Algorithms and Data Structures (DAT3/SW3)

Re-Exam Assignments

Chenjuan Guo & Bin Yang

14 February 2020

Full name:	
Student number:	
E-mail at student.aau.dk:	

This exam consists of three exercises and there are **two** hours (11.00 a.m. to 13.00 p.m.) to solve them. When answering the questions in exercise 1, mark the check-boxes on this paper. Remember also to put your name and your student number on any additional sheets of paper you will use for exercises 2 and 3.

During the exam you are allowed to consult books and written notes. However, the use of any kind of electronic devices with communication functionalities, e.g., laptops, tablets, and mobile phones, is **NOT** permitted. You can bring an old fashioned calculator.

- *Read* carefully *the text of each exercise before solving it! Pay particular attentions to the terms in* **bold**.
- **CLRS** *refers to the textbook—T.H. Cormen, Ch. E. Leiserson, R. L. Rivest, C. Stein,* Introduction to Algorithms *(3rd edition).*
- *For exercises 2 and 3, it is important that your solutions are presented in a readable form. In particular, you should provide precise descriptions of your algorithms using pseudo-code or reference existing pseudo-code from the textbook CLRS. To get partial points for not completely correct answers, it is also worth to write two or three lines in English to describe informally what the algorithm is supposed to do.*
- *Make an effort to use a readable handwriting and to present your solutions neatly.*

Exercise 1 [60 points in total]

Note that each following question has only a single correct solution.

1. Identifying asymptotic notation. (Note: lg means logarithm base 2).

1.1. (*5 points*) $5(n^3)^2 + \sqrt{n^7}$ is:

☐ **a)** $\Theta(n^7)$ ☐ **b)** $\Theta(n^6)$ ☐ **c)** $\Theta(n^5)$ ☐ **d)** $\Theta(n^4)$

1.2. (*5 points*) $n^2\sqrt{\lg n}$ is:

☐ **a)** $O(n^2)$ ☐ **b)** $\Theta(n \lg n)$ ☐ **c)** $O(n \lg n)$ ☐ **d)** $\Omega(n)$

2 (*10 points*) Consider the following recurrence:

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 4 \cdot T(n/4) + n \qquad (n > 1), \end{aligned}$$

2.1. (*2 points*) Is this recurrence in the format that can be solved by the master method? If yes, choose which case should be used?

☐ **a)** No, the master method cannot solve the recurrence
☐ **b)** Yes, case 1 should be used
☐ **c)** Yes, case 2 should be used
☐ **d)** Yes, case 3 should be used

2.2. (*4 points*) If you could apply the master method, choose possible a and b values.

☐ **a)** $a = 4, b = 4$ ☐ **b)** $a = 4, b = \frac{1}{4}$
☐ **c)** $a = 1, b = 4$ ☐ **d)** None of above.

2.3. (*4 points*) Choose the correct solution for $T(n)$.

☐ **a)** $\Theta(n \lg n)$ ☐ **b)** $\Theta(n)$ ☐ **c)** $\Theta(\sqrt{n})$ ☐ **d)** $\Theta(\lg n)$

3. (*10 points*) Consider the MERGESORT(A, p, r) algorithm on p. 34, CLRS, which is also shown below.

```
MERGESORT(A, p, r)
 1  if p < r
 2      q=⌊(p+r)/2⌋
 3      MERGESORT(A, p, q)
 4      MERGESORT(A, q + 1, r)
 5      MERGE(A, p, q, r)
```

3.1. (*4 points*) Which of the following statements is true?

- ☐ **a)** The worst case time complexity of merge sort is $\Theta(n^2)$.
- ☐ **b)** Merge sort is an in place sorting algorithm.
- ☐ **c)** The average case time complexity of merge sort is $\Theta(n \lg n)$.
- ☐ **d)** None of above is true.

3.2. (*6 points*) Given a sequence of numbers $A = \langle 4, 5, 7, 2, 1, 6, 8, 3 \rangle$, and we call MERGESORT$(A, 1, 8)$. In the following sequences, which one is the sequence A after the MERGE() procedure has been executed **three** times?

- ☐ **a)** `2, 4, 5, 7, 1, 3, 6, 8`
- ☐ **b)** `4, 5, 2, 7, 1, 6, 3, 8`
- ☐ **c)** `1, 2, 3, 4, 5, 6, 7, 8`
- ☐ **d)** `2, 4, 5, 7, 1, 6, 8, 3`

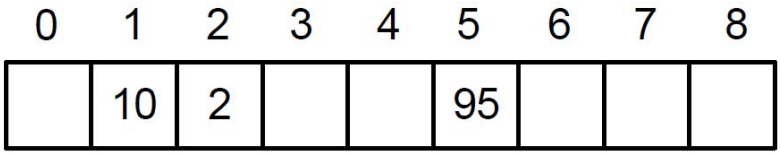
4. (*10 points*) Consider the following arrays. Which array may represent a **max-heap**?

- ☐ **a)** 27, 21, 25, 30, 32, 6
- ☐ **b)** 14, 17, 16, 18, 13
- ☐ **c)** 26, 19, 20, 17, 15, 14, 9, 10
- ☐ **d)** 5, 3, 2, 6

5. (*10 points*) Consider a hash table with 9 slots with the following hashing function that uses linear probing to address conflicts.

$$h(k, i) = (k + i) \mod 9$$

Here, k is the key and i is the probe number (where $i = 0, 1, \ldots, 8$).
Assume that we already have the following numbers in the hash table.



After inserting 3, 8, and then 11 into the table, what does the hash table look like now?

- ☐ **a)** `11, 10, 2, 3, empty, 95, empty, empty, 8`
- ☐ **b)** `empty, 10, 2, 3, empty, empty, 11, empty, 8`
- ☐ **c)** `empty, 10, 2, 3, 95, 11, empty, empty, 8`
- ☐ **d)** `empty, 10, 2, 3, 11, 95, empty, empty, 8`

6. (*10 points*) Assume that we have a ***binary tree*** T. Note that T is a binary tree, but not necessarily a binary search tree.

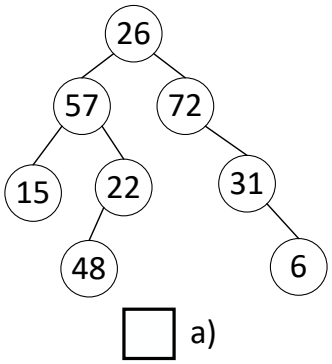
An in-order tree walk on T produces the following:

`15, 22, 57, 26, 48, 72, 6, 31`

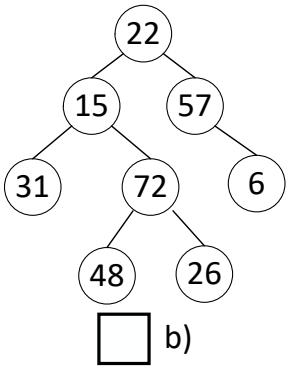
and a pre-order tree walk on T produces the following:

`26, 57, 15, 22, 48, 6, 72, 31`

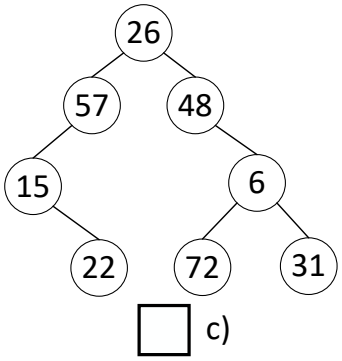
6.1. (*6 points*) Which of the following tree is correct for T?



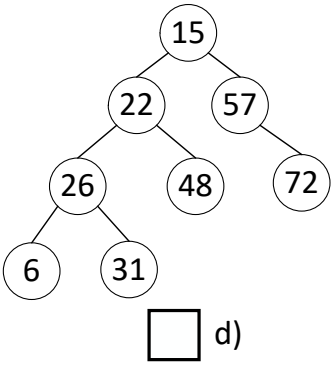
☐ a)



☐ b)



☐ c)



☐ d)

6.2. (*4 points*) What does a post-order tree walk on T produce?

☐ **a)** `22, 15, 57, 72, 31, 6, 48, 26`

☐ **b)** `26, 57, 48, 15, 6, 22, 72, 31`

☐ **c)** `22, 15, 57, 26, 48, 6, 72, 31`

☐ **d)** `15, 22, 57, 26, 72, 31, 48, 6`

Exercise 2 [20 points]

We have seen how to implement a queue using an array in lecture 7. Now let's consider how to implement a queue using a **singly** linked list.

Assume that elements in the queue are integers. It means that we use a singly linked list L to store the integers in the queue, and we need to implement the two most important operations of queues: ENQUEUE($List\ L, int\ a$) and DEQUEUE($List\ L$). Specifically,

1. ENQUEUE($List\ L, int\ a$) inserts a new element whose key equals to a to the tail of the queue, and

2. DEQUEUE($List\ L$) removes the head element of the queue, which is the element that is in the queue for the longest time.

Assume that the queue is not empty and contains N elements at the time when we call ENQUEUE($List\ L, int\ a$) and DEQUEUE($List\ L$).

Please work on:

1. (*10 points*) Briefly describe how to implement ENQUEUE($List\ L, int\ a$). Write down the pseudo code of ENQUEUE($List\ L, int\ a$). Provide asymptotic analysis of your pseudo code.

2. (*10 points*) Briefly describe how to implement DEQUEUE($List\ L$). Write down the pseudo code of DEQUEUE($List\ L$). Provide asymptotic analysis of your pseudo code.

Exercise 3 [20 points]

Given an array A that consists of n integers, we want to identify the smallest integer in the array.

1. (*7 points*) Describe a **divide-and-conquer** algorithm to solve the problem. Write the pseudo code. Note that ***ONLY*** **divide-and-conquer** algorithms will be accepted.

2. (*6 points*) Write down the recurrence of your divide-and-conquer algorithm. Explain why you have the recurrence. In particular, how many sub-problems do you get at each step, what is the size of each sub-problem, and what is the cost of combining solutions from sub-problems?

3. (*4 points*) Solve the recurrence and identify the run-time complexity of your divide-and-conquer algorithm.

4. (*3 points*) Identify the asymptotic space overhead of your divide-and-conquer algorithm.