Midterm Study Worksheet

Monday, Jan 31, 2000

This is the official worksheet for our midterm, which will be on Friday, February 4, 2000.

You are strongly encouraged to complete this worksheet. The midterm assumes that you are familiar with the content and style of these questions. You may also wish to look at the unofficial worksheet handed out last section (see calendar in course web). It has more questions on operations with data structures.

Some questions on this worksheet may be more open-ended than on the midterm (especially since we checked the worksheet a lot less than the midterm!). The midterm will have some open-ended questions, though.

Note that some diagrams may appear on the next page, in order to keep diagrams in the right place in the text. You might need to look at the next page sometimes to see the whole question.

1

Consider the following pseudocode:

```
function Erthelliant (int $S, Array[1 \ldots N]$)
$X = 0$
for $i = 1, \ldots, N$ do
    $X = X + Array[i]$
    if $X > S$ then
        return $i$
    end if
end for
```

For the function Erthelliant, what is its:

1. Worst-case running time
2. Best-case running time
3. Average-case running time

2

Elsie the Cow is often very hungry. In order to help herself to organize her grazing time, she bought a PDA (A "Hoof Pilot"). Unfortunately, she found that there isn't any software for the PDA that does what she wants.

Elsie wants to organize her time by keeping a list of the tastiest grass, which she ranks from 1 to 100 (100 being “Moooovelous”). As she discovers more grass, she needs to be able to enter it into the PDA.

One complication is interference from other grazing cows. Most other cows graze at certain locations on an hourly schedule. So, some grass may not be available to her at certain hours of the day (e.g. 4-5 pm). So, she needs to be able to enter which hours grass is available, and she may have to update this information as she learns more about the schedules of the other cows.

Whenever she’s hungry, she wants to be able to tap into her PDA. Her PDA should be able to tell her the location of the tastiest grass that she hasn’t already eaten. The PDA should only tell her about grass that is available during the current hour.

1. State the requirements of Elsie’s application. What data will it store, what operations does it need to perform, and what are the requirements on those operations?
2. Describe 2-3 of the best ways to design this application for Elsie, and what the appropriate data structures and algorithms would be. Why do you think that these are the most appropriate data structures/algorithms?
3. Of these options, pick one solution as the best. Justify your decision in the context of the requirements.
4. Write pseudocode for some of the major operations in the application (you can assume that someone has already implemented the relevant data structures for you).
5. What if Elsie wanted to be able to share her information on available grass with her friend Beatrice the Cow? Sketch how this might change things.

3

Suppose we have two algorithms. The first algorithm takes $o(n^2\sqrt{n})$ time, while the second algorithm takes $\omega(n^{2.51})$ time.

How big does n have to be before we can say anything about which algorithm is faster?

4

Give a tight worst case bound (Θ notation) on the running time of the following code:

```
function Heapmin ($A[1 \dots n]$) : integer
$Q$ = new 4-Heap
for $i = 1 \dots n$ do
    $Q$.Insert($A[i]$)
end for
return $Q$.DeleteMin
```

5

Suppose we wish to merge two Skew heaps.

1. What is the most amount of time that this could take (in Θ-notation)? Give an example of two skew heaps that would take this amount of time (it should be possible to generalize your skew heaps to any arbitrary size, n).

2. What is the least amount of time that this could take (in Θ-notation)? Give an example.

6

A dynamic programming algorithm uses an $n \times n$ table. In each element of the table, it stores a solution to a subproblem.

Suppose it needs to calculate the value of a cell in the table, $T(i, j)$. To do this, it must look at all cells $T(x, y)$ such that $x < i$ and $y < j$ (the cells up and to left of the cell $T(i, j)$). It finds the minimum of these cells, and adds $i + j$ to this minimum value.

What is the running time of this algorithm?

7

1. Name 3 differences between Binary Search Trees and B-trees.

2. What are the differences and similarities between AVL trees and Splay Trees?

3. What are the differences and similarities between AVL trees and B-trees?

4. What are the differences and similarities between AVL trees and Binary Search trees?

5. Under what circumstances would you probably decide to use a BST?

6. How about a B-tree?

7. An AVL-tree?

8. A Splay Tree?

8

Each of the following pictures represents one of the following data structures:

- 3-Heap
- Splay tree
- B-tree with branching factor $M = 4$, and maximum number of keys in a leaf, $L = 2$.
- Circular array implementation of a queue, with strings (and no NULLs).

However, each data structure has exactly one flaw in it, which can be fixed with one of the following four operations:

- Percolate up
- Remove node

- Split
- Decrement end pointer

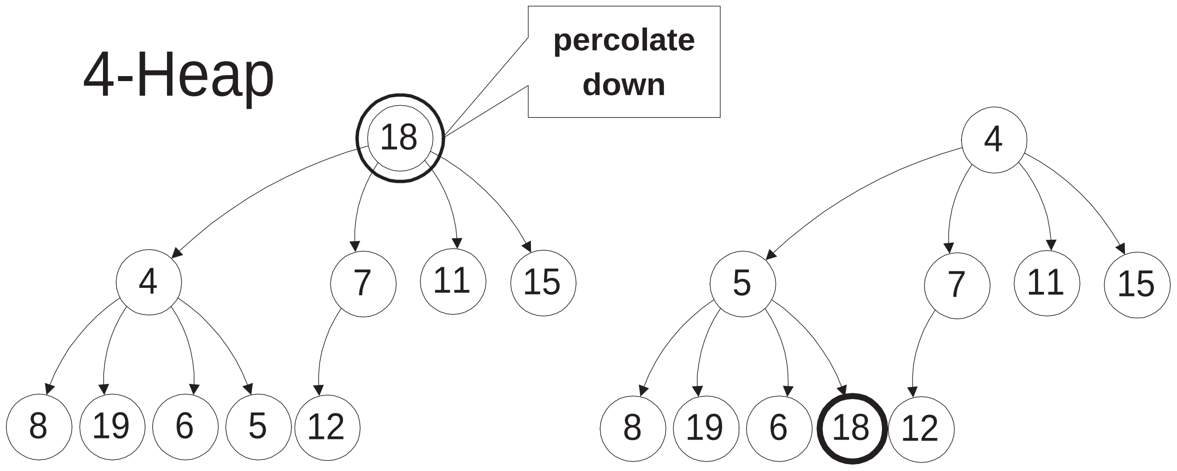
You should:

- Write exactly one data structure and one operation from these lists next to each data structure.
- Circle the problem in each data structure.
- Draw a picture of what each data structure looks like after fixing its problem with its operation.

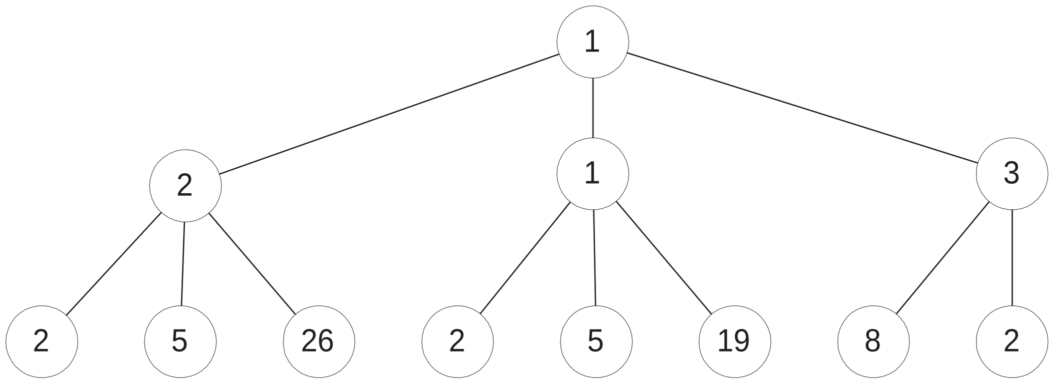
Your new data structure diagram should be a valid instance of the data structure you use to label it!

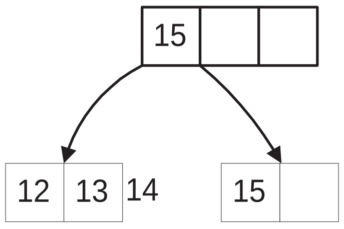
Several data structures or operations may match one picture, but only one assignment puts each data structure and each operation with exactly one picture. So, use each data structure and operation exactly once!

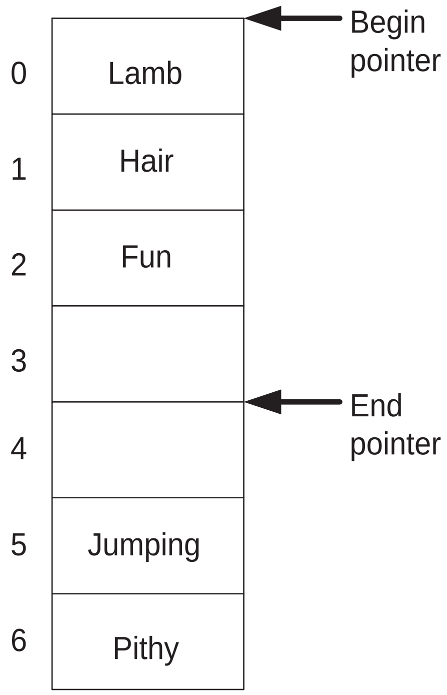
For example, here is a solution for a 4-Heap picture. I have labelled the picture as a 4-Heap, circled the problem node, listed the operation (**percolate down**) to fix the problem, and drawn the new, problem-free 4-Heap.

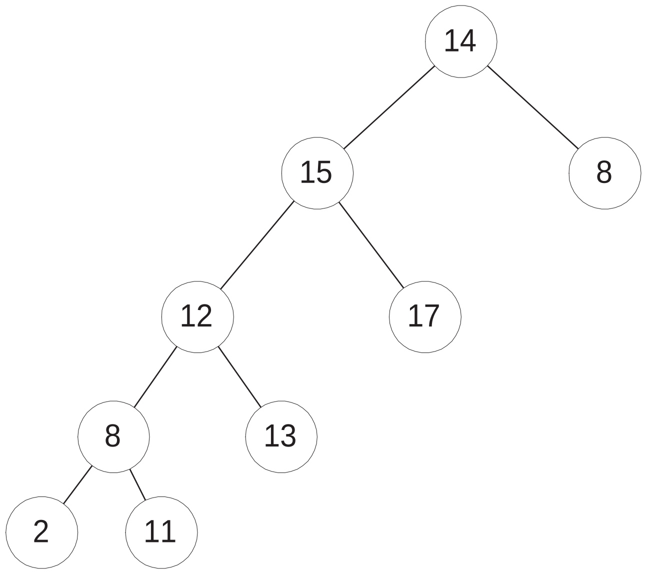


Now, here are the pictures (they start on the next page)...

1. 

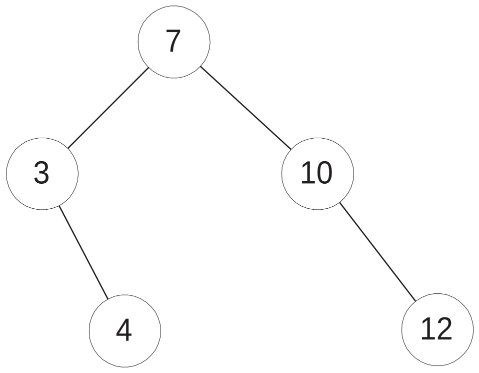
2. 

3. 

4. 

9

1. Insert 9 into the following AVL tree. What is the result?
2. What would the result have been if you had instead inserted 11?
3. What if it was instead 13?



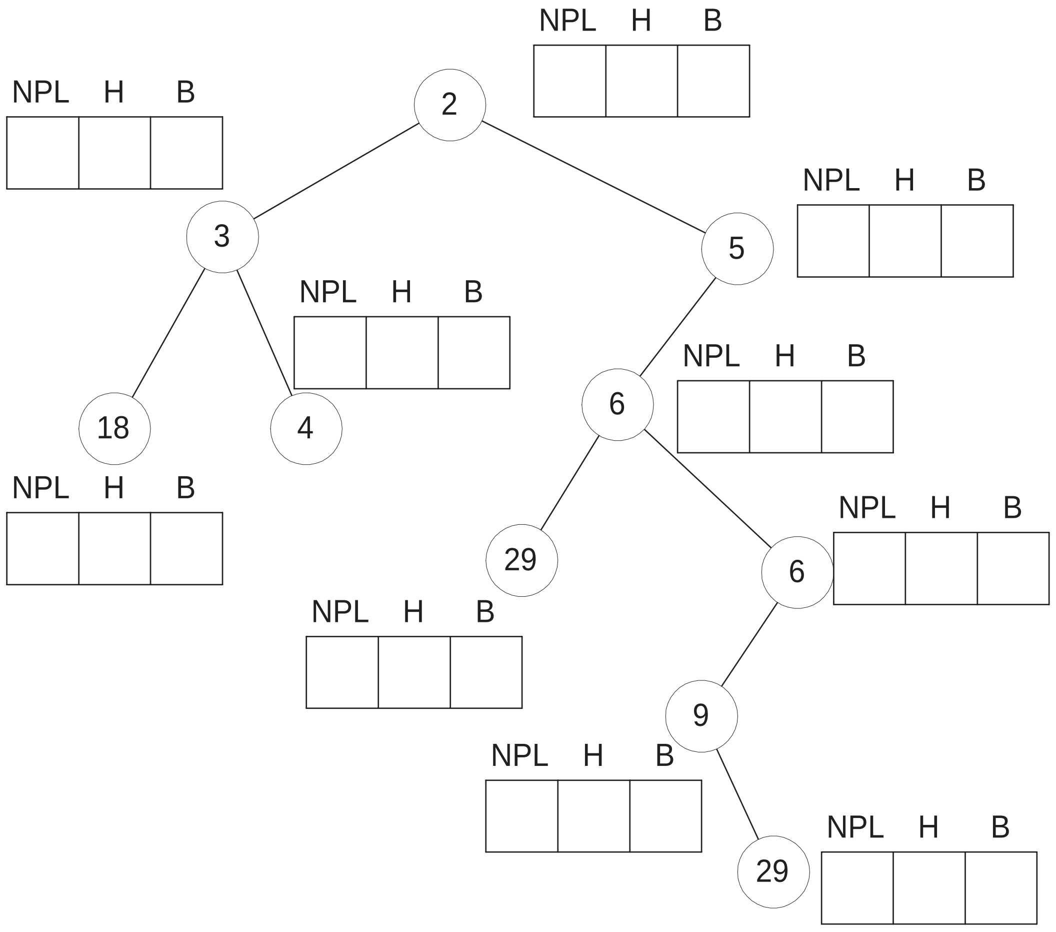
10 Capitalist trees: trees with properties

Label the nodes in the following tree with the following properties:

- NPL, the Null Path Length.
- H, the height.
- B, the balance.

Also, indicate which of the following data structures this tree might represent:

- Binary Search Tree
- AVL Tree
- B-Tree
- Splay Tree
- Stack
- Skew Heap
- Leftist Heap
- Queue
- d-Heap



11

Prof. Zara Wolfbone does research in Comparative Languages at the University of Data in Structures. She wants to find out what are the most frequent words in different languages (like “a” and “the” in English). Then, she will compare the most frequent words in different languages. She hopes to discover fundamental similarities between different human languages.

Her plan is to find the most frequent words in a variety of works in different languages, ranging from all known works in electronic form, to individual authors in those languages (e.g. Shakespeare, Lorca or Sun-Tzu). She also wishes to investigate individual short essays by people learning those languages as a second language. All works she will consider are already in electronic form.

Prof. Wolfbone’s current research grant is about to expire, so she needs something to help her quickly.

Congratulations! You have been selected to create a computer application to help in Prof. Wolfbone’s research.

1. Define a set of requirements for Prof. Wolfbone’s application. If some of the requirements are vague as described, make an educated guess as to what she wants (justify your guesses).

2. Propose 2 reasonable solutions to this problem. What data structures and algorithms have you used in each solution? Why did you pick those data structures/algorithms? How do they fit together to make the whole application.

3. Pick the solution you think is best. Defend your solution according to the requirements.

12

What are the advantages and disadvantages of coding algorithms iteratively versus recursively?