# hw2-ans

1. $T(n) = T(n-1) + n^2 = \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} = O(n^3)$

2. By Master Theorem case 2, if $T(n) = aT(n/b) + f(n)$ with $f(n) = \Theta(n^{log_b a})$, then $T(n) = \Theta(n^{log_b a} \lg n)$. Here, $a = 3$, $b = 3$, and $f(n) = \Theta(n^{log_3 3})$. Therefore, $T(n) = \Theta(n \lg n)$.

3. a. We can build the recursion tree as Figure 1.

$$T(n) = \sum_{i=0}^{n-2} 3^i \sqrt{n-i} + T(1)3^{n-1} \leq \sqrt{n-1} \sum_{i=0}^{n-2} 3^i + T(1)3^{n-1} = O(\sqrt{n}3^n)$$

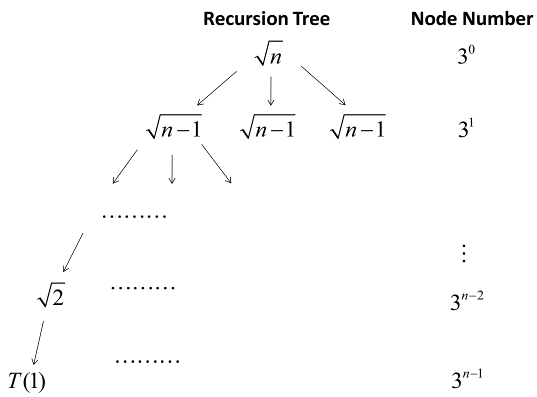



Fig. 1. The Recursion Tree for Question 3.a

b. For certain $c > 0$, we can build the recursion tree as Figure 2, where the cost sum of each layer is $cn$.

$$T(n) = (1 + \log_{\frac{3}{2}} n)O(n) = O(n \lg n)$$

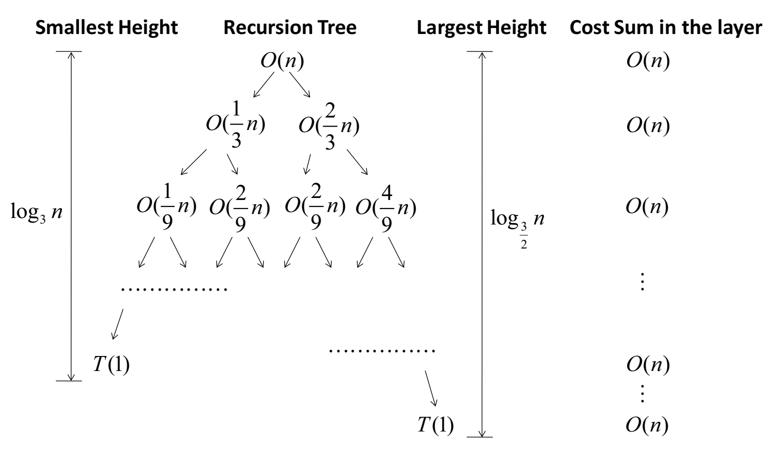



Fig. 2. The Recursion Tree for Question 3.b

c. We can build the recursion tree as Figure 3, where the height of the tree $h = \lg n$.

$$T(n) = n + T_{end} \cdot 2^h + \sum_{j=1}^{h-1} 2^j \cdot (\frac{n}{2^j} + \sum_{i=0}^{j-1} \frac{2}{2^i}) = n + nT_{end} + \sum_{j=1}^{h-1} (n + 2^j \sum_{i=0}^{j-1} \frac{2}{2^i})$$
$$= n \lg n + nT_{end} + 4 \sum_{j=1}^{h-1} (2^j - 1) = n \lg n + nT_{end} + 4n - 4 \lg n - 4$$
$$= O(n \lg n)$$

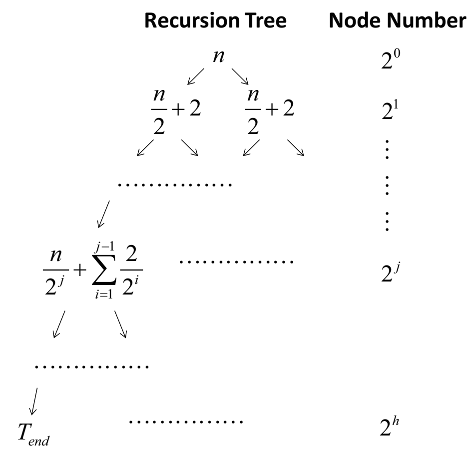



Fig. 3. The Recursion Tree for Question 3.c

4. a. $\because 1 \in O(n^{\log_4 2 - \frac{1}{4}}) = O(\sqrt[4]{n}) \qquad \therefore T(n) \in \Theta(\sqrt{n})$

   b. $\because \sqrt{n} \in \Theta(n^{\log_4 2}) \qquad \therefore T(n) \in \Theta(n^{\log_4 2} \lg n) = \Theta(\sqrt{n} \lg n)$.

   c. $\because n \in \Omega(n^{\log_4 2 + \frac{1}{2}})$ and $2 \cdot \frac{n}{4} = \frac{n}{2} < \frac{3}{4} \cdot n \qquad \therefore T(n) \in \Theta(n)$.

   d. $\because n^2 \in \Omega(n^{\log_4 2 + \frac{3}{2}})$ and $2 \cdot \frac{n^2}{4} = \frac{n^2}{2} < \frac{3}{4} \cdot n^2 \qquad \therefore T(n) \in \Theta(n^2)$.

   e. $\because \sqrt{n} \lg n \in \Theta(n^{\log_4 2} \lg n) \qquad \therefore T(n) \in \Theta(n^{\log_4 2} \lg^2 n) = \Theta(\sqrt{n} \lg^2 n)$.

5. Please refer to the slides of Lecture 6, Page 15.

6. The process of BUILD-MAX-HEAP can be shown in Figure 4.

7. When performing MAX-HEAPIFY in the BUILD-MAX-HEAP algorithm, it is required that the subtrees of the node have already been Max-heaps. Otherwise, the finally result might violate the Max-heap property. Taking the heap in Figure 5(a) for example, if we perform the correct BUILD-MAX-HEAPIFY, we can get the Max-heap as shown in Figure 5(b). However, if we make the change mentioned in Question 7, the obtained heap will be as Figure 5(c). It is obviously not a Max-heap.

8. The process of HEAPSORT should be performed in two steps: Build-Max-Heap firstly and then iteratively MAX-HEAPIFY. The first step is shown in Figure 6, and the second step is shown in Figure 7.

9. It is to ensure that the inserted key is greater than the element $A[A.heap\_size]$. Otherwise, an error might occur when performing MAX-HEAP-INSERT in the following.

10. HEAP-DELETE(A,i)
    {

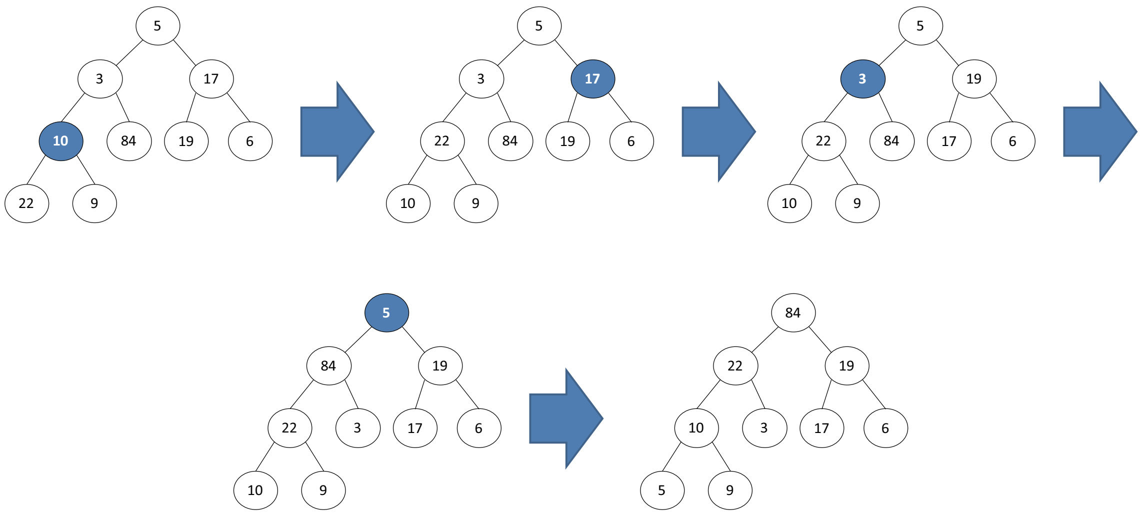
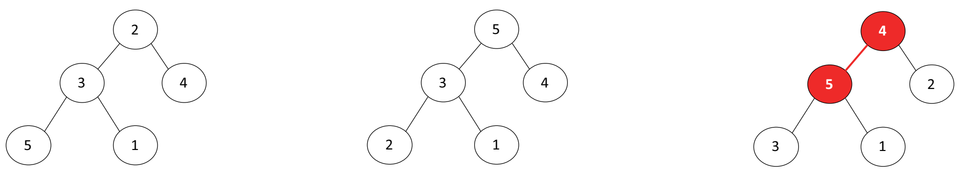



Fig. 4. The BUILD-MAX-HEAP Process for Question 6




(a) Original heap (b) After correct BUILD-MAX-HEAP (c) After incorrect BUILD-MAX-HEAP

Fig. 5. The Examples for Question 7

```
    if (A.heap_size<1 OR i>Aheap_size)   {error;}
    if (A.heap_size==1)   {A=NULL;}
    A[i]=A[A.heap_size];
    A.heap_size=A.heap_size-1;
    MAX-HEAPIFY(A, i);
}
```

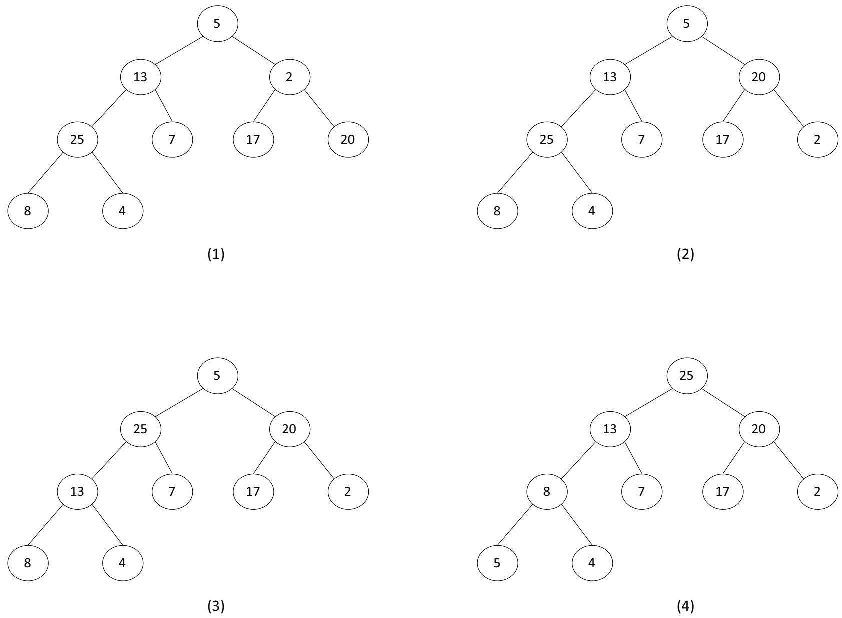



Fig. 6. Step 1 for Question 8: BUILD-MAX-HEAP

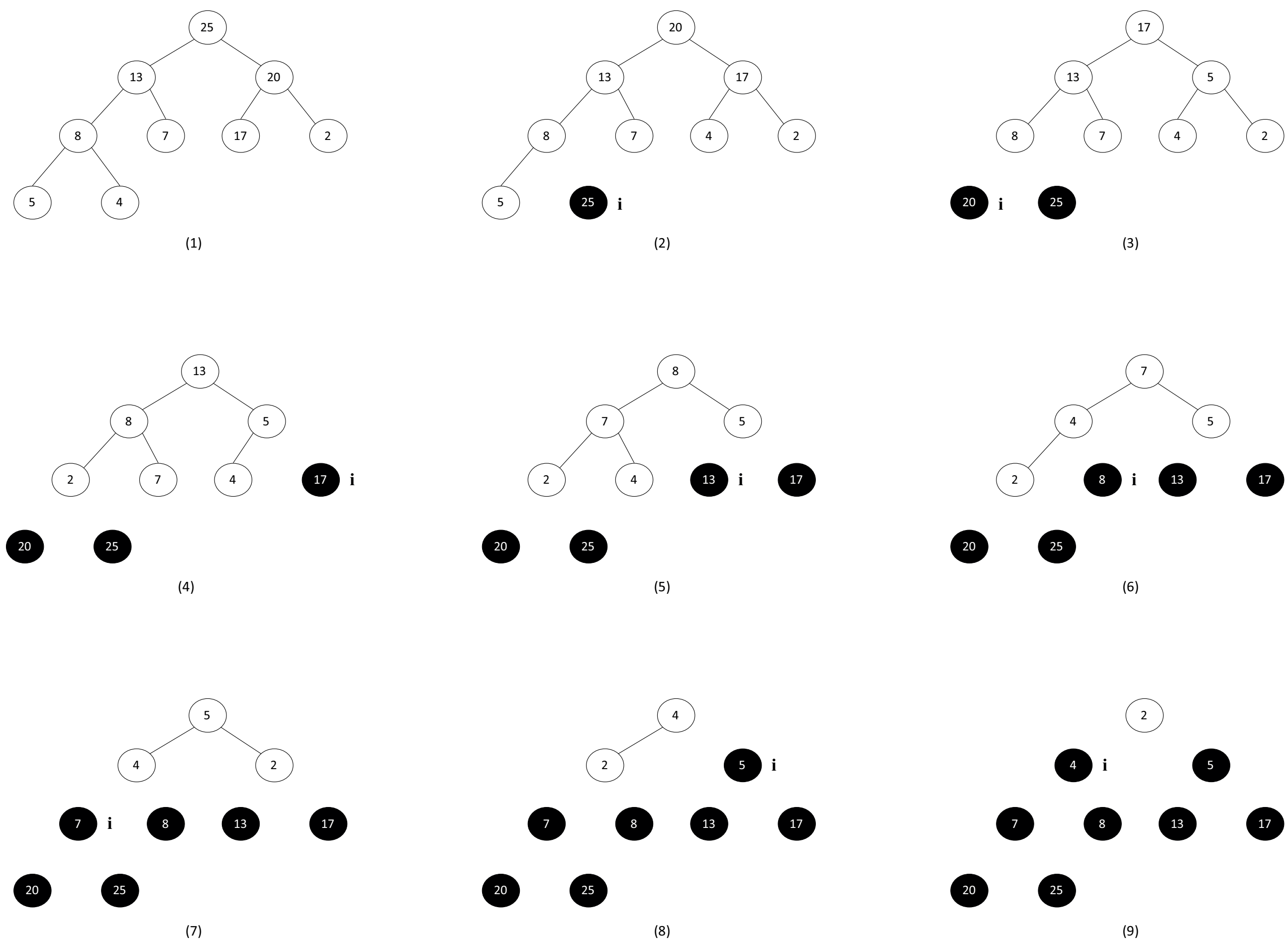



Fig. 7. Step 2 for Question 8: MAX-HEAPIFY iteratively