B+ Trees Lecture April 10

* + - Minimal disk accesses
    - As broad as possible (height determines number of accesses)
    - Sean’s Rules:
    - When you split, (arbitrary thing that makes something nondeterministic deterministic – removes randomness)
    - New leaf nodes will always hold larger values (split to the right)
    - Leaf nodes have an array of integers (values)
    - and count (how many are actually used)
    - Push each thing in the array back by 1 until a lesser value is reached; insert there.
    - Last (largest) – holds the largest var.
    - All leafnodes/internal nodes have to be at least half full
    - If (uneven number in splitting)
    - New leaf (with larger values) is also then larger number of values – 4+1 = 2+3 not 3+2.
    - 1) New Node contains larger / more (if odd) values
    - 2) Look left, give to left if possible. Else, give right if possible. Else, split
    - If direct siblings are full, don’t keep looking – split.
    - Discussion
    - 1) Data items in leaves
    - 2) Internal nodes have m-1 keys (m in your program)
    - 3) The Ith key is the smallest key in the i+1th subtree
    - 4) The root is either a leaf or internal node which has 2+ children
    - 5) All nonLeaf (except root) are ½ full (ceiling m/2 and M children)
    - (m+1)/2
    - 6) All leaves are half full
    - Sean’s Rules:
      * When splitting, new node contains largest values
      * When adopting (passing off children), look left then right then split
    - Insertion:
      * Case 1: Room in leaf
      * Case 2: Adoption
      * Case 3: split