CS1102s Data Structures and Algorithms

12/3/2010

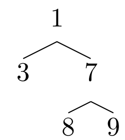
Examination Questions Midterm 2

This examination question booklet has 6 pages, including this cover page, and contains 12 questions.

You have 30 minutes to complete the exam. Use a 2B pencil to fill up the provided MCQ form. Leave Section A blank. Fill up Sections B and C.

Trees

Question 1: Consider the following binary tree:



Which one of the following statements is correct?

1 [A] This tree is a binary search tree.

1 [B] This tree is a complete binary tree.

1 [C] This tree is a binary heap.

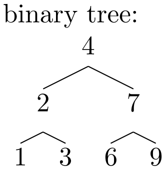
1 [D] This tree is a tree of height 3.

1 [E] None of the above.

Answer 1:

1 [E] None of these is true for the given tree. It is not a complete binary tree, because the last row is not filled from left to right. It is of height 2, because the tree height is the number of edges in the longest path that starts at the root.

Question 2: Consider the following binary tree:



Which one of the following statements is correct?

2 [A] This tree is a binary search tree.

2 [B] This tree is a binary heap.

2 [C] This tree is a tree of height 3.

2 [D] This tree is a tree of height 7.

2 [E] None of the above.

Answer 2:

2 [A] This is a binary search tree.

Question 3: Let N be an arbitrary natural number with $N > 1$. Which one of the following statements is *false*?

3 [A] A binary tree of height N may have $N + 1$ nodes.

3 [B] A binary tree of height N may have N^2 nodes.

3 [C] A binary tree of height N may have 2^N nodes.

3 [D] A binary tree of height N may have $N!$ nodes.

Answer 3:

3 [D] A binary tree of height 5 cannot have $5! = 120$ nodes. The highest number of nodes is $2^{5+1} - 1 = 63$.

Question 4: We consider log to the base of 2. Thus for $\log N$ to be a natural number, N must be a power of 2.
Let N be an arbitrary number with $N > 1$. Which one of the following statements is *true*?

4 [A] A binary tree of height $\log N$ can have exactly N nodes.

4 [B] A binary tree of height $\log N$ can have exactly $\log N$ nodes.

4 [C] A binary tree of height $\log N$ can have exactly $N \log N$ nodes.

4 [D] A binary tree of height $\log N$ can have exactly N^2 nodes.

4 [E] None of the above.

Answer 4:

4 [A] A binary tree of height $\log N$ can have between $1 + \log N$ and $2^{1+\log N} - 1 = 2N - 1$ nodes. For any $N > 1$, N is between $1 + \log N$ and $2^{1+\log N} - 1 = 2N - 1$.

Sorting

Question 5: Assume a set A of arrays of size N all of which are sorted already. Which one of the following statements it correct?

5 [A] Quicksort on elements of A has a runtime in $\Omega(N^2)$.

5 [B] Mergesort on elements of A has a runtime in $\Omega(N^2)$.

5 [C] Insertion Sort on elements of A has a runtime in $\Omega(N^2)$.

5 [D] Heapsort on elements of A has a runtime in $\Omega(N^2)$.

5 [E] None of the above.

Answer 5:

5 [A] Quicksort with naive pivot selection (choose the first element) has a runtime in $\Theta(N^2)$ on sorted arrays of size N.

Answer 5:

5 [E] Quicksort with median of two pivot selection has a runtime in $\Theta(N \log N)$ on sorted arrays of size N.

Question 6:
Which statement on Mergesort is correct?

6 [A] There are arrays of varying size N such that Mergesort runs in $\Theta(N^2)$.

6 [B] There are arrays of varying size N such that Mergesort runs in $o(N^2)$.

6 [C] There are arrays of varying size N such that Mergesort runs in $O(N)$.

6 [D] There are arrays of varying size N such that Mergesort runs in $\Omega(N^2)$.

6 [E] None of the above.

Answer 6:

6 [B] Mergesort on any arrays runs in $\Theta(N \log N)$, and thus in $o(N^2)$. (In case of doubt, review the definitions of o, O, Θ, Ω.

Question 7: Which statement on Insertion sort is correct?

7 [A] There are arrays of varying size N such that Insertion Sort runs in $\Theta(N)$.

7 [B] There are arrays of varying size N such that Insertion Sort runs in $\Theta(1)$.

7 [C] There are arrays of varying size N such that Insertion Sort runs in $\Theta(N^3)$.

7 [D] There are arrays of varying size N such that Insertion Sort runs in $\Theta(2^N)$.

7 [E] None of the above.

Answer 7:

7 [A] If you restrict the arrays on which you apply Insertion Sort to arrays that are already sorted, you get a run time in $\Theta(N)$, because the inner loop runs in $O(1)$.

Question 8: Some sorting algorithms require extra space, apart from the space needed for the original array that needs to be sorted. Which one of the following statements on the space usage of sorting algorithms is correct?

8 [A] Heapsort for sorting an array of size N requires an amount of extra space proportional to N.

8 [B] Insertion Sort for sorting an array of size N requires an amount of extra space proportional to N.

8 [C] Mergesort for sorting an array of size N requires an amount of extra space proportional to N.

8 [D] Quicksort for sorting an array of size N requires an amount of extra space proportional to N.

8 [E] None of the above.

Answer 8:

8 [C] Mergesort is the only algorithm of these that requires a helper array of the same size as the given array.

Question 9: The Shellsort algorithm sorts sub-arrays of elements of decreasing distance from each other. These distances are called *increments*. Which statemement on these increments is correct?

9 [A] For every array size N, there is a sequence of increments such that Shellsort runs in $\Theta(N \log N)$.

9 [B] For every array size N, there is a sequence of increments such that Shellsort runs in $\Theta(N)$.

9 [C] For every array size N, there is a sequence of increments such that Shellsort runs in $\Theta(\sqrt{N})$.

9 [D] For every array size N, there is a sequence of increments such that Shellsort runs in $\Theta(N^2)$.

9 [E] None of the above.

Answer 9:

9 [D] Shell's original increments lead to a runtime in $\Theta(N^2)$.

Heaps

Question 10: Which statement on the `buildHeap` operation is correct?

10 [A] `buildHeap` calls `percolateUp` only on all internal nodes.

10 [B] `buildHeap` calls `percolateUp` only on all leaf nodes.

10 [C] `buildHeap` calls `percolateUp` only on all nodes with maximal depth.

10 [D] `buildHeap` calls `percolateUp` on all nodes.

10 [E] None of the above.

Answer 10:

10 [E] `buildHeap` uses `percolateDown`, not `percolateUp`.

Question 11: Heaps are usually implemented using arrays. Which of the following statements is correct?
The removal of an element with known array index from an array of size N requires a time in

11 [A] $\Theta(\log N)$

11 [B] $\Theta(N)$

11 [C] $\Theta(1)$

11 [D] $\Theta(N^2)$

11 [E] None of the above.

Answer 11:

11 [A] A straightforward way to implement the remove operation is to replace the given element with the element with the largest index, and then call `percolateDown`. Regardless of the implementation technique, there is no way of avoiding either going up or down from the position, until the root or a leaf is reached, and thus a correct and efficient implementation will run in $\Theta(\log N)$.

Hash Tables

Question 12: Consider a hash table using separate chaining for storing integer values in the range from 0 to 9999. Consider a table size (size of the array) of 1000. Which statement about the linked lists of the hash table is correct.

12 [A] One of the linked lists can reach a size of 10, but not more.

12 [B] One of the linked lists can reach a size of 100, but not more.

12 [C] One of the linked lists can reach a size of 1000, but not more.

12 [D] One of the linked lists can reach a size of 10000.

12 [E] None of the above.

Answer 12:

12 [A] A good hash function for this application will be $hash(N) = N \mod 1000$, which makes sure that any linked list will have a size up to 10.

Answer 12:

12 [B]

Answer 12:

12 [C]

Answer 12:

12 [D] Since the hash function was not specified, you could use different hash functions, which could be as bad as $hash(N) = 0$, which could lead to linked lists of length 10,000.