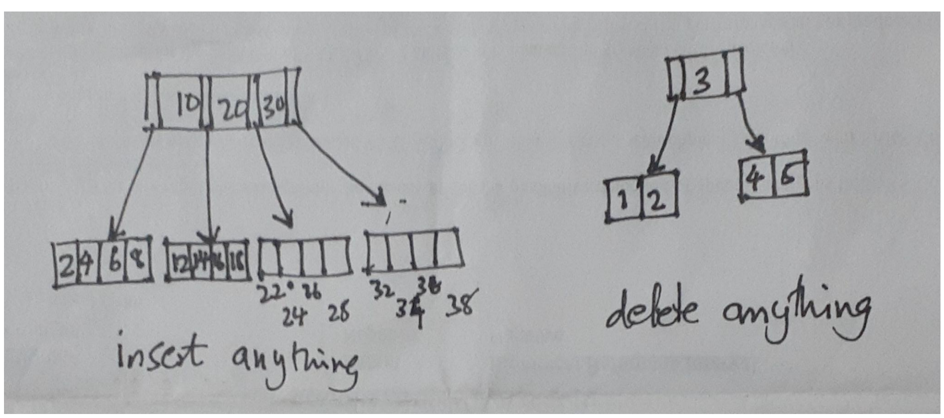
Tutorial 5 Solutions

1. **Note that in both 2-4 and B-trees, the heights of all the children of any node are the same. With this requirement, would 2-6 trees or 5-7 trees make sense. What about general A-B trees?**

   Let us look at an A-B tree, that is, an internal node, except the root can have between A and B children, both inclusive. Let us see what happens during:
   - Delete: A node x may then have A-1 children and become inconsistent. Then the first option is to check if one of its siblings' children may be transferred. That can be done if either has more than A children. If both have A, then there is no option but to merge x with a sibling y to a new node z. Then z will have A+A-1 =2A-1 children. Thus we must have 2A-1 <= B, for z to be consistent.
   - Insert: Suppose a node x becomes inconsistent due to its children going from B to B+1. It could shed one of its children with its siblings. But what if both have B children? Then we must split x into 2 nodes, each with at least A children. Thus we must have B-1>=2A, giving us the same condition. Thus 2-4 trees make sense but not 5-7.

2. **Show two examples of 2-4 trees where an insertion increases the levels by 1 and a deletion drops the level by 1. Can these be the same trees?**

   See the examples below. No, the two trees cannot be the same since they must differ at the root level. While a tree which causes the level to increase must have B children at its root, a tree which causes the level to drop will have 2 children. Note that this does not depend on A or B.



3. **Why is insertion in 2-4 and B-trees simpler than deletion?**

This is because an insert must happen at the leaf level. A delete can be at any internal node.This makes all the difference. If deletions were at the leaf nodes then delete would not be as complicated.

4. **In real-life B-trees, the number of children could be 100-1000. In this case, what is the fraction of values which are stored at the leaf level of the total number of values?**
   For an A-B tree of L levels, the minimum number of keys are (A-1)+(A-1)*A + (A-1)*A^{L-1) +A^L.. The maximum number is the same replaced by B. Thus, thus the number of internal keys are (B-1)(B^L-1)/(B-1)=B^L-1 which is of the same order as the last level! So really, even for large Bs, storing just at the leaves would be quite wasteful.

5. **Design a structure where the values are stored only at the leaf level. Only end-markers are stored in the intermediate levels. Would insertion and deletion be easier in such "simple-B-trees"?**

   This is not difficult. For any internal node, with say B children, we store the numbers [(L1,U1)(L2,U2)...(L_B,U_B)] where (Li,Ui) is the lowest and the highest key values stored under that child. It is then easy to delete since it happens at the leaf level.