

School of Computing and Information Systems
COMP90038 Algorithms and Complexity Tutorial Week 8

16–20 September 2019

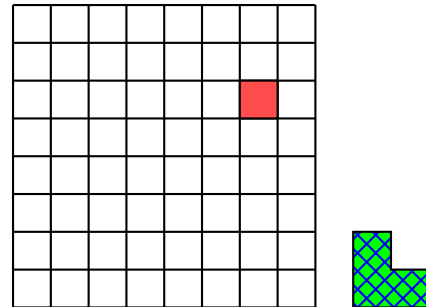
Plan

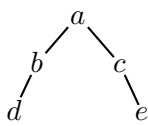
This week we will cover the rest of Levitin's Chapter 6 (except we skip 6.2 and 6.5). There are many exercises this week, but most are quick drill-and-practice questions.

The exercises

50. A *tromino* is an L-shaped tile made up of three 1×1 squares (green/hatched in the diagram below). You are given a $2^n \times 2^n$ chessboard with one missing square (red/grey in the diagram below). The task is to cover the remaining squares with trominos, without any overlap. Design a divide-and-conquer method for this. Express the cost of solving the problem as a recurrence relation and use the Master Theorem to find the order of growth of the cost.

Hint: This is a nice example where it is useful to split the original problem into *four* instances to solve recursively.



51. Traverse  in preorder, inorder, and postorder.
52. A certain binary tree yields a, b, c, d, e when traversed preorder, and it yields the same sequence when traversed inorder. What does the binary tree look like?
53. A certain binary tree yields 14, 83, 63, 42, 19, 27, 74, 99, 51, 37 when traversed preorder and 63, 83, 19, 42, 14, 27, 99, 51, 74, 37 when traversed inorder. Which sequence does it yield when traversed postorder?

54. The following algorithm was designed to compute the number of leaves in a binary tree T . We denote the empty tree by *null*.

```
function LEAFCOUNT( $T$ )  
  if  $T = \text{null}$  then  
    return 0  
  else  
    return LEAFCOUNT( $T.\text{left}$ ) + LEAFCOUNT( $T.\text{right}$ )
```

Fix the error in this algorithm.

55. (Optional, for those who are keen to practice induction proofs.) In lecture 13 (on slide 16) we analyzed the worst-case time complexity of bottom-up heap creation. The analysis made use of this fact:

$$\sum_{i=1}^h i \cdot 2^{h-i} = 2^{h+1} - h - 2$$

Use mathematical induction to prove it.

56. Make a max-heap out of the keys A, L, G, O, R, I, T, H, M, using the bottom-up algorithm.
57. Construct a max-heap from the empty heap by inserting the keys A, L, G, O, R, I, T, H, M, one by one, in that order. Is the result the same as the heap from the previous question?
58. Give an algorithm for deciding whether an array $A[1]..A[n]$ is a heap.
59. Apply the heapsort algorithm to A, L, G, O, R, I, T, H, M.