**CPSC 6109:** [**Advanced**](https://colstate.view.usg.edu/d2l/lp/ouHome/home.d2l?ou=1218642) **Algorithms**

**Spring 2018**

**Assignment #1**

**Due: 11:59 PM Tuesday, February 6**

**Student name: Lu Lin**

**Date: 03/02/2018**

Do the following exercises/problems. Each problem is worth 20 points with a total of 100 points.

1. Give asymptotic upper and lower bounds for T(n) in the following recurrences. Assume that T(n) is constant for n ≤ 2. Make your bounds as tight as possible, and justify your answers.

T(n) = 4 T(n/4) + n3

Solution 1: I guess that the solution is T(n) = Θ(n3). My strategy is to prove T(n) ≤ c n3 for an appropriate choice of constant c > 0. Substituting into the recurrence I got:

)3 + n3  
= + n3

=

≤

Where the last step holds for c ≥ n > 0.

Solution 2: According to Master Theorem a = 4; b = 4; f(n) =

Let me check case 3 conditions: f(n) = Ω() for some constant and f(n) satisfies the regularity condition that af(n/b) ≤ cf(n) for some constant c < 1 and all sufficiently large n, then T(n) = Θ(f(n)) = Θ(n3).

1. Exercise 4.5-1 (c) on page 96. Use the master method to give tight asymptotic bounds for the following recurrences: T(n) = 2T(n/4) + n

Solution: Here I use master theorem and check the regularity condition as following: a = 2; b = 4; f(n) = n; f(n) = Ω() for some constant and 2f(n/4) / f(n) = 1/8 ≤ c for some constant c < 1 and all sufficiently large n satisfies the regularity condition. Thus T(n) = Θ(f(n)) = Θ(n)

1. Exercise 4.5-1 (d) on page 96. T(n) = 2T(n/4) + n2

Solution: Similar to problem(2), I check the regularity condition as: a = 2; b = 4; f(n) = n2; f(n) = Ω() for some constant and 2f(n/4) / f(n) = 1/16 ≤ c for some constant c < 1. Thus T(n) = Θ(f(n)) = Θ(n2)

1. Exercise 6.4-1 on page 160 on the array

A = <20, 13, 2, 15, 1, 18, 7, 14, 10, 16, 3>.

Solution:

First step is to call BUILD-MAX-HEAP(A) as following:

i = ˻11/2˼ = 5, A = <20, 13, 2, 15, 16, 18, 7, 14, 10, 1, 3>

i = 4, A = <20, 13, 2, 15, 16, 18, 7, 14, 10, 1, 3>

i = 3, A = <20, 13, 18, 15, 16, 2, 7, 14, 10, 1, 3>

i = 2, A = <20, 16, 18, 15, 13, 2, 7, 14, 10, 1, 3>

i = 1, A = <20, 16, 18, 15, 13, 2, 7, 14, 10, 1, 3>

Second step let A = <3, 16, 18, 15, 13, 2, 7, 14, 10, 1> and A’ = <20>. Call MAX-HEAPIFY(A, 1) we got A = <18, 16, 7, 15, 13, 2, 3, 14, 10, 1>

Third step let A = <1, 16, 7, 15, 13, 2, 3, 14, 10> and A’ = <18, 20>. Call MAX-HEAPIFY(A, 1) we got A = <16, 15, 7, 14, 13, 2, 3, 1, 10>

Fourth step let A = <10, 15, 7, 14, 13, 2, 3, 1> and A’ = <16, 18, 20>. Call MAX-HEAPIFY(A, 1) we got A = <15, 14, 7, 10, 13, 2, 3, 1>

Fifth step let A = <1, 14, 7, 10, 13, 2, 3> and A’ = <15, 16, 18, 20>. Call MAX-HEAPIFY(A, 1) we got A = <14, 13, 7, 10, 1, 2, 3>

Sixth step let A = <3, 13, 7, 10, 1, 2> and A’ = <14, 15, 16, 18, 20>. Call MAX-HEAPIFY(A, 1) we got A = <13, 10, 7, 3, 1, 2>

Seventh step let A = <2, 10, 7, 3, 1> and A’ = <13, 14, 15, 16, 18, 20>. Call MAX-HEAPIFY(A, 1) we got A = <10, 3, 7, 2, 1>

Eighth step let A = <1, 3, 7, 2> and A’ = <10, 13, 14, 15, 16, 18, 20>. Call MAX-HEAPIFY(A, 1) we got A = <7, 3, 1, 2>

Ninth step let A = <2, 3, 1> and A’ = <7, 10, 13, 14, 15, 16, 18, 20>. Call MAX-HEAPIFY(A, 1) we got A = <3, 2, 1>

Tenth step let A = <1, 2> and A’ = <3, 7, 10, 13, 14, 15, 16, 18, 20>. Call MAX-HEAPIFY(A, 1) we got A = <2, 1>

Last step let A = <1> and A’ = <2, 3, 7, 10, 13, 14, 15, 16, 18, 20>. Call MAX-HEAPIFY(A, 1) we got A = <1> and finally we got an output A’ = <1, 2, 3, 7, 10, 13, 14, 15, 16, 18, 20>

1. Exercise 6.5-1 on page 164 with the heap

A = <20, 15, 10, 5, 12, 8, 7, 4, 2, 6, 5, 1>.

Solution: the following array shows the maximum element is extracted from the heap A and A as a heap with this element removed.

Input original A = <20, 15, 10, 5, 12, 8, 7, 4, 2, 6, 5, 1>

Move the last element to the top of the heap, A = <1, 15, 10, 5, 12, 8, 7, 4, 2, 6, 5>

MAX-HEAPIFY(A, 1), A = <15, 1, 10, 5, 12, 8, 7, 4, 2, 6, 5> and then A = <15, 12, 10, 5, 1, 8, 7, 4, 2, 6, 5> and then A = <15, 12, 10, 5, 6, 8, 7, 4, 2, 1, 5>

Output the max element 20 and heap A = <15, 12, 10, 5, 6, 8, 7, 4, 2, 1, 5>

|  |  |
| --- | --- |
| Submission Feedback |  |
| 1)  2) -2  3) -2  4) -5 need to follow the steps in the textbook and show the figures at each step  5) -5 need to follow the steps in the textbook and show the figures at each step | |

**Assignment #1**

**Solutions**

1. Give asymptotic upper and lower bounds for T(n) in the following recurrences. Assume that T(n) is constant for n ≤ 2. Make your bounds as tight as possible, and justify your answers.

T(n) = 4 T(n/4) + n3.

Since this is a recurrence equation of the form T(n) = aT(n/b) + f(n) where a = b = 4 and f(n)= n3, log4 4 = 1.

We can use the Master theorem to get the asymptotically tight bound. Noting that f(n) = n3=Ω(n1+ϵ) with ϵ=1, and

4f(n/4) = 4n3/64 = n3/16 ≤ f(n)/16.

Thus, we can apply the case 3 of the Master theorem to see that

T(n)=Θ(n3).

1. Exercise 4.5-1 (c) on page 96.

T(n) = 2T(n/4) + 1. We see that a=2, b=4, and log42 = ½. Let ϵ=1/4.

Since f(n) = n= Ω(n1/2 + ϵ) with ϵ=1/4, and

2f(n/4) = 2n/4 = n/2 ≤ f(n)/2,

we can apply the case 3 of the Master theorem to see that

T(n)=Θ(n).

1. Exercise 4.5-1 (d) on page 96.

T(n) = 2T(n/4) + n2.

We see that a=2, b=4, and log42 = ½. Let ϵ=1/4.

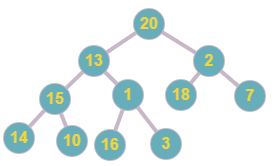
This equation falls to the case 3 of the Master theorem since f(n) = n2 = Ω (n1/2+ϵ) and 2f(n/4) = 2n2/16 = n2/8 ≤ f(n)/8.

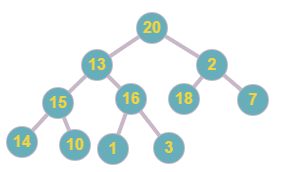
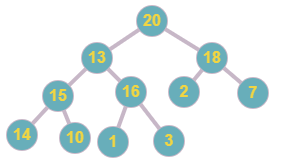
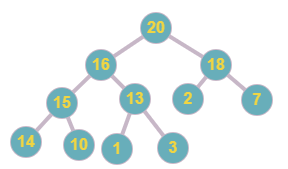
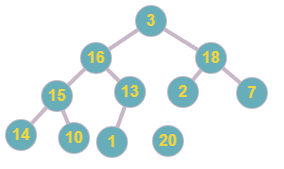
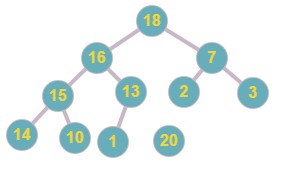
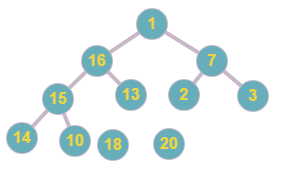
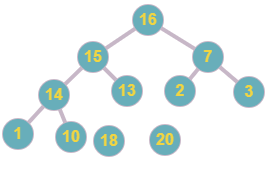
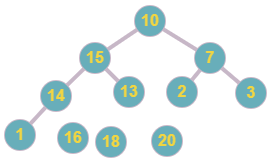
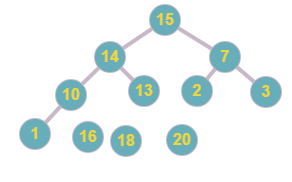
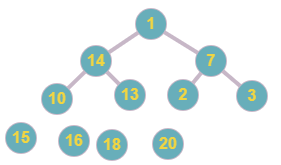
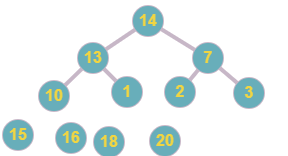
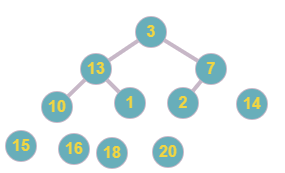
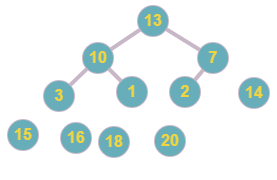
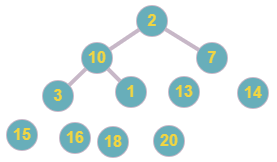
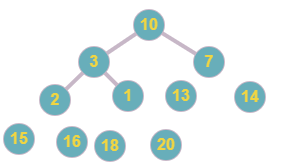
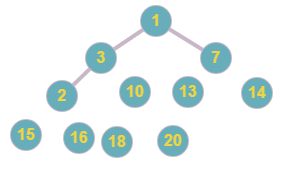
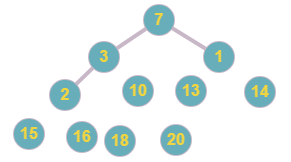
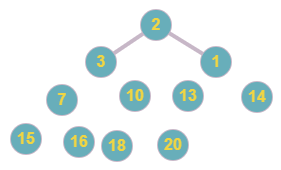
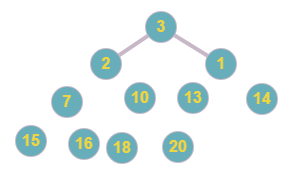
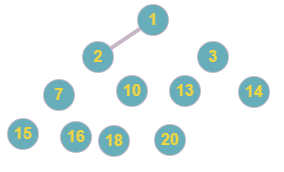
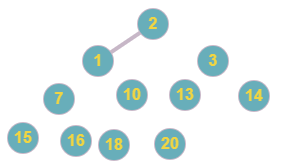
Hence T(n) = Θ (n2)

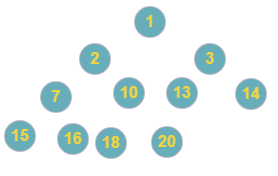
(4) Exercise 6.4-1 on page 160 on the array (HEAPSORT)

A = <20, 13, 2, 15, 1, 18, 7, 14, 10, 16, 3>.

1. Initial state



1. BUILD-MAX-HEAP(A), MAX-HEAPIFY(A, floor(11/2)=5)  
   
2. MAX-HEAPIFY(A,3)  
   
3. MAX-HEAPIFY(A,2)  
   
4. Exchange A[1] and A[11]. Heap size decrements by 1.  
   
5. MAX-HEAPIFY(A,1)  
   
6. Exchange A[1] and A[10]. Heap size decrements by 1.  
   
7. MAX-HEAPIFY(A,1)  
   
8. Exchange A[1] and A[9]. Heap size decrements by 1.  
   
9. MAX-HEAPIFY(A,1)  
   
10. Exchange A[1] and A[8]. Heap size decrements by 1.  
    
11. MAX-HEAPIFY(A,1)  
    
12. Exchange A[1] and A[7]  
    
13. MAX-HEAPIFY(A,1)  
    
14. Exchange A[1] and A[6]. Heap size decrements by 1.  
    
15. MAX-HEAPIFY(A,1)  
    
16. Exchange A[1] and A[5]. Heap size decrements by 1.  
    
17. MAX-HEAPIFY(A,1)  
    
18. Exchange A[1] and A[4]. Heap size decrements by 1.  
    
19. MAX-HEAPIFY(A,1)  
    
20. Exchange A[1] and A[3]. Heap size decrements by 1.  
    
21. MAX-HEAPIFY(A,1)  
    
22. Exchange A[1] and A[2]. Heap size decrements by 1.

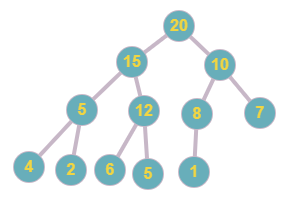


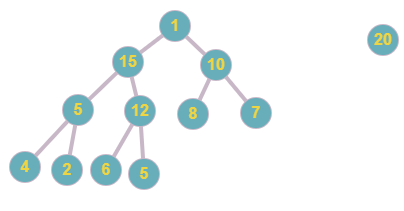
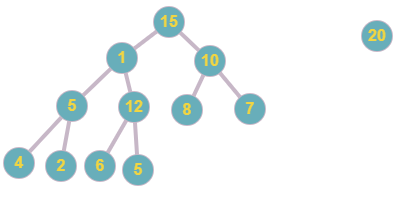
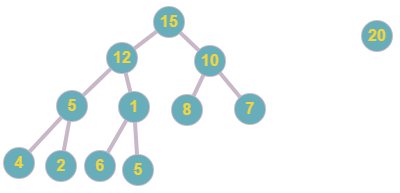
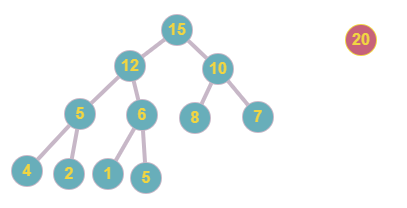
1. A is now sorted. A=<1, 2, 3, 7, 10, 13, 14, 15, 16, 18, 20>

(5) Exercise 6.5-1 on page 164 with the heap

A = <20, 15, 10, 5, 12, 8, 7, 4, 2, 6, 5, 1>.

1. Initial state



1. Set 20 as max. Move the last leaf node to top. Decrement heap size by 1.  
   
2. MAX-HEAPIFY(A, 1)  
   
3. MAX-HEAPIFY(A, 2)  
   
4. MAX-HEAPIFY(A, 5).  
   
5. Return 20.