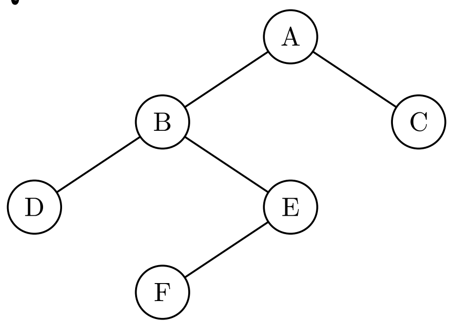
PROTOTYPE MIDTERM II, WINTER 2020 — COMPUTER SCIENCE 251

Examiner: Luc Devroye
Instructions: Calculators and computers are not allowed.
Time: 90 minutes.
Weight: Question 2 and 7 have slightly higher weight than the other questions.

Question 1.

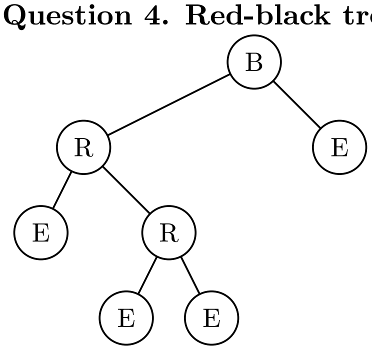


For the tree shown here, answer these questions:
(i) Give the preorder listing.
(ii) Give the postorder listing.
(iii) Give the inorder listing.
(iv) Give the least common ancestor of nodes D and F.
(v) Give the Harris walk sequence.

Question 2. Augmented binary search trees. With the standard implementation using fields `left[·]`, `right[·]`, and `key[·]`, and augmented with subtree size information in the field `size[·]`, give an efficient non-recursive algorithm, `select`(k, t), for finding the k-th smallest key stored in tree t. For simplicity, assume that t is not empty, and that k is in the allowed range (between 1 and the size of t).

Question 3. Binary heap. A binary heap (with minimal key at the root) is implemented using cells that are connected with pointers, and fields `left[·]`, `right[·]`, and `key[·]`. Give an $O(k \log k)$ algorithm for `smallest` (k, t), that prints (but does not delete) the k smallest keys stored in the binary heap. Note that this complexity does not depend upon the size n of the binary tree. If this is too hard, get partial credit by giving an $O(k \log n)$ algorithm.

Question 4. Red-black trees.



For the (illegal) red-black tree shown here, the labels R, B and E refer to red, black, and external nodes, respectively. Rotate the red-black tree to fix the problem. Fill in the labels R, B and E in your solution.

Question 5. Binary search tree. We are given a binary search tree t with just pointers to left and right children, but no parent pointer. Its height is h. Without using key comparisons, and performing only $O(h)$ computations in all, outline the strategy for an algorithm, `check (t, x, y)`, that checks whether x and y are consecutive nodes in an inorder traversal.

Question 6. Interval trees. We have an interval tree with root t, in which the fields in a cell are denoted by `left[·]`, `right[·]`, `color[·]`, `rank[·]`, `low[·]`, `high[·]`, `max[·]`. Recall that `max[`u`]` is the maximal `high` value in the subtree rooted at u. The following program is supposed to answer "yes" or "no" to the query: "Is a given interval $[a,b]$ completely contained in an interval stored in the interval tree?" (An interval $[a,b]$ is contained in $[c,d]$, written $[a,b] \subseteq [c,d]$, if $c \le a \le b \le d$.) Only XXX, YYY and ZZZ are missing in the algorithm below. Here XXX is a comparison, and YYY and ZZZ are each one of these: `contained (left[t], [a,b])`, `contained (right[t], [a,b])`, "yes", "no". Please determine XXX, YYY and ZZZ. The program is recursive and should take time $O(\log n)$ where n is the size of the tree.

```
contained (t, [a,b])
  if t = nil, then return ‘‘no’’
  if [a,b] ⊆ [low[t],high[t]] then return ‘‘yes’’
  if left[t]=nil then return contained (right[t], [a,b])
  if a ≤ low[t], then return contained (left[t], [a,b])
  else if XXX then return YYY
              else return ZZZ
```

Question 7.

(i) True or false: A k-d tree a binary tree.

(ii) True or false: $2n$ bits suffice to store the shape of an ordered tree of size n.

(iii) True or false: Any unsorted list of size n can be turned into a binary heap using $O(n)$ comparisons.

(iv) True or false: It is possible to turn any unsorted list of size n into a binary search tree using $O(n)$ comparisons.

(v) True or false: Given both a preorder and a postorder listing of the nodes of a binary tree, one can reconstruct that tree.

(vi) If a node in a red black tree has rank 5, what are the possible ranks of its great-grandparent (i.e., three generations above)?

(vii) In the binary comparison model, rank from small to large: (A) the worst-case time of quicksort, (B) the expected time of quicksort, (C) the worst-case time of mergesort.

(viii) In a treap with n nodes, what is the expected number of nodes on the left roof? (An exact expression valid for all n is expected. E.g., for $n = 1$, the answer should be one.)

(ix) If a tree of size n has only nodes with zero or two children, then how many leaves does it have?

(x) If a tree of size n has only nodes with zero or three children, then how many leaves does it have?

(xi) Consider a red-black tree whose root has rank r. What is the maximal number of nodes in the tree (including the external nodes)?