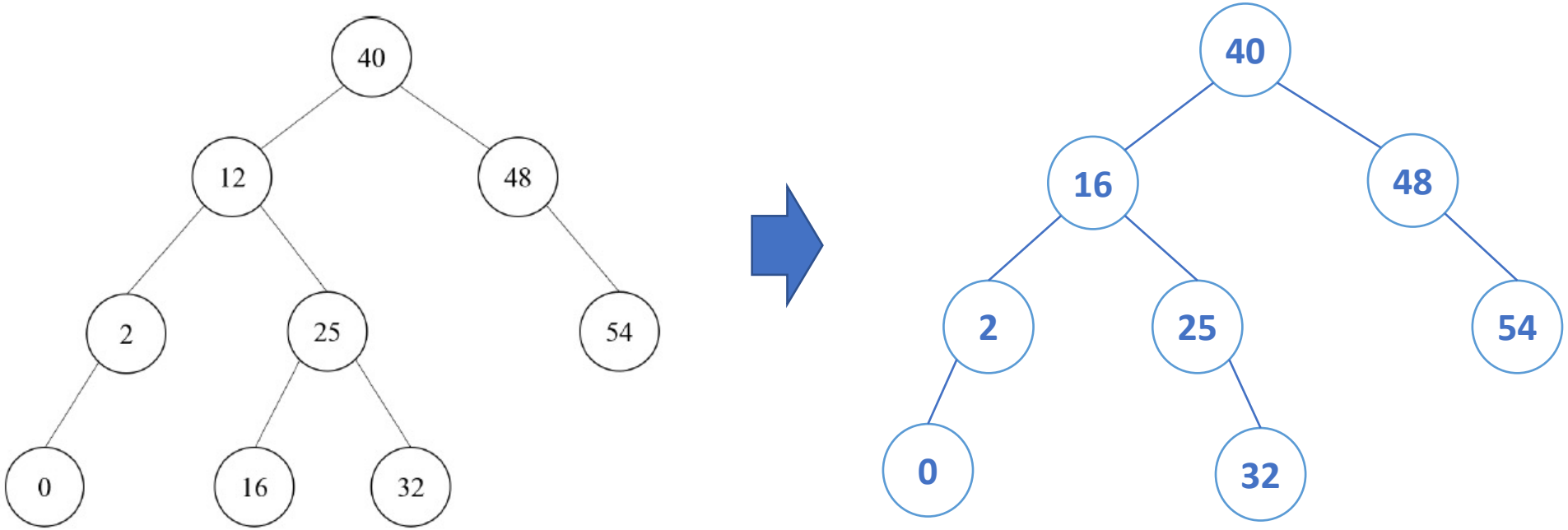
- Delete node 16 in the tree. Node 16 has no children: simply delete.





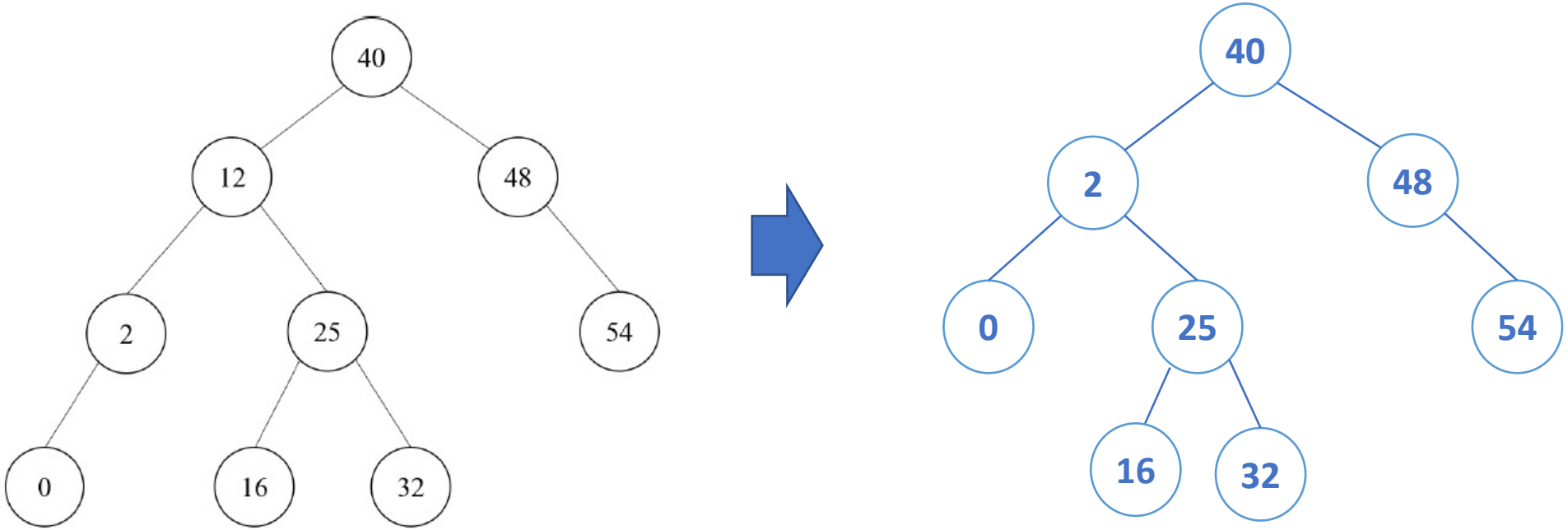
Question 6.3

- Delete node 12 in the tree by using the minimum key in the right subtree. Node 12 has two children: find the minimum key K = 16 in the right subtree, delete that node, and replace the key of node 12 by K.



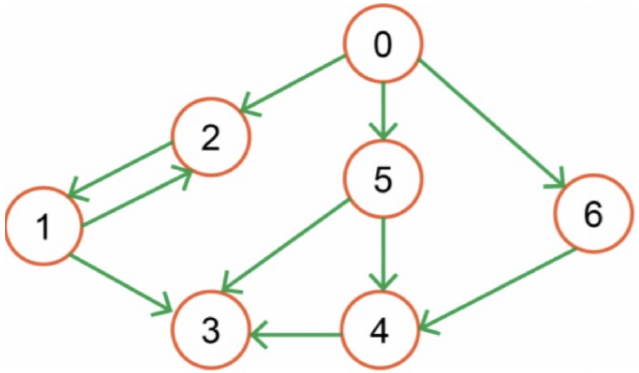
Question 6.4

- Delete node 12 in the tree by using the maximum key in the left subtree. Node 12 has two children: find the maximum key K = 2 in the left subtree, delete that node, and replace the key of node 12 by K.



Question 7.1-7.4

1. What is the order and size of G? Order = 7, Size = 10
2. Identify all sources and sinks in G. Source: 0, Sink: 3
3. Write down the adjacency matrix representation of G. Adjacency matrix as below
4. Consider the vertex sequence 0, 2, 1, 3, 4, 5, is it a walk, path or cycle? None of them



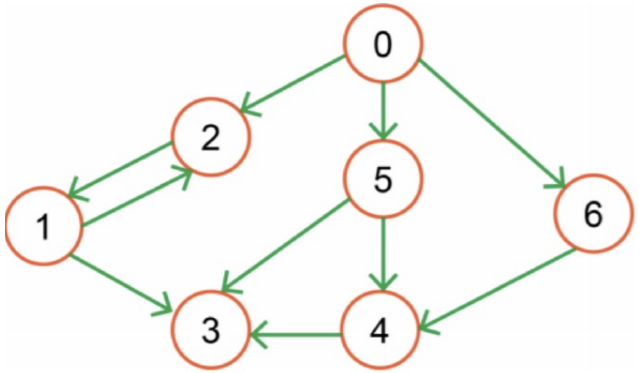
$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Adjacency Matrix

Question 7.5

Give the distance of the following pairs of nodes, d(0, 1), d(0, 3) and d(5, 6).

Definition: The distance, $d(u,v)$, from a node u to a node v in G is the **minimum length of a path from u to v**. If no path exists, the distance is $+\infty$.



$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Adjacency Matrix

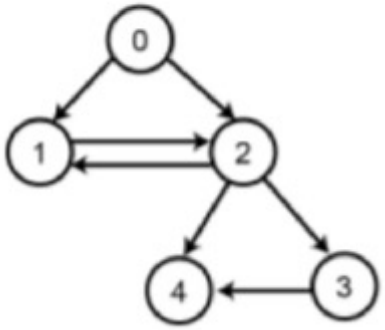
d(0, 1) = 2
d(0, 3) = 2
d(5, 6) = +∞

Question 8

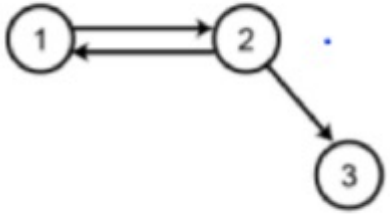
1. Draw the digraph G.
2. Draw the sub-digraph induced by {1, 2, 3}.
3. Draw the underlying graph of G.

0: 1 2
1: 2
2: 1 3 4
3: 4
4:

digraph G



sub-digraph induced by {1,2,3}



underlying graph