Practise midterm 2021

Due No due date **Points** 25 **Questions** 25 **Time limit** None
Allowed attempts Unlimited

Instructions

These questions are from past years and will give you a good feel for what you can expect on the actual midterm.

Take the quiz again

Attempt history

	Attempt	Time	Score
LATEST	Attempt 1	125 minutes	10.33 out of 25

Submitted 1 Oct at 14:06

Question 1 — 0 / 1 pts

What is the minimal number of leaves that a decision tree for some sorting algorithm for an input of size n has?

ou Answered
- (•) 2^(n-1)
- ( ) 2^n

rrect answer
- ( ) n!
- ( ) n log(n)

Question 2

0 / 1 pts

Consider the following recurrence equation: T(0) = 1 and T(n)= T(n-1) + n for n > 1. What is the big-O of T?

rrect answer

- T(n) in O(n^2)
- T(n) in O(n log(n))
- T(n) in O(2^n)

u Answered

- T(n) in O(n)

nanswered

Question 3

0 / 1 pts

We consider an application of buildMaxHeap (refer to the book for the algorithm) to A=[1,2,3,4,5,6,7]. What is A after the first iteration of the for-loop?

- [3,2,7,4,5,6,1]
- [1,5,3,4,2,6,7]

rrect answer

- [1,2,7,4,5,6,3]
- [3,2,1,4,5,6,7]

Question 4

0 / 1 pts

We consider the algorithm partition used in quicksort (refer to the book for the algorithm). What is the worst-case time complexity?

ou Answered

- (●) The worst-case time complexity is in Theta (log (n)) with n the number of elements in the array A.

- ( ) The worst-case time complexity is in Theta (n^2) with n the number of elements in the array A.

rrect answer

- ( ) The worst-case time complexity is in Theta (n) with n the number of elements in the array A.

- ( ) The worst-case time complexity is in Theta (n log(n)) with n the number of elements in the array A.

Question 5

0.67 / 1 pts

Which of the following is(are) correct?

Correct!

- [x] n^2 in Theta (n^2)
- [ ] n^3 in O(n^2)

- [ ] n^3 in Theta (n^2)

Correct!

- [x] n^2 in O(n^2)
- [ ] n^2 in Theta (n^3)

rrect answer

- [ ] n^2 in O(n^3)

Question 6

0.67 / 1 pts

Which of the following recurrence equations (one or more) describe(s) correctly a function T for the worst-case time complexity of merge sort (assuming that T(1) = 1)?

- [ ] T(n/2) + n
- [ ] 4T(n/2) + 2n

Correct!

- [x] 2T(n‘2) + n

rrect answer

- [ ] 2T(n/2) + 4n

Correct!

- [x] 2T(n/2) + cn with c a constant

Question 7

0 / 1 pts

Why does the linear time complexity of counting sort not contradict the lower bound on the worst-case complexity of comparison-based sorting?

rrect answer

u Answered

- Because counting sort is not comparison-based.
- Because counting sort uses additional memory. (selected)
- Because the lower bound is linear time.
- Because the lower bound is on worst-case and the linear time complexity of counting sort is best-case.

Question 8

1 / 1 pts

An algorithm with worst-case time complexity Theta (n log(n)) is always, for any input, faster than an algorithm with worst-case time complexity in Theta (n^2). Is this statement true or false, and why?

- True, because Theta gives a strict bound
- False, because n log(n) and n^2 have the same rate of growth anyway.
- True, because n log(n) has a lower rate of growth than n^2

Correct!

- False, because for a small input or for a special input an algorithm with worst-case time complexity in Theta (n^2) may perform better than an algorithms with worst-case time complexity in Theta (n log(n)) (selected)

Question 9

0 / 1 pts

We consider insertionsort. What happens if we swap the order of the tests for the while-loop in line 5?

- The algorithm then sorts in reverse order.
- rrect answer
- We may then possibly compare A[0] with key, but A[0] does not exist.
- The program then does not terminate.
- u Answered
- It does not matter.

Question 10

0 / 1 pts

Which of the following sorting algorithms has/have a worst-case time complexity in Theta (n log(n))?

- insertion sort
- quicksort
- rrect answer
- merge sort
- selection sort
- bucket sort
- u Answered
- counting sort
- rrect answer
- heapsort

Question 11

1 / 1 pts

The heapsort algorithm consists of two parts: first build a max-heap and then iterate removing elements from it. Which of the two parts is responsible for the worst case complexity of heapsort?

Correct!

- (•) The second part, because the first part is linear.
- ( ) Both parts have the worst time complexity of heapsort.
- ( ) The first part, because the second part is linear.
- ( ) It depends on the input.

Question 12

0 / 1 pts

We consider counting sort (refer to the book for the algorithm). What happens if we change the last for-loop by letting the index j go from 1 up to A.length?

ou Answered

- (•) The algorithm is no longer correct.

rrect answer

- ( ) The algorithm still sorts correctly, but is no longer stable.
- ( ) The algorithm sorts in reverse order.
- ( ) The change has no effect at all.

Question 13

0 / 1 pts

Merge-sort is a divide-and-conquer algorithm. Suppose we adapt merge sort by splitting the sequence into 4 more or less equal-size parts. Do we get an essentially better worst-case time complexity?

ou Answered

- Yes, because the merge-procedure takes smaller inputs.
- No, because then the merge-procedure takes more time,
- Yes, we get an algorithm which is twice as fast because the recursion tree has smaller height.

rrect answer

- No, because the height of the recursion tree is then still proportional to log(n).

Question 14

0 / 1 pts

Which recurrence equation correctly describes, next to T(0)=1, the worst-case time complexity of quicksort?

- T(n) = 2T(n/2)

ou Answered

- T(n) = T(n) + T(0) + cn
- T(n) = 2T(n/2) + cn

rrect answer

- T(n) = T(n-1) + T(0) + cn

Question 15

1 / 1 pts

What is the best-case time complexity of insertionsort ?

Correct!

- (•) Theta (n)
- ( ) Theta (n log(n))
- ( ) Theta (n^2)
- ( ) Theta (log(n))

Question 16

1 / 1 pts

We consider two statements.
Statement A is: $2^{(n+1)}$ is in $O(2^n)$.
Statement B is: $2^{(2n)}$ is in $O(2^n)$.
Which of the following holds?

- ( ) B is true but A is false
- ( ) A and B are both true
- ( ) A and B are both false

Correct!

- (•) A is true but B is false

Question 17

0 / 1 pts

What is the result of adding the key 10 to the max-heap [7,6,5,4,3,2,1,0,0,0] with initially heapsize 7, provided we apply the adding-operation `on the fly'?

- [10,7,6,5,4,3,2,1,0,0] with heapsize 8.
- ou Answered [10,7,6,5,2,3,4,1,0,0] with heapsize 10.
- rrect answer [10,7,5,6,3,2,1,4,0,0] with heapsize 8.
- [7,5,6,1,2,3,4,10,0,0] with heapsize 8.

Question 18

0 / 1 pts

We consider the algorithm for the bottom-up max-heap construction. What happens if we let the loop-index increase from 1 upwards ito floor of A.length/2 ?

- ou Answered It works equally well.
- rrect answer Then the algorithm is no longer correct.
- Then we need to take the ceiling of A.length / 2 instead of the floor in order for the algorithm to be correct.

Then we should do the recursive call of MaxHeapify on A and i-1 instead of on A and i in order for the algorithm to be correct.

Question 19

1 / 1 pts

We consider a max-heap containing n different keys. Why do we have that the height of such a max-heap is approximately log(n)?

Correct!

- (•) Because a max-heap is an almost-complete binary tree.
- ( ) Because the keys in a max-heap are partially ordered.
- ( ) Because a max-heap is a binary tree.
- ( ) Because all elements are different.

Question 20

0 / 1 pts

What is the time complexity of quicksort on an array in which all keys are identical?

ou Answered

- (•) Theta(n log(n)), because the array is already sorted, but you still have the recursive calls.
- ( ) Theta(1), because the array is trivial.
- ( ) Theta(n), because the array is already sorted.

rrect answer

Theta(n^2), because when partitioning one side will always be empty.

Question 21

1 / 1 pts

When is a sorting algorithm said to be stable?

If sorting the array twice in a row does not affect the result.

If it does not need extra memory next to the input array.

Correct!

If it respects the order of equal keys in the array.

If the algorithm is linear time on an array that is already sorted.

Question 22

1 / 1 pts

Why is quicksort a good sorting algorithm?

Correct!

Because it is rather efficient, despite the fact that it does not have an optimal worst case time complexity.

Because it has good worst-case time complexity.

Because it is the default sorting algorithm in the libraries of major programming languages.

Because it is a recursive algorithm.

Question 23

1 / 1 pts

What is the running time of heapsort if the input is an array that is already sorted (in increasing order)?

Correct!

- The running time is then in O (nlog(n)).
- The running time is then in O(n).
- The running time is then in O (log(n)).
- The running time is then in O(n2).

Question 24

0 / 1 pts

We consider the algorithm for partition (refer to the book for the algorithm). Apply partition to the input array A = [3,7,6,1,8,5,2,4] and indices p=1 and r=8. What is A after having executed the iterations for j=1,2,3,4?

ou Answered

- A = [1,3,6,7,8,5,2,4]
- A = [3,1,7,6,8,5,2,4]
- A = [3,7,6,4,8,5,2,1]

rrect answer

- A = [3,1,6,7,8,5,2,4]

Question 25

1 / 1 pts

What is the effect of applying the extract-max procedure (refer to the book for the algorithm) on the max-heap [8,7,4,6,1,2,3,5]?

Correct!

- [7,6,4,5,1,2,3,5] with heapsize 7
- [7,6,5,4,3,2,1,5] with heapsize 7
- [7,5,4,6,1,2,3,5] with heapsize 7
- [7,6,4,5,1,2,3,5] with heapsize 8