Question **11**

Not yet answered

Marked out of 1.00

Write a subalgorithm that finds the maximum value in a binary tree containing integer numbers.

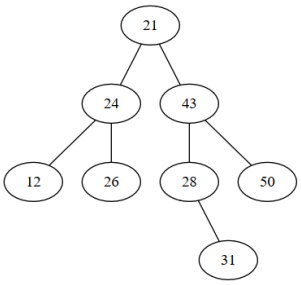
Requirements:

- Give the representation of the binary tree

- Implement the subalgorithm.

- Specify and explain the complexity of the operation.

For example, in the binary tree below, the maximum value is 50.



Maximum file size: 2MB, maximum number of files: 3

Files

You can drag and drop files here to add them.

Question **12**

Not yet answered

Marked out of 1.00

Consider the following problem: we want to implement ADT Quartiler, which contains integer numbers and has the following operations (with the specified complexity requirements).

- init(q) - creates a new, empty Quartiler - Θ(1) - total complexity

- add(q, elem) - adds a new element to the Quartiler q - $O(\log_2 n)$ - amortized complexity

- getTopQuartile(q) - returns the element closest to the 75^{th} percentile (explanations below). If there is no such element, throws an exception - Θ(1) - total complexity

- deleteTopQuartile(q) - removes the element closest to the 75^{th} percentile. If there is no such element, throws an exception - $O(\log_2 n)$ - total complexity

Explanation: the 75^{th} percentile (called 3^{rd} quartile as well) of a sequence is a value from the sequence, the one below which 75% of the values from the sequence can be found if we sort the sequence. So, if you have the values from 1 to 100 (in any order), the 75^{th} percentile is the value 75. If you have the values 1,2,3,4, the 75^{th} percentile is 3. In case of a tie (for example, if you have values 1,2,3,4,5,6 the value 4 or 5 can be returned).

1. Which data structure (out of the ones discussed during the lectures) would you use as a representation for the Quartiler?

2. Explain in short how would you implement each operation of the Quartiler, and why the implementation fits the complexity requirement.

3. Give the representation of the Quartiler and implement in pseudocode the deleteTopQuartile operation.

Obs: 1. You can ignore memory deallocation altogether when computing the complexity of an operation. You can assume that you can deallocate anything (including a linked list with dynamic allocation, for example) in Θ(1) complexity.

2. You do not have to implement resize.

3. You need to implement every function you need for question 3, you cannot assume that you already have operations implemented (unless specified differently). But you can define (and implement) auxiliary functions.

Maximum size for new files: 2MB

Files

You can drag and drop files here to add them.

Question **13**

Not yet answered

Marked out of 1.00

Show how a stack can be implemented using two queues. You can assume that the Queue ADT is already implemented (we don't know how and we don't care how). You will have to implement the Stack ADT using the operations of the Queue.

Requirements:

- Give the specifications for the operations of the Queue.
- Give the representation of the Stack.
- Implement ALL Stack operations.
- Specify the complexity for every stack operation, assuming Θ(1) complexity for the queue operations.

Maximum file size: 2MB, maximum number of files: 3

Files

You can drag and drop files here to add them.

◄ Lecture 14 Quiz

Jump to...