CSDS 233 Midterm Summary Session

SI Leader: Jakob Danninger

3/2/2023

Disclosure: This is a supplement to class, not a replacement. This should not be your only study activity for exams, it should aid you in studying. I do not have access to the actual exam so questions here will differ from those on the exam.

Session Objectives:

1) Enhance understanding of content which includes
   a. Recursion
   b. Big O notation of functions (both recursive and iterative)
   c. Linked Lists (including using an iterator)
   d. Stacks, Queues, and circular Queues
   e. Binary Trees (and in order, post order, pre order)
   f. Binary Search Trees
   g. AVL Trees

Questions

1) What is the output of the function printFun(8), printFun(10), printFul(55)? Circle the base case. Box the recursive call

```
static void printFun(int test)
{
    if (test < 1)
        return;

    else {
        System.out.println(test);
        printFun(test/2);
        return;
    }
}
```

8 → 8, 4, 2, 1

10, → 10, 5, 2, 1

55 → 55, 27, 13, 6, 3, 1

O = worst case or worse

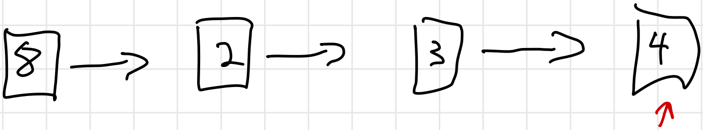
Θ = worst case

o = worse than worst case

Ω = better than worst case

Itterators :

↳ tool to acces an encapsclated List



iter.next() -> retrn 8

.next() -> return curr value and move itter

.hasNext()

2) Determine the big O of the following

```
public void example1 (int N) {
        for (int i = 0; i < N; i++) {
                System.out.println(”do something ”);
        }
        for (int i = 0; i < N; i++) {
                System.out.println(”do something ”);
        }
}
```

$\rightarrow O(n)$

$\rightarrow O(N)$

$\rightarrow O(2n) = O(n)$

$\Omega(\log n)$

$\Theta(n)$

$\Omega(1)$

$\Omega(\frac{1}{n})$

3) Determine the big O of the following

```
public void example2 (int N) {
        for (int i = 0; i < N; i++) {
                int j = N;
                while(j>0){
                        j = j/2;
                }
        }
}
```

$\rightarrow O(n)$

$O(\log n)$

$O(n \log n)$

4) Determine the big O of the following

```
public boolean example(int N){
        if(N < 1){
                return true;
        }
        N= N/2;
        example(N);
}
```

$O(\log(n))$

5) What is the big O of adding an item to the end of an array (that is not full) and to the end of a singly linked list?

↳ O(1)

↳ O(N)

6) Code an iterative method to get the number of occurrences of a given integer (target) in a linked list using an iterator

Iterator has the following methods

- getNext() → returns current value (integer) and moves iterator one forward
- hasNext() → returns whether the current node has a next node (boolean)

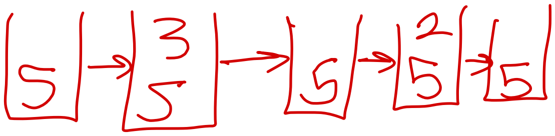
```
public int numOccur(LinkedList LL, int target){

    int numOccur = 0;

    LinkedList.Iterator iter = LL.iterator();  //this creates the iterator named iter
            while (iter.hasNext()){
                if (iter.getNext() == target){
                    numOccur ++
                }

}           }
        return(numOccur)
```

7) What are the main methods of stacks and queues (there are two for stack and two for queue)

enqueue   push

deque   pop

8) Draw the following: Stack: push 5, push 3, pop, push 2, pop



9) Draw the following: Queue: enqueue 5, enqueue 3, dequeue, enqueue 2, dequeue

5 → 53 → 3 → 32 → 2

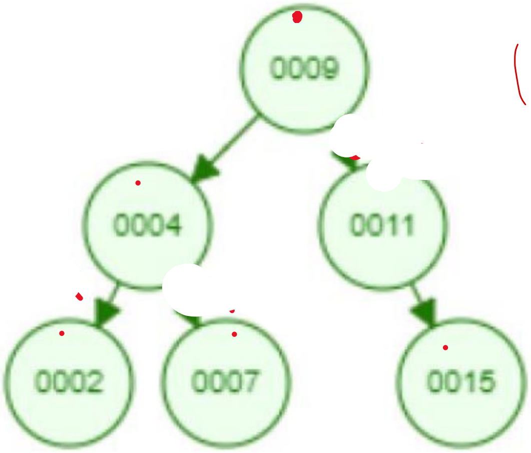
10) What is the difference between binary trees, binary search trees, and AVL trees

is 2 child

binary tree
left small
right big

balanced
BST

11) Write the in order, post order, and pre order



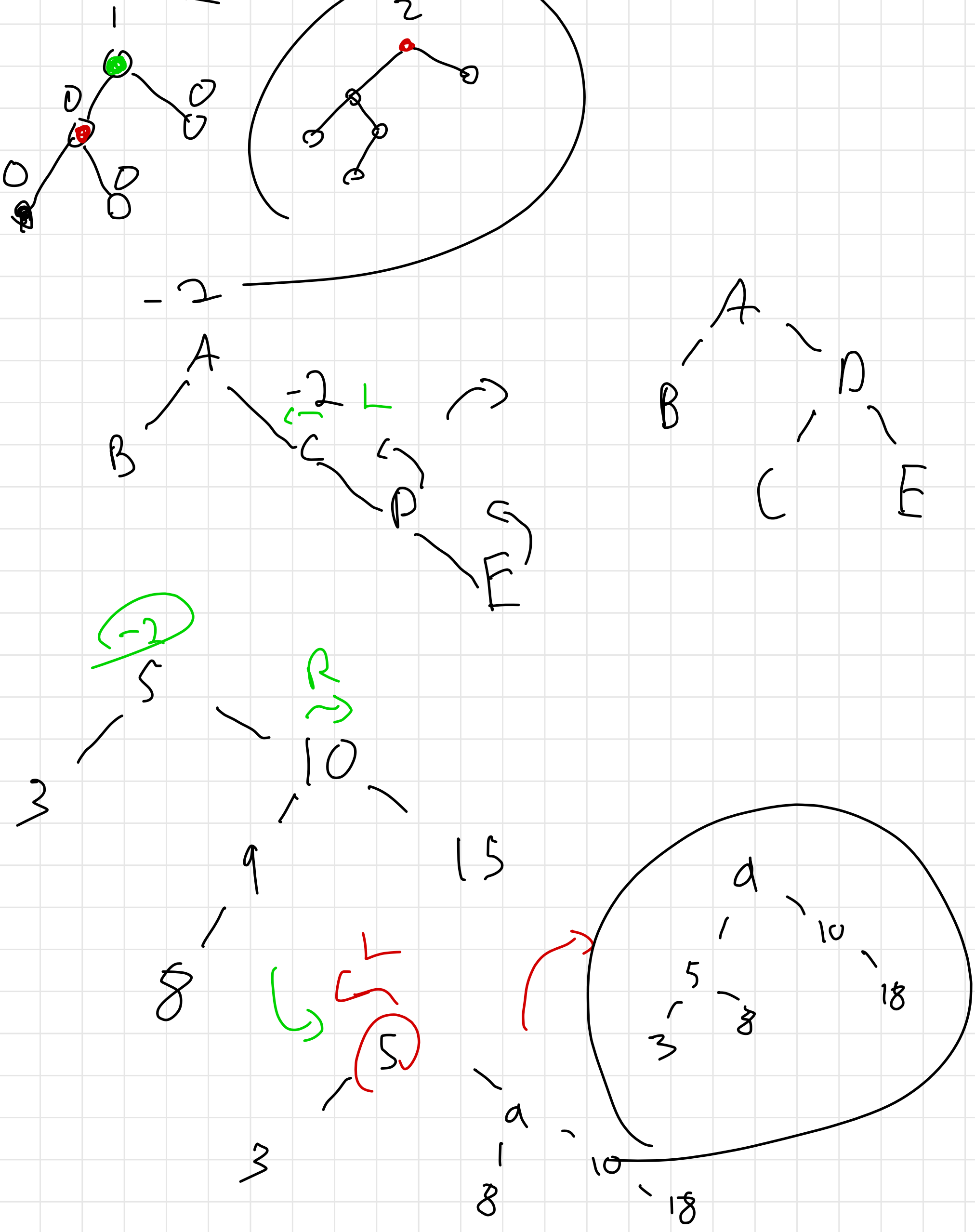
In order 2 4, 7, 9, 11, 15

pre 9, 4, 2, 7, 11, 15

post 2, 7, 4, 15, 11, 9

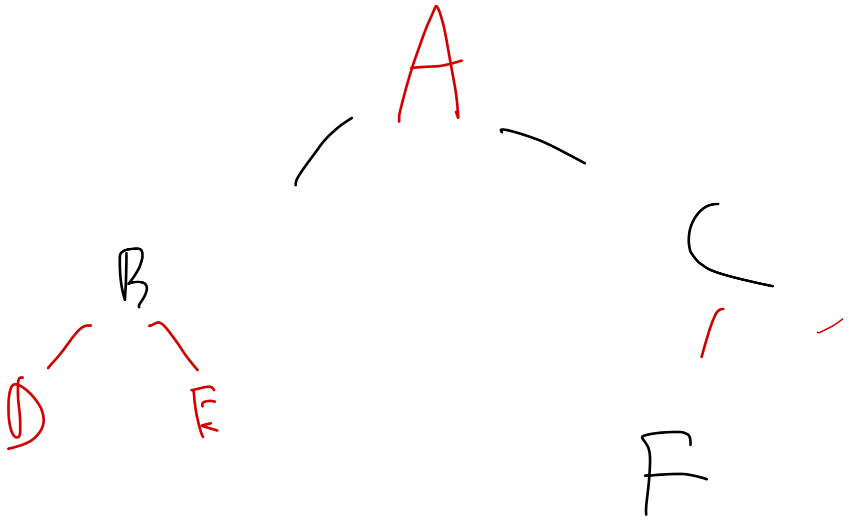
AVL Tree balanced BST

$-1 \leq \text{balance} \leq 1$



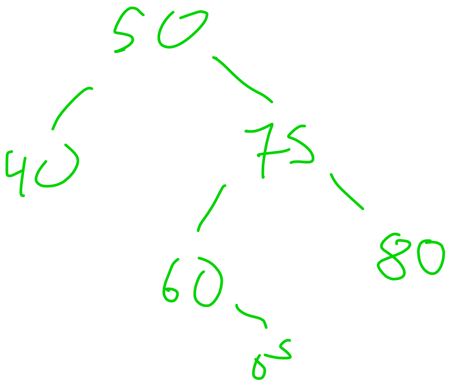
12) Create a binary tree from the following

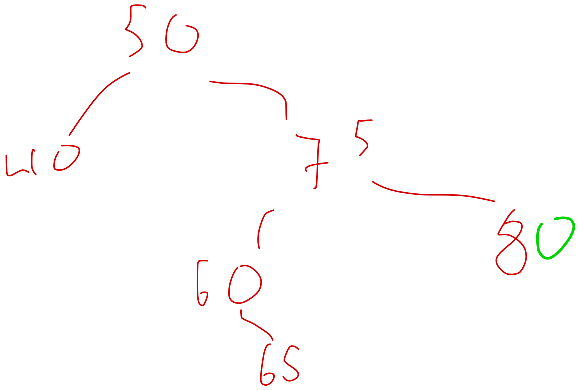
In Order: D, B, E, A, F, C   Pre-order: A, B, D, E, C, F



13) Create a Binary Search Tree from the following operations

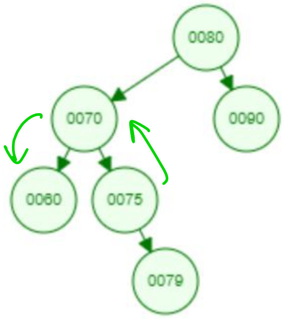
Add(50) add(40) add(75) add(60) add(65) add(80).   What is the height of the tree? Is this tree balanced?

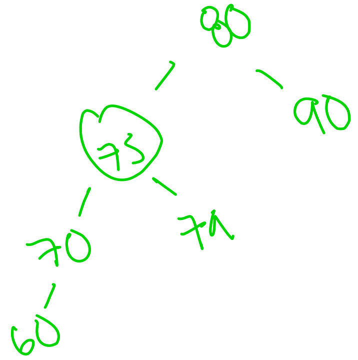




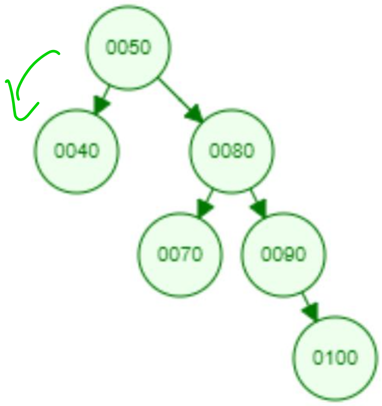
Now we will look at AVL Trees, Use the AVL cheat sheet to figure out what to rotate

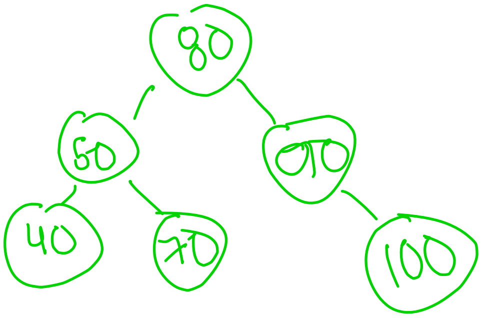
14) Rotate 70 left



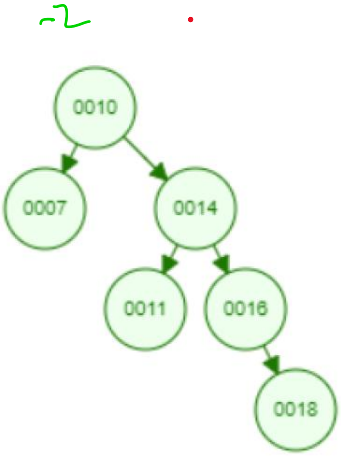


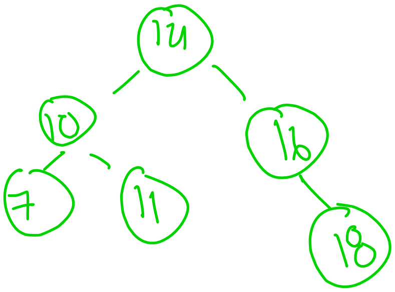
15) Rotate 50 left



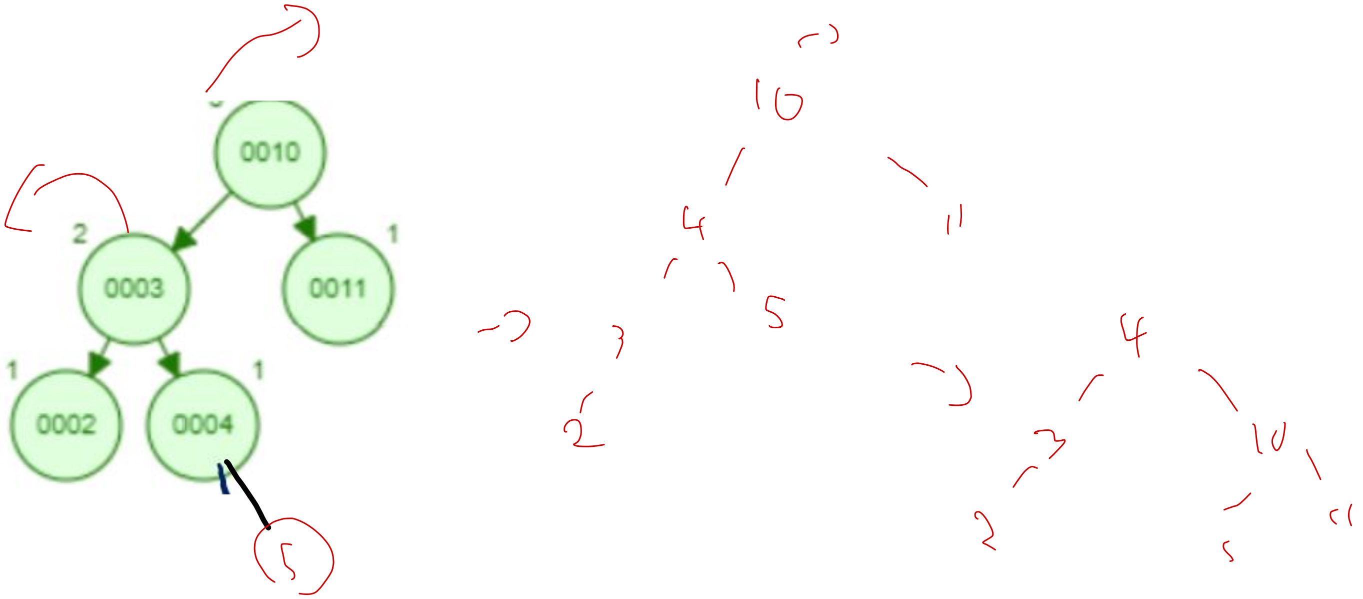


16) Where is the imbalance in the tree? Rotate the tree to balance it

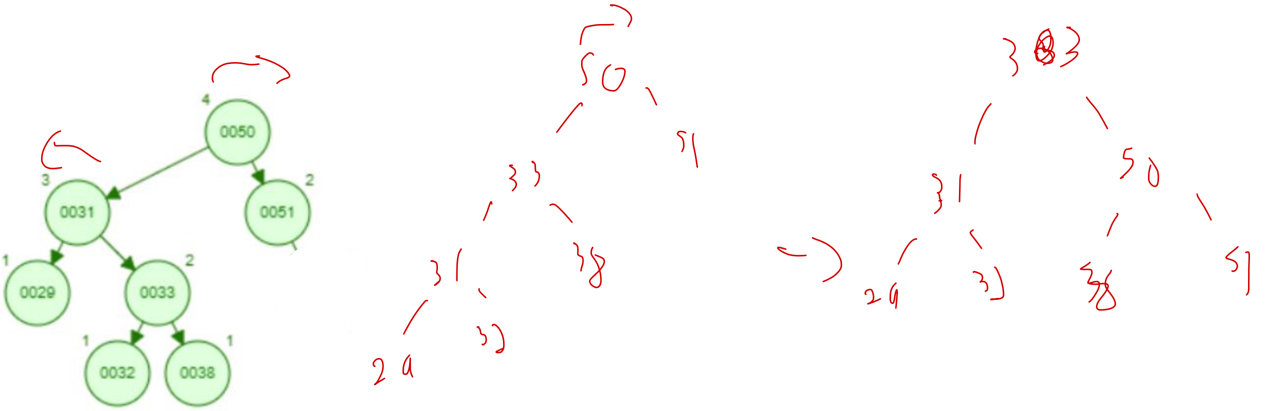




17) Add 5 to the following AVL tree

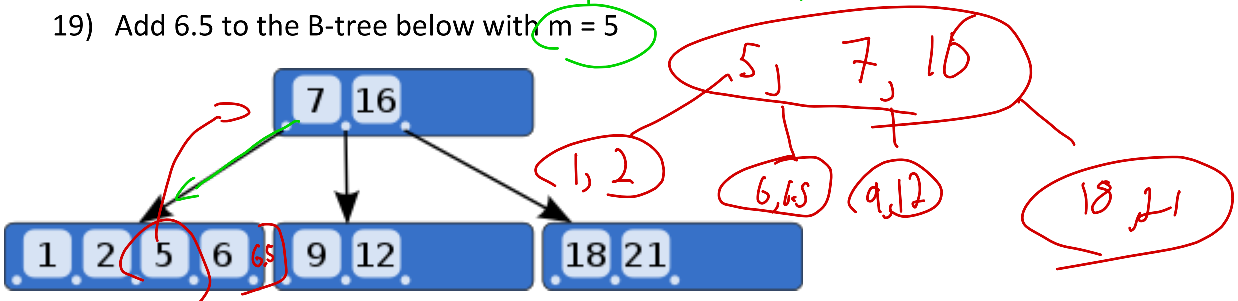


18) Balance this tree



19) Add 6.5 to the B-tree below with m = 5

max Index



20) Create a B tree using by adding the following numbers with m = 3

5, 3, 6, 7, 8, 1, 2,