Elementary Data Structures and Algorithms

Binary Heaps

Printable Version

If the heap is stored in an array, it is stored in the standard way.
Arrays use zero-based indexing, unless otherwise indicated.
Assume integer division.

Concept: heap shapes

1. In a heap, the upper bound on the number of leaves is:
   - A. O (1)
   - B. O (n)
   - C. O (log n)
   - D. O (n log n)
2. In a heap, the distance from the root to the furthest leaf is:
   - A. θ (n log n)
   - B. θ (1)
   - C. θ (log n)
   - D. θ (n)
3. In a heap, let d f be the distance of the furthest leaf from the root and let d c be the analogous distance of the closest leaf. What is d f - d c , at most?
   - A. 0
   - B. 2
   - C. θ (log n)
   - D. 1
4. What is the most number of nodes in a heap with a single child?
   - A. 0
   - B. 1
   - C. 2
   - D. Θ (n)
   - E. Θ (log n)
5. What is the fewest number of nodes in a heap with a single child?
   - A. 0
   - B. 2
   - C. one per level
   - D. 1
6. **T** or **F**: There can be two or more nodes in a heap with exactly one child.
7. **T** or **F**: A heap can have no nodes with exactly one child.
8. **T** or **F**: All heaps are perfect trees.
9. **T** or **F**: No heaps are perfect trees.
10. **T** or **F**: All heaps are complete trees.
11. **T** or **F**: No heaps are complete trees.
12. **T** or **F**: A binary tree with one node must be a heap.
13. **T** or **F**: A binary tree with two nodes and with the root having the smallest value must be a min-heap.
14. **T** or **F**: If a node in a heap is a right child and has two children, then its sibling must also have two children.
15. **T** or **F**: If a node in a heap is a right child and has one child, then its sibling must also have one child.

Concept: heap ordering

16. In a min-heap, what is the relationship between a parent and its left child?
    - A. the parent has a smaller value
    - B. there is no relationship between their values
    - C. the parent has a larger value

D. the parent has the same value

17. In a min-heap, what is the relationship between a left child and its sibling?

    A. the right child has a larger value

    B. both children cannot have the same value

    C. the left child has a smaller value

    D. there is no relationship between their values

18. **T** or **F**: A binary tree with three nodes and with the root having the smallest value and two children must be a min heap.

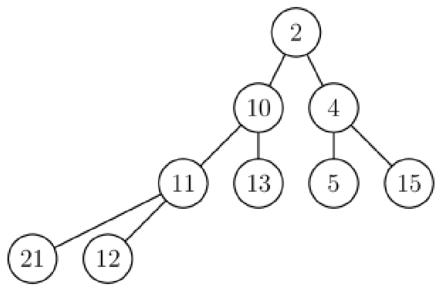
19. **T** or **F**: The largest value in a max-heap can be found at the root.

20. **T** or **F**: The largest value in a min-heap can be found at the root.

21. **T** or **F**: The largest value in a min-heap can be found at a leaf.

Concept: heaps stored in arrays

22. How would this heap be stored in an array?



    A. `[2,4,5,10,11,12,13,15,21]`

    B. `[2,10,4,11,13,5,15,21,12]`

    C. `[21,11,12,10,13,2,5,4,15]`

    D. `[2,10,11,21,12,13,4,5,15]`

23. Printing out the values in the array yield what kind of traversal of the heap?

    A. level-order

    B. post-order

    C. in-order

    D. pre-order

24. Suppose the heap has *n* values. The root of the heap can be found at which index?

    A. *n*-1

    B. *n*

    C. 0

    D. 1

25. Suppose the heap has *n* values. The left child of the root can be found at which index?

    A. *n*-1

    B. 1

    C. 0

    D. *n*

    E. *n*-2

    F. 2

26. Left children in a heap are stored at what kind of indices?

    A. all even but one

    B. a roughly equal mix of odd and even

    C. all odd but one

    D. all odd

    E. all even

27. Right children in a heap are stored at what kind of indices?

    A. all odd but one

    B. all even

    C. a roughly equal mix of odd and even

    D. all odd

    E. all even but one

28. The formula for finding the left child of a node stored at index *i* is:

A. i * 2 - 1

B. i * 2

C. i * 2 + 2

D. i * 2 + 1

29. The formula for finding the right child of a node stored at index *i* is:

A. i * 2

B. i * 2 + 2

C. i * 2 + 1

D. i * 2 - 1

30. The formula for finding the parent of a node stored at index *i* is:

A. i / 2

B. (i + 1) / 2

C. (i - 1) / 2

D. (i + 2) / 2

31. If the array uses one-based indexing, the formula for finding the left child of a node stored at index *i* is:

A. i * 2 - 1

B. i * 2

C. i * 2 + 2

D. i * 2 + 1

32. If the array uses one-based indexing, the formula for finding the right child of a node stored at index *i* is:

A. i * 2 + 1

B. i * 2 - 1

C. i * 2 + 2

D. i * 2

33. If the array uses one-based indexing, the formula for finding the parent of a node stored at index *i* is:

A. (i - 1) / 2

B. i / 2

C. (i + 2) / 2

D. (i + 1) / 2

34. Consider a trinary heap stored in an array. The formula for finding the left child of a node stored at index *i* is:

A. i * 3 - 2

B. i * 3 + 3

C. i * 3 + 1

D. i * 3 - 1

E. i * 3

F. i * 3 + 2

35. Consider a trinary heap stored in an array. The formula for finding the middle child of a node stored at index *i* is:

A. i * 3

B. i * 3 + 3

C. i * 3 + 2

D. i * 3 - 2

E. i * 3 - 1

F. i * 3 + 1

36. Consider a trinary heap stored in an array. The formula for finding the right child of a node stored at index *i* is:

A. i * 3 - 1

B. i * 3 + 2

C. i * 3 + 1

D. i * 3 - 2

E. i * 3 + 3

F. i * 3

37. Consider a trinary heap stored in an array. The formula for finding the parent of a node stored at index *i* is:

A. i / 3 + 1

B. i / 3 - 1

C. (i + 1) / 3

D. (i - 1) / 3

E. (i - 2) / 3

F. (i + 2) / 3

Concept: *heap operations*

38. In a max-heap with no knowledge of the minimum value, the minimum value can be found in time:

    A. θ (n)

    B. θ (n log n)

    C. θ (log n)

    D. θ (1)

39. Suppose a min-heap with *n* values is stored in an array *a*. In the *extractMin* operation, which element immediatelyreplaces the root element (prior to this new root being sifted down).

    A. `a[n-1]`

    B. the minimum of `a[1]` and `a[2]`

    C. `a[2]`

    D. `a[1]`

40. The *findMin* operation in a min-heap takes how much time?

    A. Θ (1)

    B. Θ (n log n)

    C. Θ (log n)

    D. Θ (n)

41. The *extractMin* operation in a min-heap takes how much time?

    A. Θ (n log n)

    B. Θ (log n)

    C. Θ (1)

    D. Θ (n)

42. Merging two heaps of size *n* and *m*, *m* < *n* takes how much time?

    A. Θ (log n + log m)

    B. Θ (n log m)

    C. Θ (n * m)

    D. Θ (log n * log m)

    E. Θ (n + m)

    F. Θ (m log n)

43. The *insert* operation takes how much time?

    A. Θ (n)

    B. Θ (1)

    C. Θ (n log n)

    D. Θ (log n)

44. Turning an unordered array into a heap takes how much time?

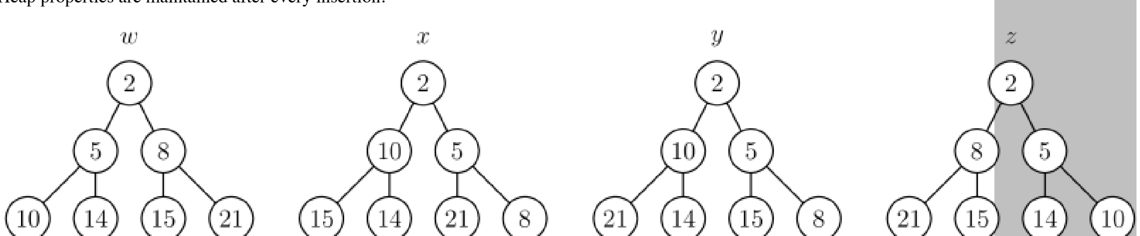
    A. Θ (log n)

    B. Θ (n log n)

    C. Θ (1)

    D. Θ (n)

45. Suppose the values 21, 15, 14, 10, 8, 5, and 2 are inserted, one after the other, into an empty *min*-heap. What does the resulting heap look like? Heap properties are maintained after every insertion.



    A. *y*

    B. *w*

    C. *x*

    D. *z*

46. Using the standard *buildHeap* operation to turn an unordered array into a *max*-heap, how many parent-child swaps are made if the initial unordered array is `[5,21,8,15,25,3,9]`?

    A. 2

B. 4

C. 3

D. 7

E. 5

F. 6