Assignment No. 6: Dynamic Order Statistics

Allocated time: 4 hours

Implementation

You are required to implement **correctly** and **efficiently** the management operations of an **order statistics tree** (chapter 14.1 from the book[1]).
You have to use a balanced, augmented Binary Search Tree. Each node in the tree holds, besides the necessary information, also the *size* field (i.e. the size of the sub-tree rooted at the node).

The management operations of an **order statistics tree** are:

- BUILD_TREE(n)
  - *builds* a balanced BST containing the keys 1,2,...n (*hint:* use a divide and conquer approach)
  - make sure you initialize the size field in each tree node
- OS-SELECT(tree, i)
  - selects the element with the ith smallest key
  - the pseudo-code is available in chapter 14.1 from the book[1]
- OS-DELETE(tree, i)
  - you may use the deletion from a BST, without increasing the height of the tree (why don't you need to rebalance the tree?)
  - keep the size information consistent after subsequent deletes
  - there are several alternatives to update the size field without increasing the complexity of the algorithm (it is up to you to figure this out).

Does OS-SELECT resemble anything you studied this semester?

Thresholds

Threshold	**Requirements**
5	BUILD_TREE - correct and efficient implementation; demo for n=11,
7	OS_SELECT & OS_DELETE - correct and efficient implementation, demo
9	Management operations evaluation
10	Interpretations, discussion

Evaluation

! Before you start to work on the algorithms evaluation code, make sure you have a correct algorithm! You will have to prove your algorithm(s) work on a small-sized input (11) i.e. pretty-print the initially built tree and, for a few elements (3), OS-SELECT by a randomly selected index and pretty-print the tree after its OS-DELETE).

Once you are sure your program works correctly:

- vary n from 100 to 10000 with step 100;
- for each n (don't forget to repeat 5 times),
    - build a tree with elements from 1 to n
    - perform n sequences of OS-SELECT and OS-DELETE operations using a randomly selected index based on the remaining number of elements in the BST

Evaluate the computational effort as the sum of the comparisons and assignments performed by each individual management operation.