T-301-REIR, REIKNIRIT
HAUST 2018
D3 - SYMBOL TABLES

Problem 1. *Suppose that the keys A through G, with the hash keys given below, are inserted in some order into an initially empty table of size 7 using linear probing (M=7, no resizing).*

key	*A*	*B*	*C*	*D*	*E*	*F*	*G*
hash	*2*	*0*	*5*	*4*	*4*	*4*	*2*

Which of the following (more than one might apply) could not possibly result from inserting these keys? Explain briefly.

(1) B E A G D F C
(2) C F A G D E B
(3) F B G A E C D
(4) F C B G A D E

Problem 2. *Write the constructor BST(double a[]), that converts the unsorted array of numbers into a corresponding (properly ordered) binary search tree. The tree should be of minimum height, and the method efficient. You may use support functions.*

Problem 3. *Josie needs a data structure that can handle the following operations:* `push, pop, contains, remove`, where `contains(Item x)` answers whether a given item is in the data structure and `remove(Item x)` removes the given item from the data structure. How can this be implemented efficiently?

Problem 4. *Given a collection of intervals (on the real line) and a real value x, a stabbing count query is the number of intervals that contain x. Design a data structure that supports interval insertions intermixed with stabbing count queries, in logarithmic time per operation.*

Problem 5. *Give a sequence of N insertions into a red-black tree that lead to the tree having height $\sim 2 \log N$. (You may assume $N = 2^k - 2$, for some k).*

Date: September 25, 2018.

Class Exercises

These questions will be addressed during exercise class. They are not to be turned in.

Problem 6. *Consider the following linear-probing hash table, using modular hashing with M=8. Five of its items have been inserted correctly; which one is incorrect?*

i	0	1	2	3	4	5	6	7
h[i]	7	22	-	11	19	-	17	14

Problem 7. *Implement in Java the method* `delEven()` that deletes from the tree all nodes, and their subtrees, whose values are even integers. You may define private methods.

Problem 8. *Give an efficient searchable stack, where in addition to push/pop, one can ask if a given element is contained in the stack.*

Problem 9. *Suppose elements are inserted in a reverse order into a red-black tree. Explain the red nodes will all be on the path from the root to the leftmost leaf.*

Problem 10. *(Optional) Design an algorithm that takes a sequence of N document words and a sequence of M query words, and finds the shortest interval in which the M query words appear. An 'interval' corresponds here to a contiguous sequence of words in the document; its size is the number of words that it contains. Give the algorithm in prose or pseudocode. It should run in linearithmic time.*

School of Computer Science, Reykjavik University, Menntavegi 1, 101 Reykjavik

Email address: mmh@ru.is