Chapter 23 Check Point Questions

Section 23.2

▼**23.2.1**
Describe how an insertion sort works. What is the time complexity for an insertion sort?

See the text. The time complexity for an insertion sort is O(n^2).

Hide Answer

▼**23.2.2**
Use Figure 23.1 as an example to show how to apply an insertion sort on {45, 11, 50, 59, 60, 2, 4, 7, 10}.

Omitted.

Hide Answer

▼**23.2.3**
If a list is already sorted, how many comparisons will the insertionSort method perform?

n - 1 times.

Hide Answer

Section 23.3

▼**23.3.1**
Describe how a bubble sort works. What is the time complexity for a bubble sort?

See the text. The time complexity for a bubble sort is O(n^2).

Hide Answer

▼**23.3.2**
Use Figure 23.3 as an example to show how to apply a bubble sort on {45, 11, 50, 59, 60, 2, 4, 7, 10}.

Omitted.

Hide Answer

▼**23.3.3**
If a list is already sorted, how many comparisons will the bubbleSort method perform?

n - 1 times.

Hide Answer

Section 23.4

▼23.4.1

Describe how a merge sort works. What is the time complexity for a merge sort?

See the text. The time complexity for a merge sort is O(nlogn).

Hide Answer

▼23.4.2

Use Figure 23.4 as an example to show how to apply a merge sort on {45, 11, 50, 59, 60, 2, 4, 7, 10}.

Omitted.

Hide Answer

▼23.4.3

What is wrong if lines 6-15 in Listing 23.6, MergeSort.java, are replaced by the following code?

```java
// Merge sort the first half
int[] firstHalf = new int[list.length / 2 + 1];
System.arraycopy(list, 0, firstHalf, 0, list.length / 2 + 1);
mergeSort(firstHalf);

// Merge sort the second half
int secondHalfLength = list.length - list.length / 2 - 1;
int[] secondHalf = new int[secondHalfLength];
System.arraycopy(list, list.length / 2 + 1,
  secondHalf, 0, secondHalfLength);
mergeSort(secondHalf);
```

Consider a list with two elements. firstHalf will be the entire list.

Hide Answer

Section 23.5

▼23.5.1

Describe how quick sort works. What is the time complexity for a quick sort?

See the text. The time complexity for a quick sort is O(n^2) in the worst case and O(nlogn) in the average case.

Hide Answer

▼23.5.2

Why is quick sort more space efficient than merge sort?

Quick sort does not need to create temporary arrays, while merge sort needs temporary arrrys.

Hide Answer

▼23.5.3
Use Figure 23.7 as an example to show how to apply a quick sort on {45, 11, 50, 59, 60, 2, 4, 7, 10}.

Omitted.

Hide Answer

▼23.5.4
If lines 37-38 in the QuickSort program is removed, will it still work? Give a counter example to show that it will not work.

Try to sort this list {2, 22, 2, 5, -3} without the code in ines 37-38.

Hide Answer

Section 23.6

▼23.6.1
What is a complete binary tree? What is a heap? Describe how to remove the root from a heap and how to add a new object to a heap.

A binary tree is complete if every level of the tree is full except that the last level may not be full and all the leaves on the last level are placed left-most. A heap is a binary tree with the following properties:
1. It is a complete binary tree.
2. Each node is greater than or equal to any of its children.

Hide Answer

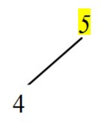
▼23.6.2
When a heap in Figure 23.11a is stored in a list, what is index for element 44? What is the index of parent for 44 and what is the index of left and right children of 44?

The index for 44 is 5. The parent index for 44 is 2. The left child index for 44 is 11 and the right child index for 44 is 12.

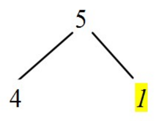
Hide Answer

▼23.6.3
Add the elements 4, 5, 1, 2, 9, and 3 into a heap in this order. Draw the diagrams to show the heap after each element is added.

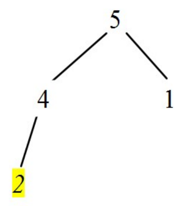
(a) After adding 4



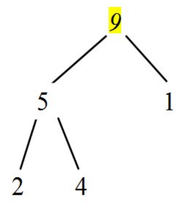
(b) After adding 5



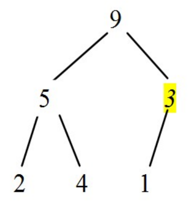
(c) After adding 1



(d) After adding 2



(e) After adding 9



(f) After adding 3

Hide Answer

▼ **23.6.4**

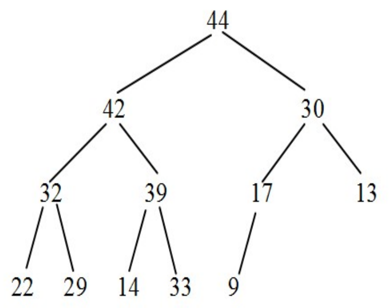
Show the steps of creating a heap using {45, 11, 50, 59, 60, 2, 4, 7, 10}.

Omitted.

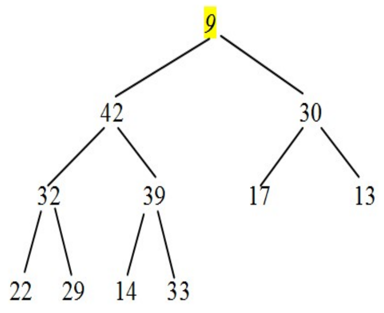
Hide Answer

▼ **23.6.5**

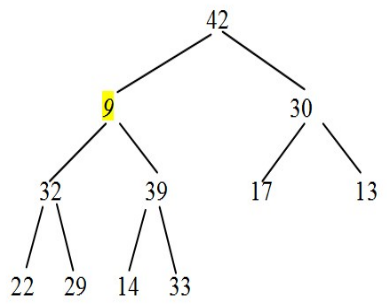
Show the heap after the root in the heap in Figure 23.15c is removed.



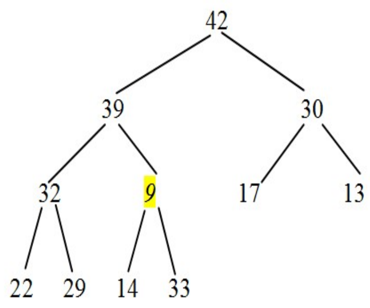
(a) Remove 44



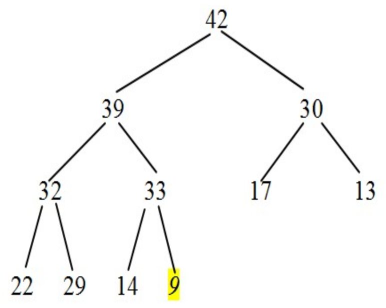
(b) Move 9 to the root



(c) Swap 9 with 42



(d) Swap 9 with 39

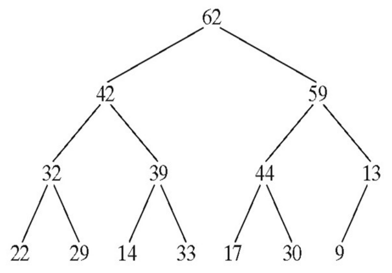


(e) Swap 9 with 33

Hide Answer

▼**23.6.6**

Given the following heap, show the steps of removing all nodes from the heap.



Omitted.

Hide Answer

▼**23.6.7**

Which of the following statements are wrong?

```
1  Heap<Object> heap1 = new Heap<>();
2  Heap<Number> heap2 = new Heap<>();
3  Heap<BigInteger> heap3 = new Heap<>();
4  Heap<Calendar> heap4 = new Heap<>();
5  Heap<> heap5 = new Heap<>();
```

Lines 1 and 2 are wrong, because Object and Number do not implement Comparable.

Hide Answer

▼**23.6.8**

Modify line 12 in Listing 23.10 so that the elements are sorted in a non-increasing order.

Change it to

```
    for (int i = 0; i < list.length; i++)
```

Hide Answer

▼**23.6.9**

What is the return value from invoking the remove method if the heap is empty?

The return value will be null.

Hide Answer

▼**23.6.10**

What is the time complexity of inserting a new element into a heap and what is the time complexity of deleting an element from a heap?

O(logn) for both insertion and deletion.

Hide Answer

▼**23.6.11**

What is the height of an empty heap? What is the height of a heap with only one element? What is the height of a non-empty heap? What is the height of a heap with 16, 17, and 512 elements? If the height of a heap is 5, what is the maximum number of nodes in the heap?

The height of an empty heap is -1 be definition. The height of a heap with only one element is 0. The height of a non-empty heap is the length of a longest path from the root to a leaf. The height of heap with 16 elements is 4. The height of heap with 17 elements is 4. The height of heap with 512 elements is 9. The maximum number of nodes in a heap of height 5 is 63.

Hide Answer

Section 23.7

▼**23.7.1**

Can you sort a list of strings using a bucket sort?

Bucket sort is not suitable for sorting strings.

Hide Answer

▼**23.7.2**

Show how the radix sort works using the numbers 454, 34, 23, 43, 74, 86, and 76.

Omitted.

Hide Answer

Section 23.8

▼**23.8.1**

Describe how external sort works. What is the complexity of the external sort algorithm?

See the text. The time complexity for an external sort is O(nlogn) on disk I/O of blocks of data.

Hide Answer

▼**23.8.2**

Ten numbers {2, 3, 4, 0, 5, 6, 7, 9, 8, 1} are stored in the external file largedata.dat. Trace the SortLargeFile program by hand with MAX_ARRAY_SIZE 2.

Omitted.

Hide Answer