LE/EECS 4101 GS/EECS 5101
Advanced Data Structures

Sample Final Exam

Shahin Kamali

York University

April 21st, 2023

Write your name and student id here:

"Ignorance is departure from home and enlightenment is returning. ..." *D.T. Suzuki*

- **Do not open this booklet until instructed**.
- You are NOT allowed to use any printed/written material. Calculators are OK.
- Please **turn off your cell phones** and put them in your bags.
- Manage your time. We start the exam at 9:00 and end at 12:00. **You have 180 minutes**. **Don't waste too much time on a single question. Note that the first questions are likely to be the hardest ones.**
- The exam is printed **double-sided**. There will be **20 pages** in the final exam, including this cover page and two blank pages (use them if you need more space). This sample exam is typeset more compactly in 17 pages.
- It is OK to take the staples off. You must submit ALL pages.
- The marks will be scaled so that the highest mark gets the full mark.

1. Amortized Analysis & Competitive Ratio (8 marks)

(a) Suppose an algorithm A has an amortized cost of $O(1)$ per operation for the first $n/2$ operations of an input of size n, and amortized cost of $\Theta(\log n)$ per operation for the second $n/2$ operations.

True or False: The amortized cost of A per operation, over all n operations, is $\Theta(\log n)$.

(b) Consider two online algorithms A and B for the same online problem. Suppose A has a better competitive ratio than B.

True or False: The expected cost of A for a randomly generated input is better than B.

(c) An "f-bit" is a quinary digit that can have any of the values $\{0, 1, 2, 3, 4\}$. Consider a quinary counter that starts from an initial configuration where all f-bits are '0'. This is followed by m operations each adding **two units** the encoded number.
Use the potential-function method to find an upper bound for the amortized number of changed f-bits per operation. Show and justify your work.

2. Self-adjusting Structures (9 marks)

(a) Write down the BWT transform of the following string in the provided box (no need to show steps).

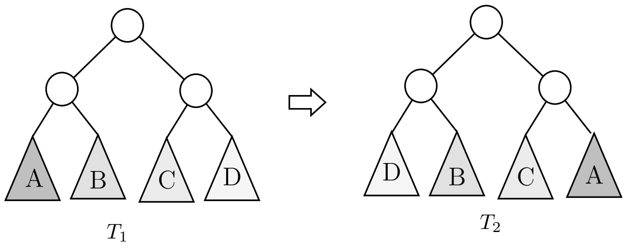
$$kootah\$$$

(b) Consider a self-adjusting list that is updated by the following variant of the Move-To-Front algorithm, which we call VAR: upon accessing an item x at position i of the list, VAR uses a free exchange to move x to the position $\lfloor i/3 \rfloor$.

Use an adversarial argument to provide a lower bound for the competitive ratio of VAR. That is, you need to present a value $\gamma > 2$ so that the competitive ratio of VAR is at least γ.

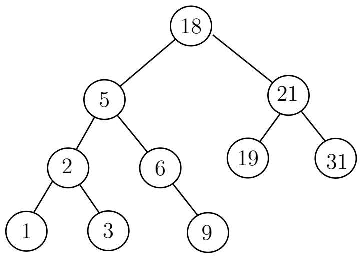
3. AVL Trees (6 marks)

(a) Suppose the following tree T_1 represents any height-balanced tree with balanced factor 1. We swap two pointers' contents, left.left with right.right, to create a new tree T_2.



True or False: T_2 is also a height-balanced tree.

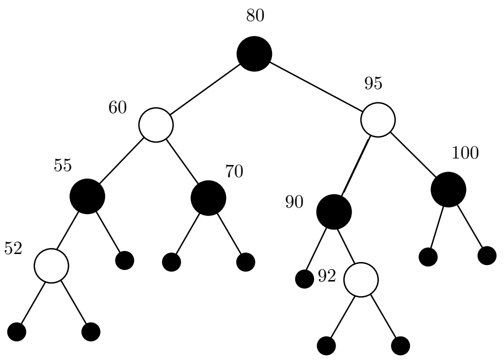
(b) Apply the operation insert(8) in the following AVL tree. It suffices to show the final tree (no need to include balance factors).



4. Red-Black Trees (6 marks)

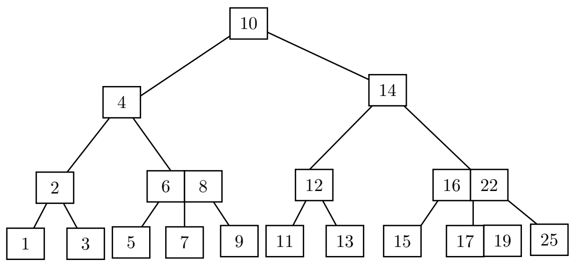
(a) **True or False:** The number of white nodes in a Red-Black tree is within a constant factor of the number of black nodes.

(b) Apply the operation insert(57) in the following Red-Black tree. It suffices to show the final tree (specify colors via shading black nodes or labelling them with letter 'B').



5. B-Trees (8 marks)

(a) Apply the operation insert(18) in the following 2-3 tree. It suffices to show the final tree.

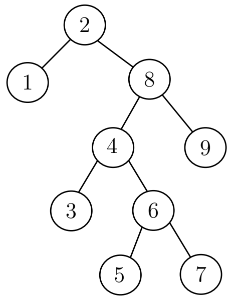


(b) Consider a "full" 2-3 tree with n nodes such that each node (including the root) has 2 keys. Compute the height of the tree. Show your work.
Hint: given a positive value a, we have $a + a^2 + \ldots + a^k = \frac{a^{k+1}-a}{a-1}$.

6. Splay Trees (5 marks)

(a) **True or False**: The second most recently accessed node in a splay tree is a child of the root.

(b) Apply the operation splay(5) in the following 2-3 tree. It suffices to show the final tree.



7. Hashing (12 marks)

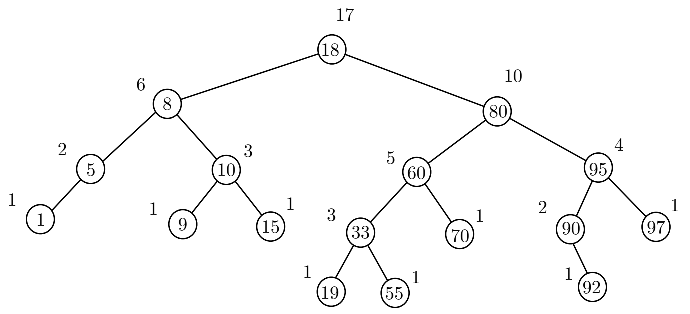
(a) Suppose we resolve collisions in a hash table using linear-probing.
True or False: After many insertions and deletions, we may need to rehash because search queries become expensive (due to "deleted" flags)

(b) Consider a hash table dictionary with size $M = 10$. If items with keys $k = 3, 53, 13, 23$ are inserted in that order, draw the resulting hash table if we resolve collisions using **double hashing** with $h_1(k) = k \mod 10$ and $h_2(k) = \lfloor k/10 \rfloor \mod 10$.

(c) Given an array of n integers, describe an algorithm that runs in $O(n)$ and reports all pairs like x and y such that $x + y = xy$.

8. Multidimensional Dictionaries (9 marks)

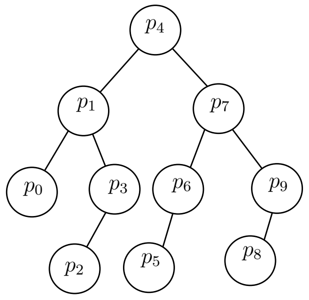
(a) Consider the following augmented AVL tree.

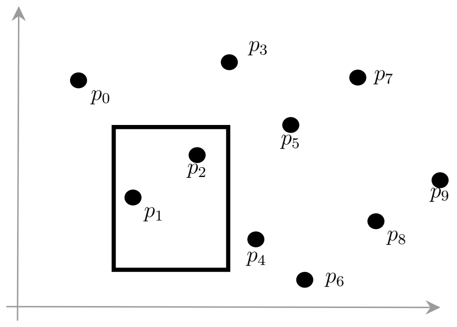


Specify the value of $rank(33)$ and how you compute it (write a sum).

(b) Consider a set of n points that are uniformly distributed in the plane.
True or False: Range queries will be faster if we store P using a quad-tree compared to when we store it using a kd-tree.

(c) We store the following set of points using a range tree; the primary tree is depicted. Specify all points whose both x- and y- coordinates are explicitly checked when reporting the points inside the highlighted query box.

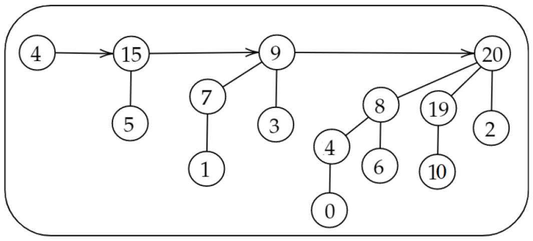




9. Binomial Heaps (6 marks)

(a) Given an unsorted array A of integers, a sorting algorithm places member of A, one by one, into a binomial heap H. Then it repeatedly extracts the maximum item from H to retrieve the sorted array (in reverse order). Indicate the time complexity of this sorting algorithm. Justify your answer in a couple of sentences.

(b) Consider the following binomial heap.
Show the resulting heap when we apply the operation insert(11) followed by operation insert(22). Show your work.

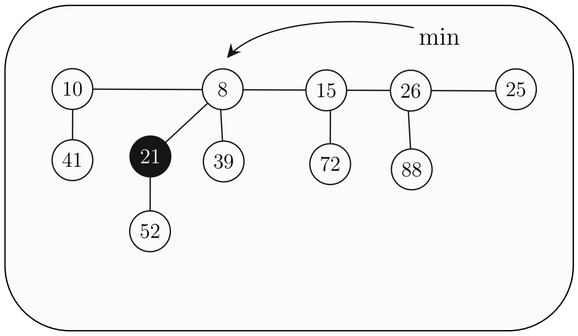


10. Fibonacci Heaps (11 marks)

(a) In the analysis of Fibonacci heaps, we define the potential to be the number of trees (and we ignore the number of marked nodes).
True or False: the new potential gives the same guarantee for the amortized time complexity of the extractMin operation.

(b) **True or False:** replacing binary heaps with Fibonacci heaps improves the running time of the algorithm for contrsucting Huffman trees.

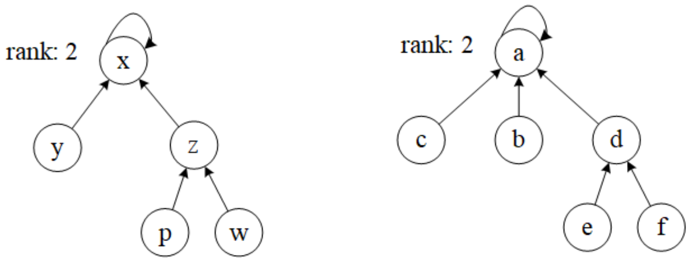
(c) In the Fibonacci heap below, apply the operation extractMin. Show your work.



11. Disjoint Sets (8 marks)

(a) **True or False**: Amortized cost of operations in a union-find data structure (forest structure with union-weight and path compression) is $o(\log \log n)$.

(b) Consider a union-find structure formed by the following trees. Draw the result after the following operations: union(y, c), find(w). For the union operation, as both trees have the same rank, assume a becomes the parent of the united tree.



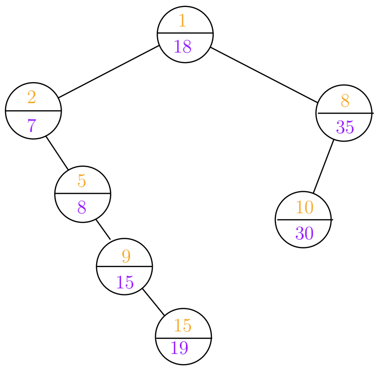
12. Randomized Structures (8 marks)

(a) Let $f(n)$ and $g(n)$ respectively denote the space complexity of a binary search tree and a skip list for maintaining the same set of keys.
True or False: $f(n) \in O(g(n))$.

(b) Starting with an empty skip list, insert keys 3, 7, 4, 5 and draw the final skip list. Assume we use the following coin tosses to determine the heights of towers. Note that not every toss is necessarily used). Recall that for each item we flip coins until we see a Tail.
(H, T, H, H, T, T, H, T)

(c) **True or False:** The following tree is a treap.

Top values indicate priorities and bottom values are keys.



13. String Structures (3 marks)

Draw the suffix tree corresponding to the text $T = cocoa$.

Use this blank page for your draft work.

Use this blank page for your draft work.