Jane Downer

CS 430 – Fall 2021
INTRODUCTION TO ALGORITHMS
HOMEWORK #4
DUE 23:59 THURSDAY, NOV. 4

Ethics: Any behavior on any homework or exam that could be considered copying or cheating will result in an immediate zero on the assignment for all parties involved and will be reported to academichonesty@iit.edu See the IIT Code of Academic Honesty, https://web.iit.edu/student-affairs/handbook/fine-print/code-academic-honesty

- Assignment Instruction
  - Team work is NOT allowed.
  - Submit your answers in PDF version to the Blackboard.
  - No late submission accepted.
  - All solutions should be explained.

!! Any unrecognized handwriting will cause ambiguity and result in a zero to your solutions!!

1. To build a Red-Black binary search tree, we start from an empty tree and a sorted array.

1a) (2 points) Show each red-black tree that results after successively inserting the keys 4 7 12 15 3 5 14 18 into an initially empty red-black tree. At the steps where a red-black tree rule is violated, explain how it is corrected to maintain properties.

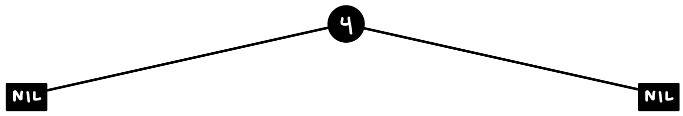
Work on scratch paper

1b) (3 points) Now delete these keys 18, 15, 7, 14 in order and show each resultant red-black tree after deleting each key. At the steps where a red-black tree rule is violated, explain how it is corrected to maintain properties.

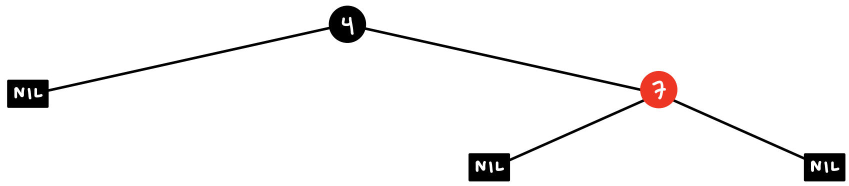
2. (3 points) Exercise 13.1-1 In the style of Figure 13.1(a), draw the complete binary search tree of height 3 on the keys {1, 2, ..., 15}. Add the NIL leaves and color the nodes in three different ways such that the black-heights of the resulting red-black trees are 2, 3, and 4. Black-height = 4 (this is the only possible coloring)

3. (2 points) What is the largest possible number of internal (key) nodes in a red-black tree with black-height k (measured from root)? What is the smallest possible number? Justify your claims.

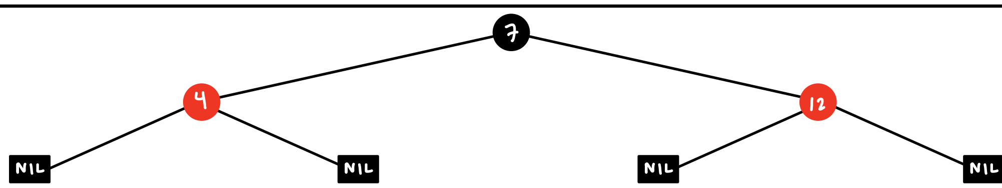
1A



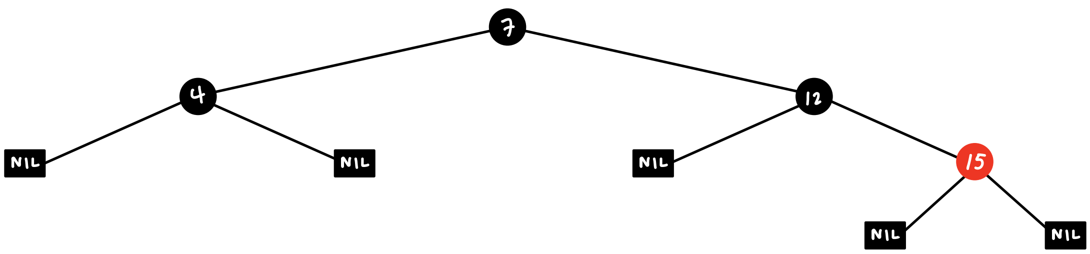
insert 4
[4 → black]



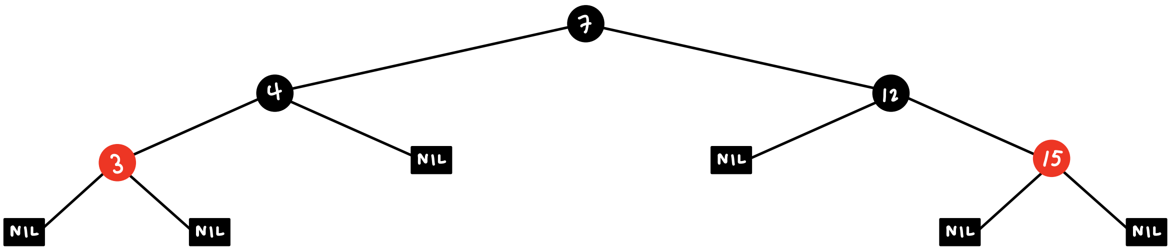
insert 7
[no transformations]



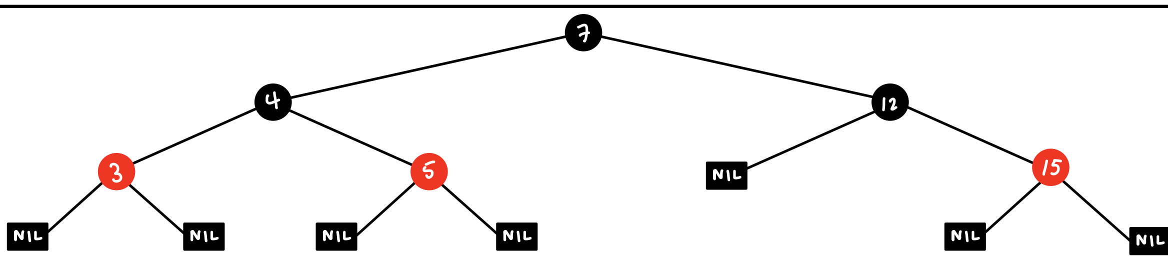
insert 12
[7 → black
4 → red
rotate left on 4]



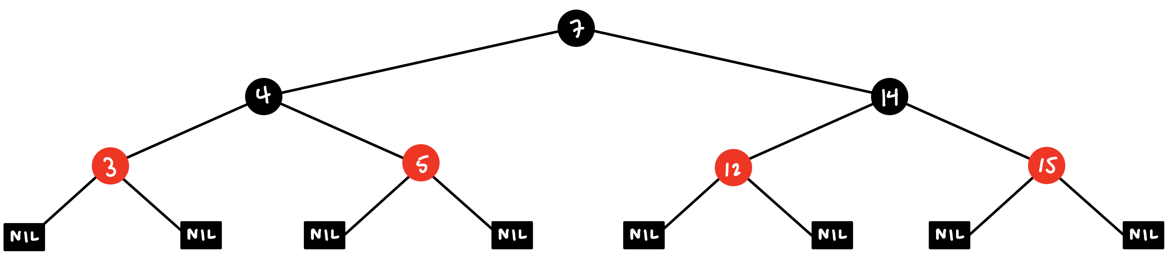
insert 15
[4 → black
12 → black]



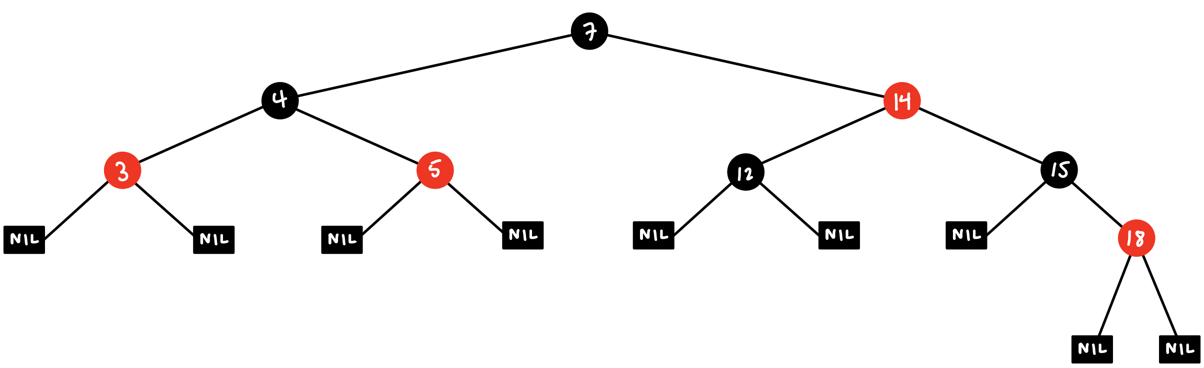
insert 3
[no transformations]



insert 5
[no transformations]



insert 14
[rotate right on 15
12 → red
14 → black
rotate left on 12]

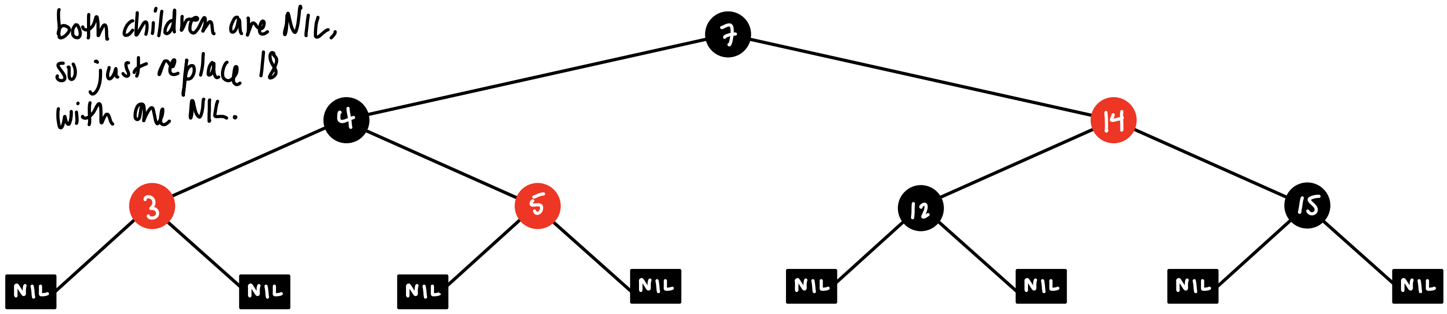


insert 18
[15 → black
12 → black
14 → red]

1B

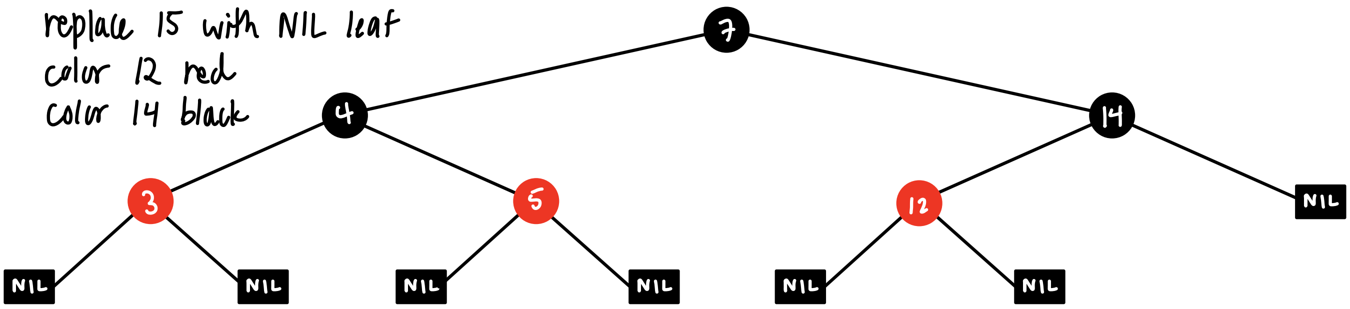
Delete 18:

both children are NIL,
so just replace 18
with one NIL.



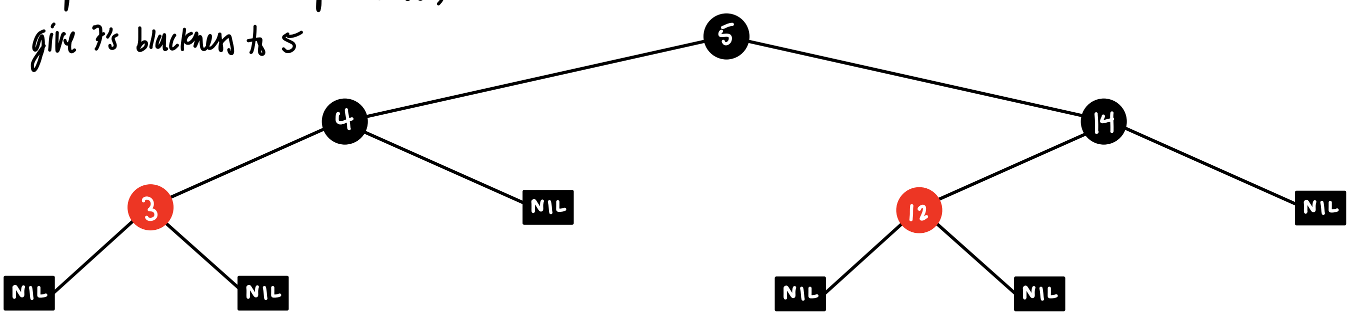
Delete 15:

replace 15 with NIL leaf
color 12 red
color 14 black



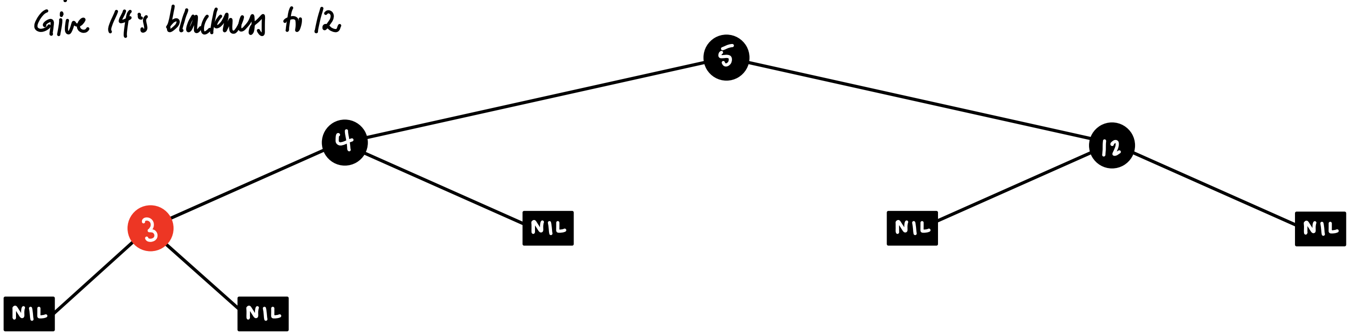
Delete 7:

replace 7 with its predecessor, 5
give 7's blackness to 5



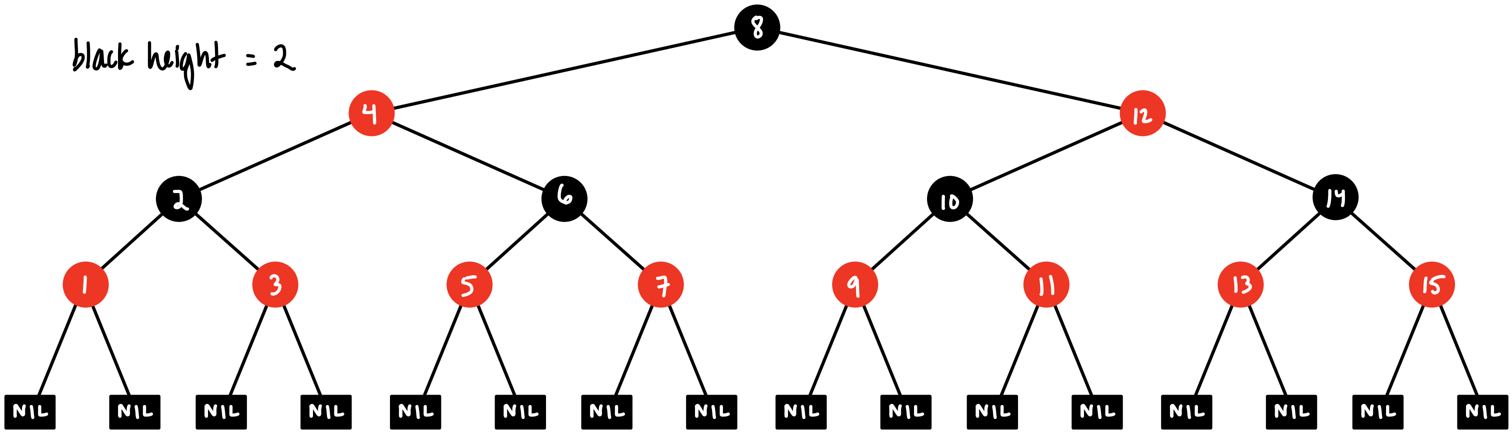
Delete 14:

Replace 14 with 12
Give 14's blackness to 12

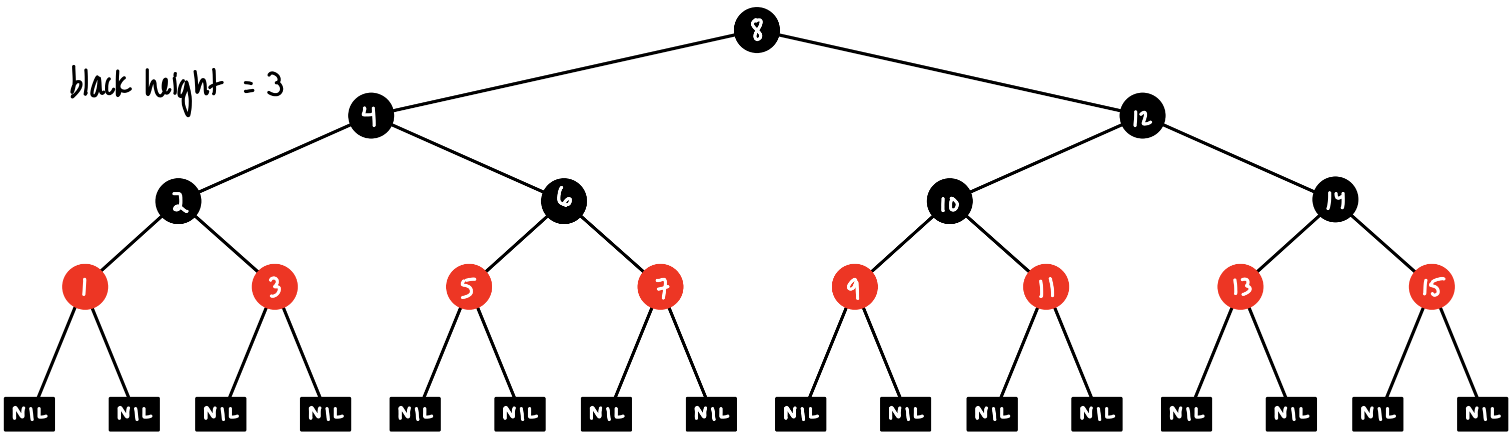


2

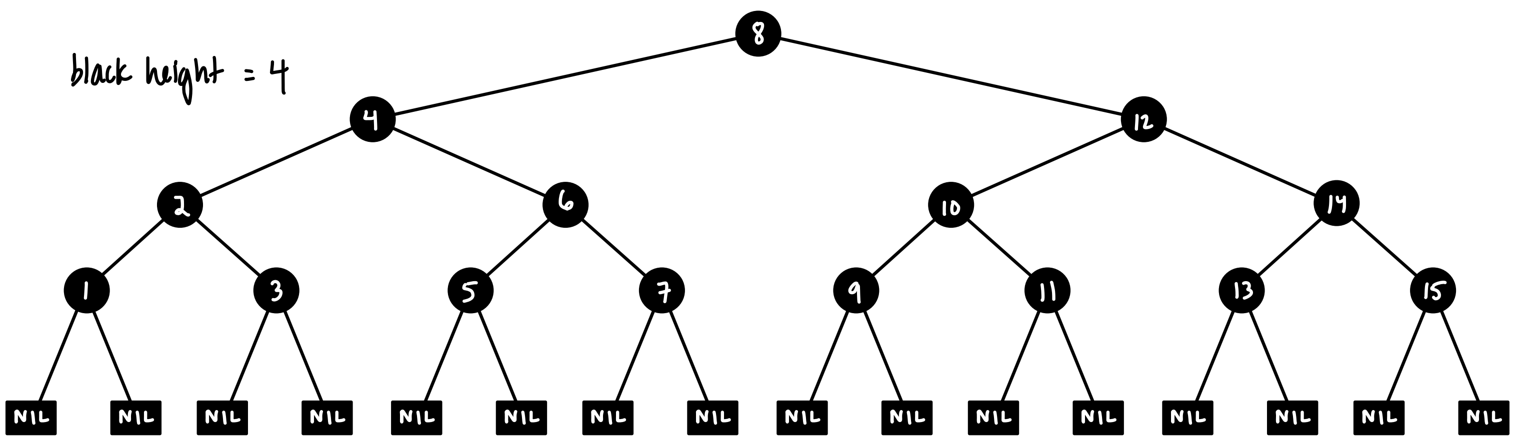
black height = 2



black height = 3



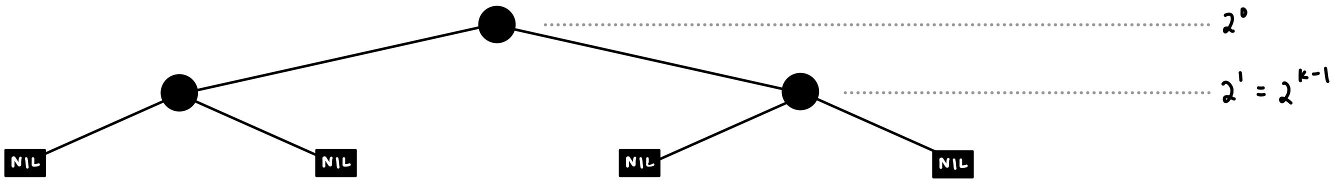
black height = 4



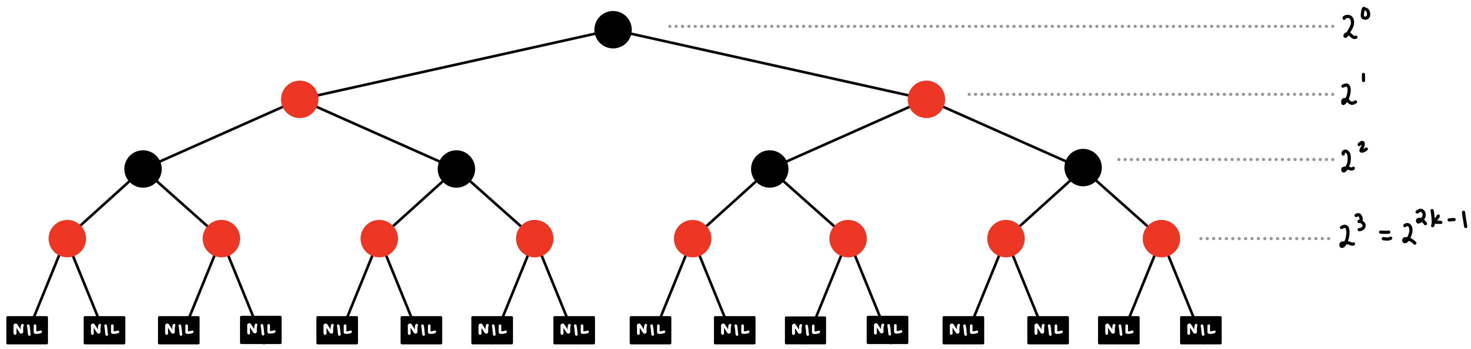
3

For a given black height k, the minimum number of internal nodes occurs in a full, all-black tree. The maximum occurs when each level of all-black nodes is followed by a single level of all-red nodes. This will also be a full tree.

Say $k = 2$. The minimum number of internal nodes occurs when:



The maximum number of internal nodes occurs when:



Generalizing this pattern, when black height is k...

minimum number of internal nodes $= \sum_{i=0}^{k-1} 2^i = \dfrac{2^{((k-1)-1)} - 1}{2-1} =$

maximum number of internal nodes $= \sum_{i=0}^{2k-1} 2^i = \dfrac{2^{((2k-1)-1)} - 1}{2-1} =$

number of internal nodes for RB tree with black height k:

$2^k - 1 \longleftarrow$ smallest possible

$2^{2k} - 1 \longleftarrow$ largest possible