LE/EECS 4101 GS/EECS 5101
Advanced Data Structures

Midterm

Shahin Kamali

York University

March 6th, 2023

Write your name and student id here:

"Your life is your life. Don't let it be clubbed into dank submission. Be on the watch. There are ways out. There is light somewhere ..."
Charles Bukowski

- **Do not open this booklet until instructed**.
- You are NOT allowed to use any printed/written material.
- Please **turn off your cell phones** and put them in your bags.
- Manage your time. We start the exam at 5:30 and end at 7:00. **You have 90 minutes**. **Don't waste too much time on a single question. Note that the first questions are likely the harder ones.**
- The exam is printed **double-sided**. There are **14 pages**, including this cover page and four blank pages (use them if you need more space). 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. True/False and Very-Short-Answer Questions (14 marks)

Provide your short answers in the provided boxes. There is no need to justify your answers.

(a) True or False: Amortized cost of search/insert/delete operations in an AVL tree with n keys is $O(\log n)$. $\boxed{\text{True}}$

(b) The following table shows the cost of an online algorithm A and Opt for four input instances of a problem.

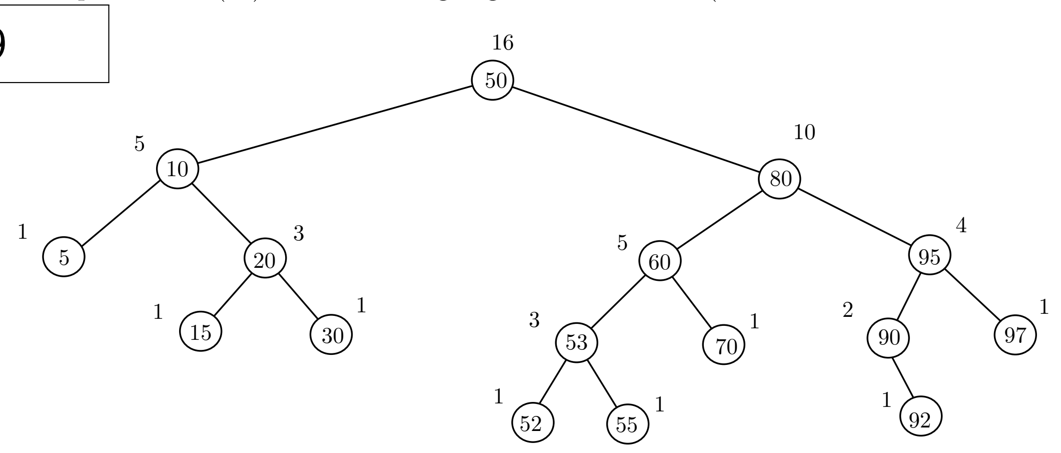
input	cost of A	cost of OPT
I_1	4	3
I_2	10	5
I_3	12	7
I_4	16	10

True or False: the competitive ratio of A is at least 2. $\boxed{\text{True}}$

(c) Suppose the list of MTF and OPT are respectively $c \to a \to b \to d$ and $a \to c \to d \to b$.

Write down the number of inversions between the two lists. $\boxed{2}$ **Answer:** The only inversions are $(a, c), (b, d)$.

(d) Write down the output to $rank(60)$ in the following augmented AVL tree (assume indices in the sorted array start at 0). $\boxed{9}$



(e) True or False: Given a red-black tree, if we switch the colour of all nodes so that all red nodes become black and all black nodes become red, the result is still a red-black tree. $\boxed{\text{False}}$ **Answer:** For example, a fully-balanced tree in which all nodes are black is a red-black tree. Switching all colors will violated red-black tree property (many adjacent white nodes will appear).

(f) Let T be a b-tree tree with $d = 10$ and with n keys, where $n \geq 2^{100}$.

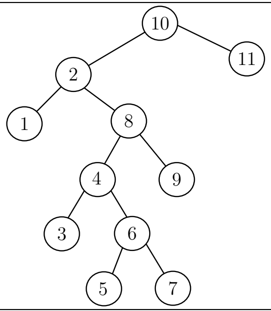
True or False: the root of T may have only one key. $\boxed{\text{True}}$ **Answer:** Root is an exception that may have as few as 1 key.

(g) True or False: a single access operation on a splay tree of size n may take $\Theta(n)$. $\boxed{\text{True}}$ **Answer:** Splay trees can be unbalanced.

2. Short Answer Questions (16 marks)

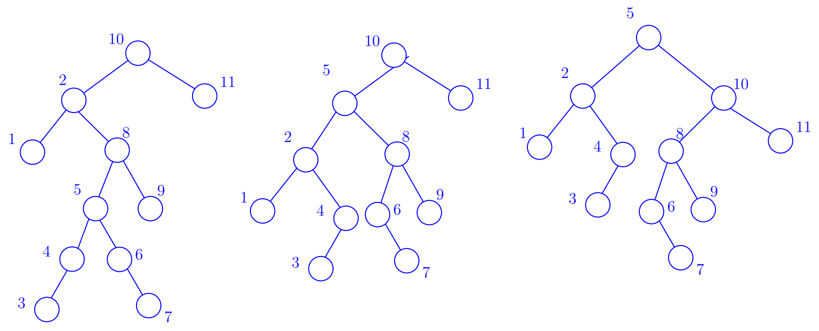
(a) We apply m search operations on an AVL tree of size n, where m and n are large integers and $m = \Theta(n)$. Suppose $3m/4$ operations take constant time each, $m/4 - \log m$ operations take $\log\log n$ time each, and the remaining $\log m$ operations take $\log n$ time each. What is the asymptotic amortized running time of each operation? Show your work.

Answer: The total cost of operations of the first type is $3m/4 \times c = \Theta(n)$. The total cost of operations of the second type is $(m/4 - \log m) \times \log\log n = \Theta(n \log\log n)$. The total cost of operations of the third type is $\log m \times \log n = \Theta(\log^2 n)$. The total cost is thus $\Theta(n) + \Theta(n\log\log n) + \Theta(\log^2 n) = \Theta(n\log\log n)$. The amortized running time is thus $\Theta(n\log\log n)/m = \Theta(\log\log n)$.



(b) Apply the splay operation on the following splay tree when there is a request to the node '5'. It suffices to show the final tree.

Answer:



(c) Apply the Burrows-Wheeler transform on the following string; show your work, and the output

$$delaram\$$$

Assume \$ precedes all characters when you sort rotations.

Answer:

```
Rotations:      Sorted Rotations:
delaram$        $delaram
elaram$d        am$delar
laram$de        aram$del          Transform:
aram$del        delaram$          mrl$deaa
ram$dela        elaram$d
am$delar        laram$de
m$delara        m$delara
$delaram        ram$dela
```

(d) Assume an initial list $\$ \to I \to O \to P \to R \to Z$. A compressing scheme that uses Move-To-Front is used to encode "PIROOZ". Show the resulting code (the first position is encoded as 0, the second as 1, and so on).

Answer:

$\$ \to I \to O \to P \to R \to Z \to$ decode P as 3
$P \to \$ \to I \to O \to R \to Z \to$ decode I as 2
$I \to P \to \$ \to O \to R \to Z \to$ decode R as 4
$R \to I \to P \to \$ \to O \to Z \to$ decode O as 4
$O \to R \to I \to P \to \$ \to Z \to$ decode O as 0
$O \to R \to I \to P \to \$ \to Z \to$ decode Z as 5
$Z \to O \to R \to I \to P \to \$$

The code is 324405

3. Amortized Analysis (8 marks)

A "trit" is a ternary digit with any of the values $\{0, 1, 2\}$. Consider a trit counter that starts from an initial configuration where all trits are '0'. This is followed by m operations, each incrementing the number encoded in ternary. In this question, we want to know how many trits are changed per operation.

(a) Use the aggregate-cost method to find an upper bound for the amortized number of changed trits.
Hint: given a positive value a, we have $1 + a + a^2 + \ldots + a^k = \frac{a^{k+1}-1}{a-1}$.

Answer: Following a similar argument that we had for a binary counter (slide 5 of Amortized Analysis), we can say the number of flips will be at most

$$\underbrace{m}_{\textit{flips of index } 0} + \underbrace{\frac{m}{3}}_{\textit{flips of index } 1} + \ldots + \underbrace{\frac{m}{3^{\lceil \log_3 m \rceil}}}_{\textit{flips of index } \lceil \log_3 m \rceil} < m \sum_{i=0}^{\infty} \frac{1}{3^i} = m \frac{1}{1 - 1/3} = 3m/2$$

Hence the amortized number of flips is 3/2 per operation.

(b) Use the potential-function method to find an upper bound for the amortized cost of each operation. Show your work.
Hint: Let the potential be the number of non-zero trits in the encoded number. Use k to denote the actual cost of an operation. You need to argue how the potential is changed.

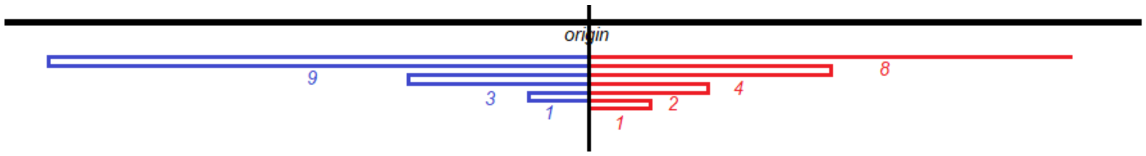
Answer: Assume an incrementing operation involves k flips. The main observation is that all the first $k - 1$ trits become '0' . For example, after incrementing '01102222', we get '01110000', where the first 5 trits are flipped, and four of them have become '0'.
For an operation involving k flips, the actual cost is k. Meanwhile, the number of zeros is increased by $k - 1$ (for the first $k - 1$ flips) and possibly decreases by 1 (when the kth trit increments from '0' to '1'). So, the number of '0's increases by at least $k - 2$. That is, if the potential is the number of non-zero items, it decreases by at least $k - 2$ after the operation. So, the actual cost k plus the difference in potential $(-(k - 2))$ will be equal to 2, which is the amortized cost.

4. Competitive Analysis (5 marks)

Consider the following algorithm for the cow-path problem. The cow starts at the origin and moves $x = 1$ unit to the right. If the target is not found, the cow returns to the origin and goes $y = 1$ unit to the left. If the target is not found, the cow comes back to the origin and repeats this procedure with $x = 2, 4, \ldots, 2^i, \ldots$ (on the right) and $y = 3, 9, \ldots, 3^i, \ldots$ (on the left) until the target is found (see the figure below).

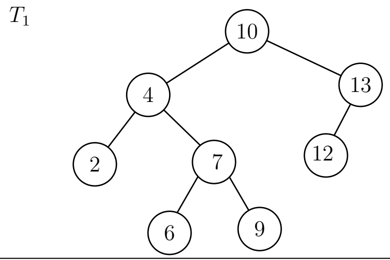
Indicate whether the algorithm is competitive or not. You need to show where the adversary places the target and what the cost of the algorithm and OPT is in that case.

Hint: You need to consider two cases, depending on the target being on the left or the right. We have $1+3+\ldots+3^k = \frac{3^{k+1}-1}{2}$.

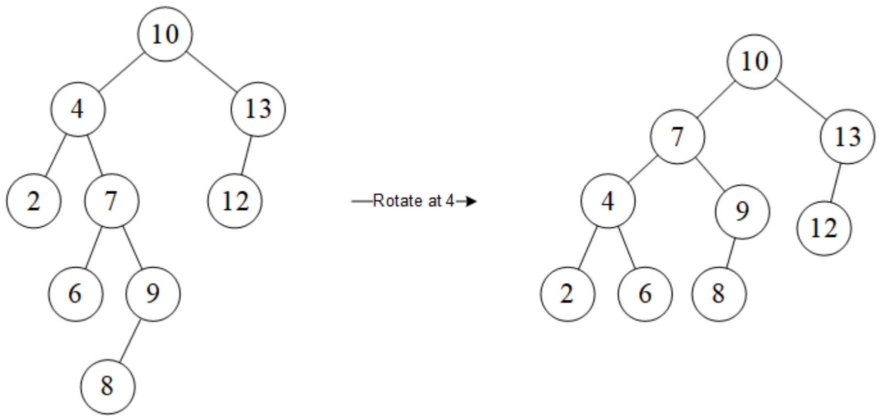


Answer: The adversary places the target either at distance $2^k + \epsilon$ on the right or $3^k + \epsilon$ on the left. In the former case, the cost of the algorithm will be $2(1 + 2 + \ldots + 2^k)) + 2^k + \epsilon \approx 5 \cdot 2^k$ for the moves on the right and $2(1 + 3 + \ldots + 3^k) \approx 3^{k+1}$ for the moves on the left. The ratio in this case will be $\frac{3^{k+1}+5\cdot 2^k}{2^k}$ which grows with k. In the latter case, the cost of the algorithm will be $2(1 + 3 + \ldots + 3^k) + 3^k + \epsilon \approx 4 \cdot 3^k$ for the moves on the left and $2(1 + 2 + \ldots + 2^{k+1}) \approx 2^{k+3}$ for the moves on the right. In this case, the ratio will be $\frac{4\cdot 3^k+2^{k+3}}{3^k}$, which is always bounded by a constant (converges to 4). So, the worst case happens when the adversary places the target at a distance $2^k + \epsilon$ on the right for large values of k; in this case, the competitive ratio grows with k, and the algorithm is not competitive.

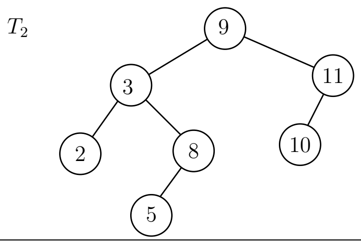
5. AVL Trees (6 marks)



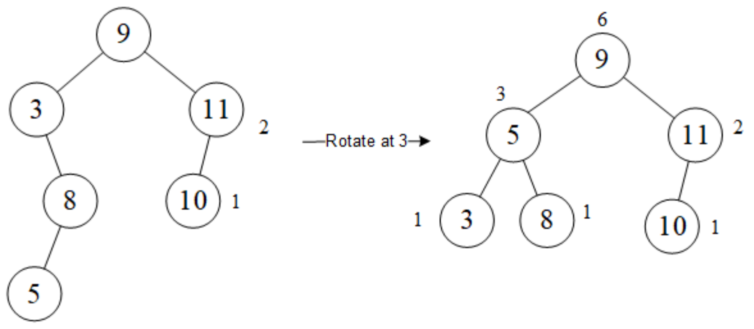
(a) Perform operation *insert*(8) on AVL tree T_1:
Draw the tree after each rotation performed.



Answer:



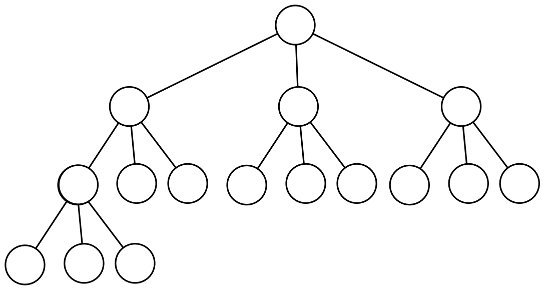
(b) Perform operation *delete*(2) on the AVL tree T_2:
Draw the tree after each rotation performed.



Answer:

6. Tree Heights (5 marks)

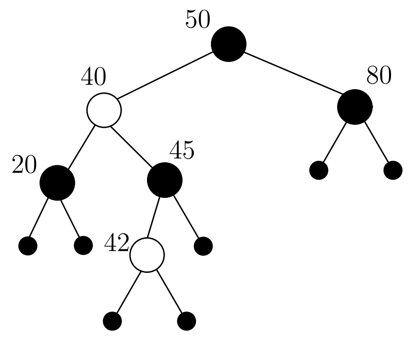
A balanced ternary tree is a tree in which each internal node has exactly three children, such that the difference between the heights of the subtrees rooted at children of each node is at most 1.
Prove that the height of a balanced ternary tree is $O(\log n)$. Show your work.



Answer: Let $n(h)$ be the number of nodes in any balanced ternary tree with height h. We can write $n(h) \geq n(h-1) + 2n(h-2) \geq 3n(h-2) \geq 9n(h-4) \geq 27n(h-6) \geq 3^i n(h-2i) \geq 3^{(h-1)/2} n(1) = 3^{(h-1)/2}$. Therefore $\log_3(n(h)) \geq (h-1)/2$, which gives $h \leq 2(\log(n(h))/\log 3) + 1 \in O(\log n)$.

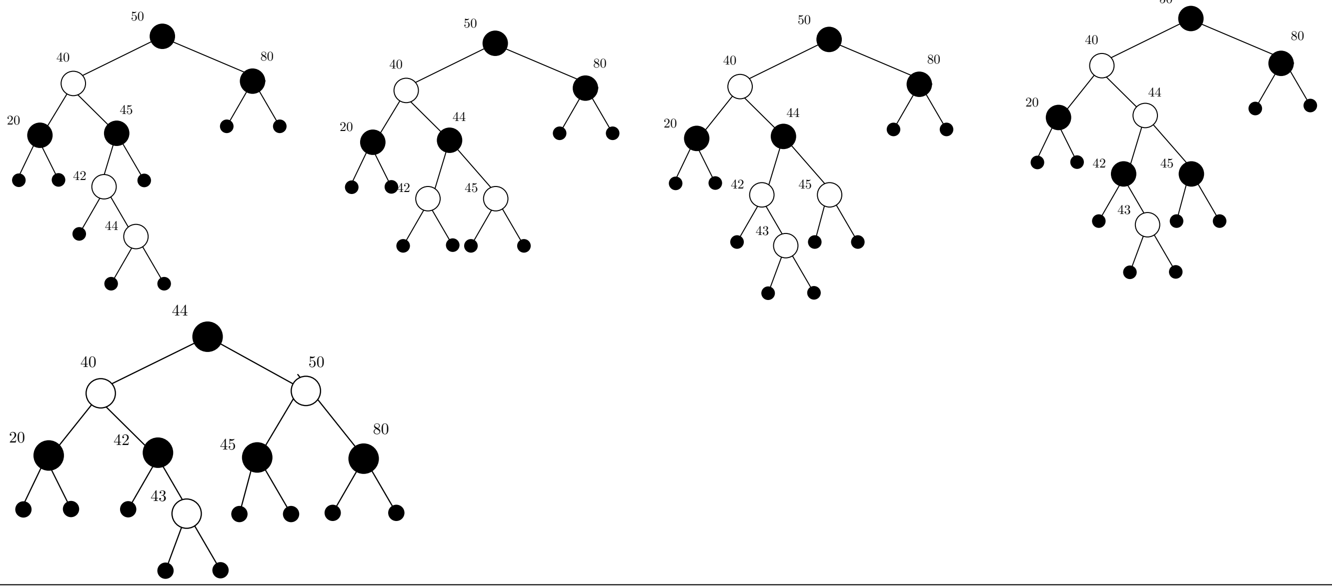
7. Red-Black Trees (8 marks)

Consider he following red-black tree T:



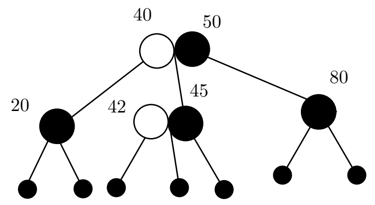
(a) Draw the tree after the operation $insert(44), insert(43)$ on it (in the same order). It suffices to draw the final tree.

Answer:



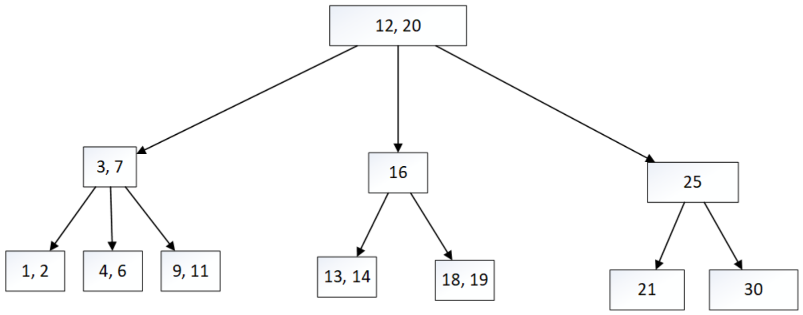
(b) Draw the b-tree associated with T (before the insert operation).

Answer:



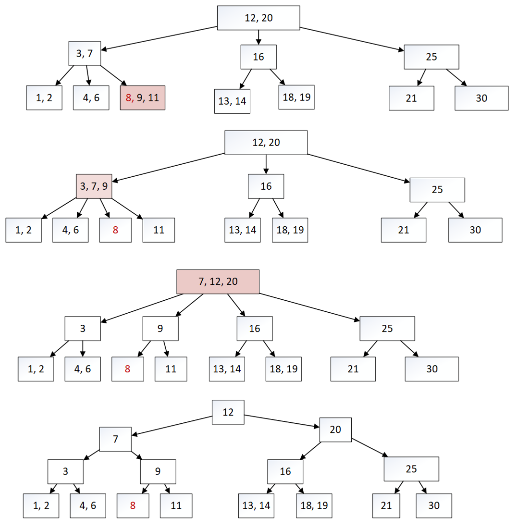
8. 2-3 Trees (8 marks)

Consider the following 2-3 tree.



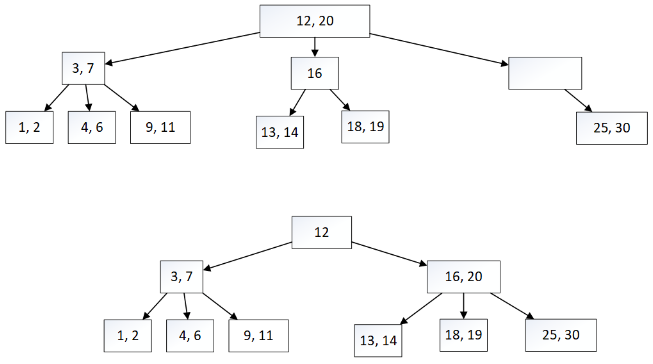
(a) Draw the tree when after we apply the operation *insert*(8) on it. It suffices to draw the final tree.

Answer:



(b) Draw the original tree after we apply the operation *delete*(21) on it. It suffices to draw the final tree.

Answer:



Use this blank page for your draft work.

Use this blank page for your draft work.

Use this blank page for your draft work.

Use this blank page for your draft work.