2-3 Tree (B – Tree)

1. (20 points)

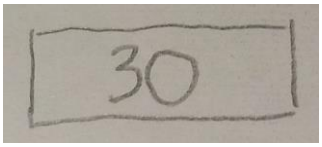
Insert: 30, 50, 80, 20, 40, 100, 25, 35

Search: 80, 100, 25

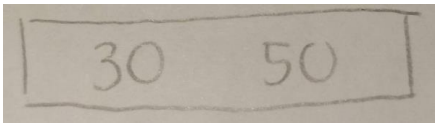
Delete: 50, 20, 100

Insertion

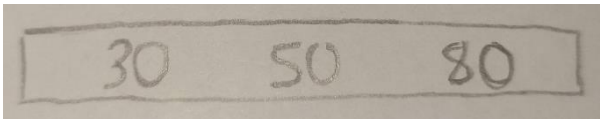
1. 30 becomes the root because there are no other nodes



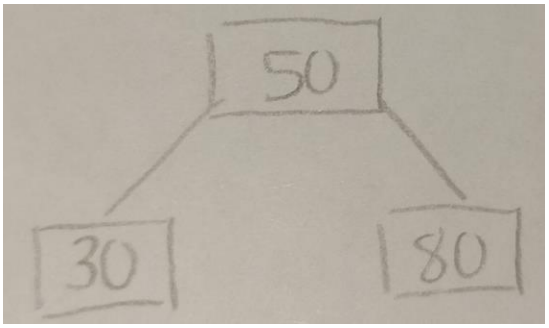
2. 50 goes to the right of 30 (because 50 > 30) and becomes the 2nd key



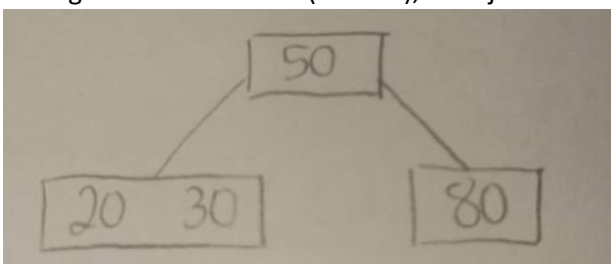
3. 80 goes to the right of 50 (because 80 > 50) and becomes the 3rd key, however the maximum number of keys has been surpassed, initiate split



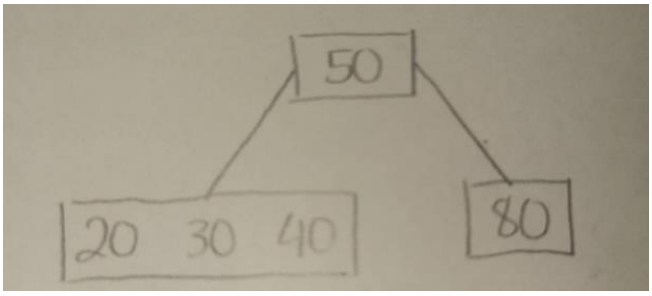
4. To split we take the median of the keys(30, 50, 80) which is 50, to become the parent of the remaining keys



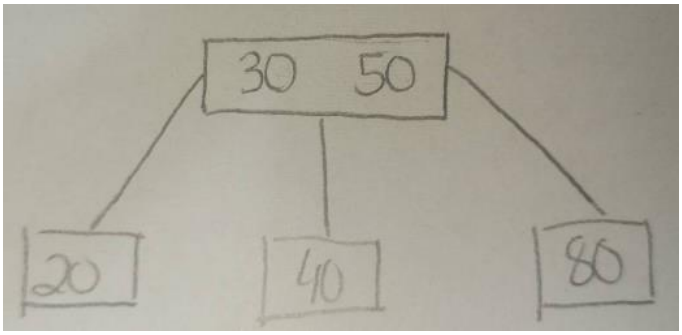
5. 20 goes to left subtree (20 < 50), then joins 30 on it's left side (20 < 30), becoming the 2nd key



6. 40 goes to left subtree (40 < 50), then joins 30 on it's right side (40 > 30), becoming the 3rd key, the maximum number of keys has been surpassed, initiate split

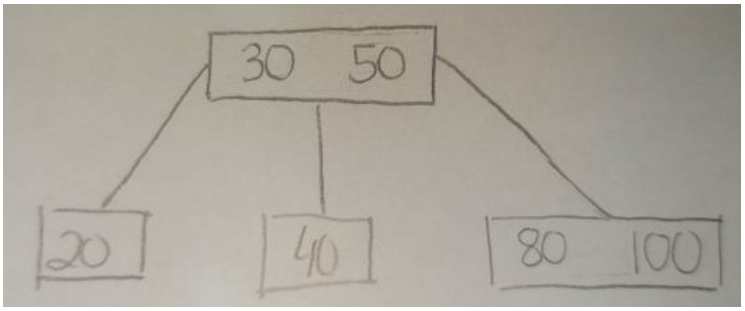


7. 30 is the median of the three keys (20, 30, 40), so 30 joins 50 on it's left side as a key in the root of the tree

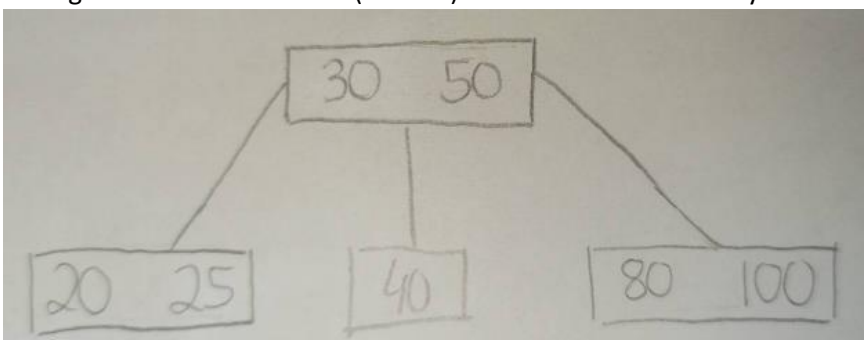


(Note : Because 20 is < 30, 20 becomes a leaf on the left subtree. Because 40 > 30 and 40 < 50, 40 becomes a leaf on the midle subtree)

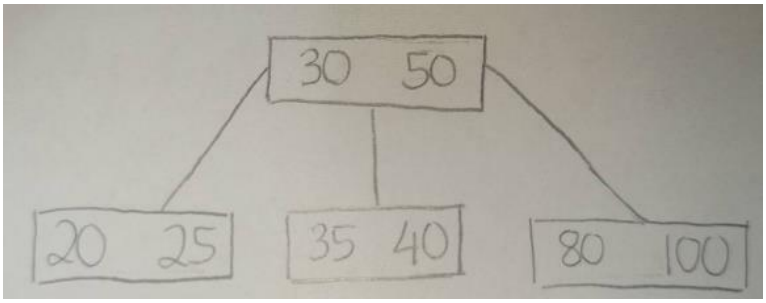
8. 100 goes to the right subtree (100 > 50) becomes the 2nd key on 80's right (100 > 80)



9. 25 goes to the left subtree (25 < 30) and becomes the 2nd key on 20's right (25 > 20)

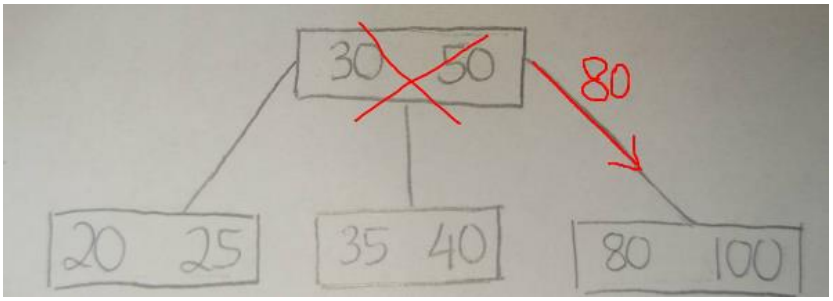


10. 35 goes to the middle subtree (35 > 30 && 35 < 50) and becomes the 2nd key on 40's left (35 < 40)

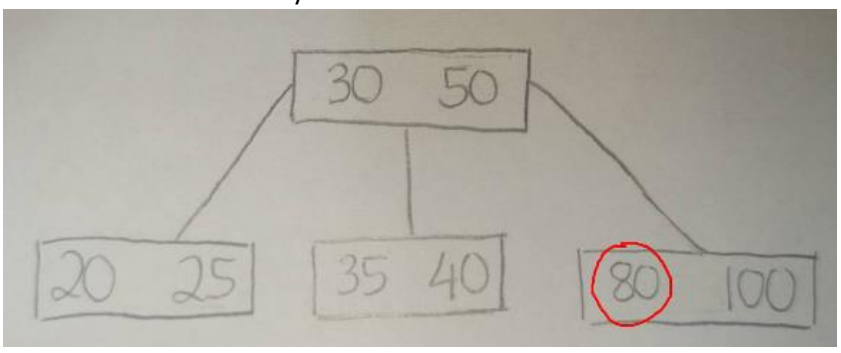


Searching

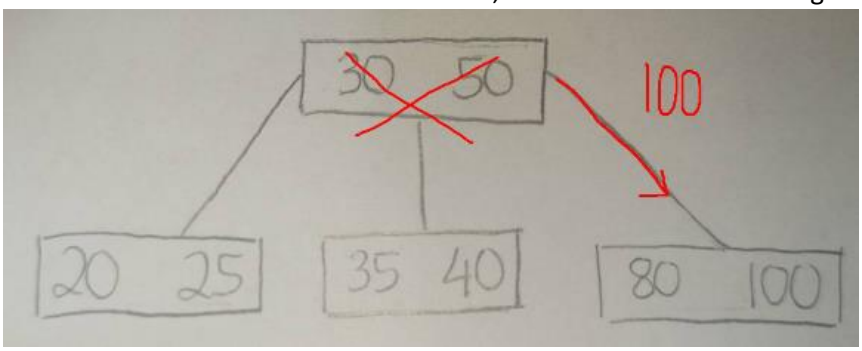
1. Because the root does not contain 80, search continues to the right subtree (80 > 50)



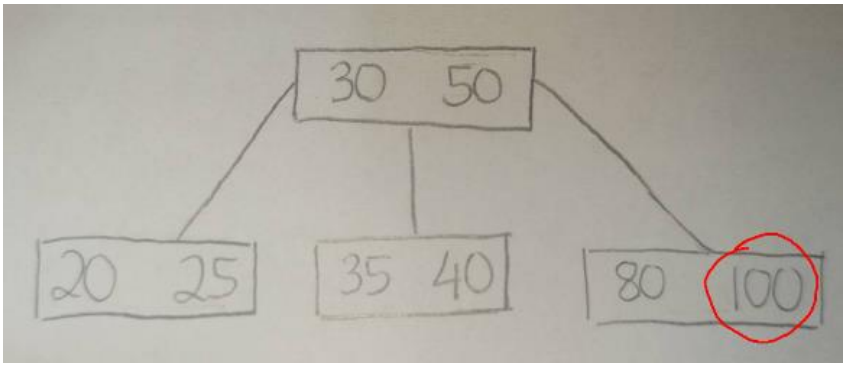
2. Found 80 as the 1st key



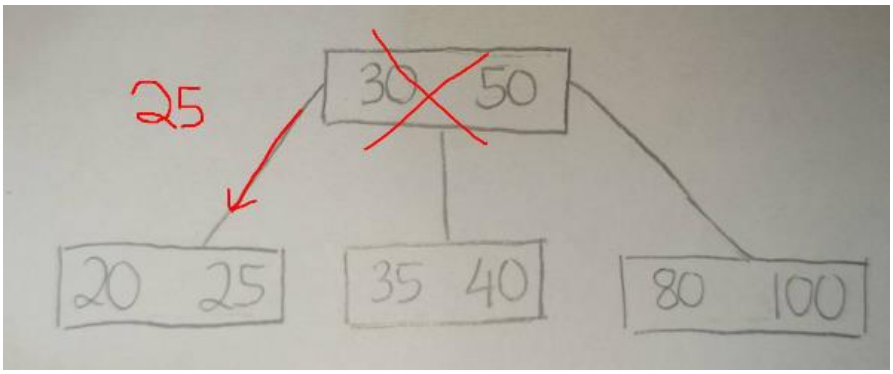
3. Because the root does not contain 100, search continues to the right subtree (100 > 50)



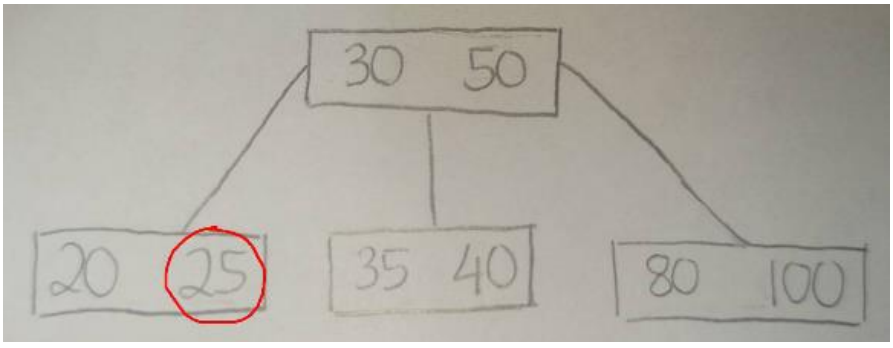
4. Found 100 as the 2nd key



5. Because the root does not contain 25, search continues to the left subtree (25 < 30)

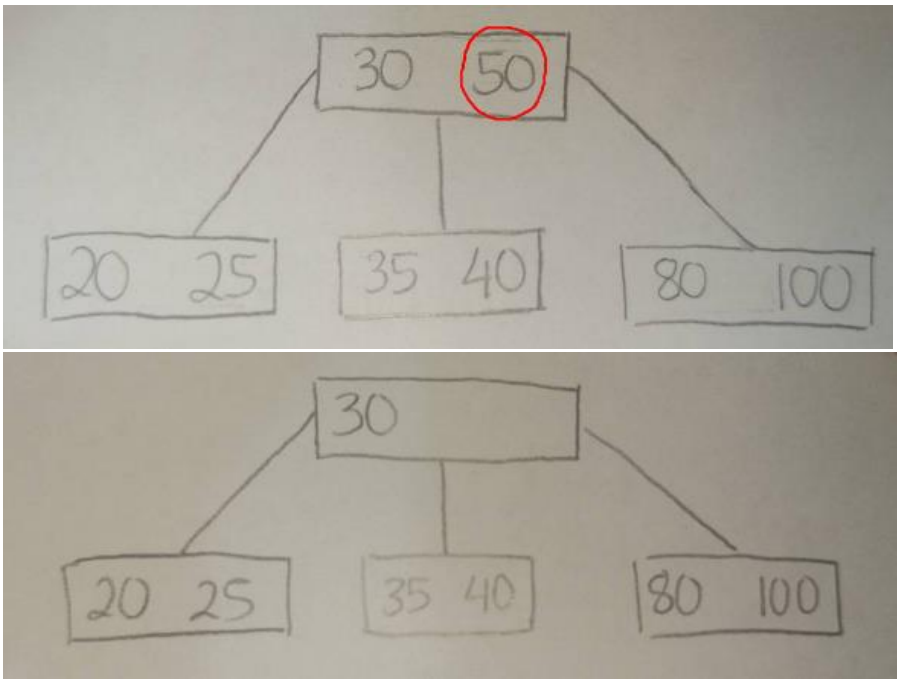


6. Found 25 as the 1st key

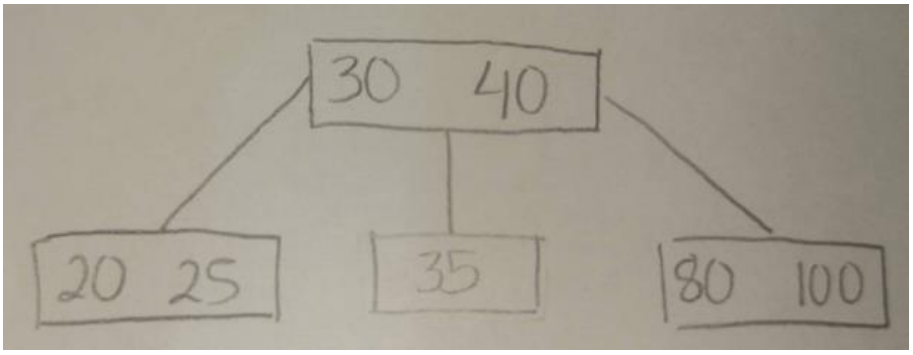


Delete

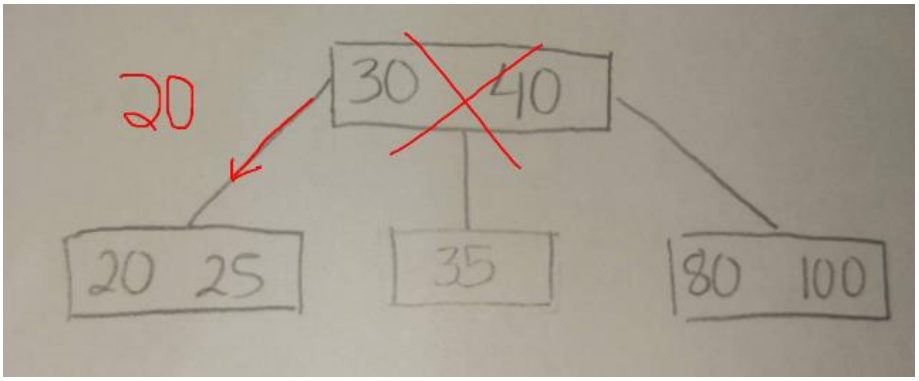
1. Because 50 has been found in the root, deleted 50



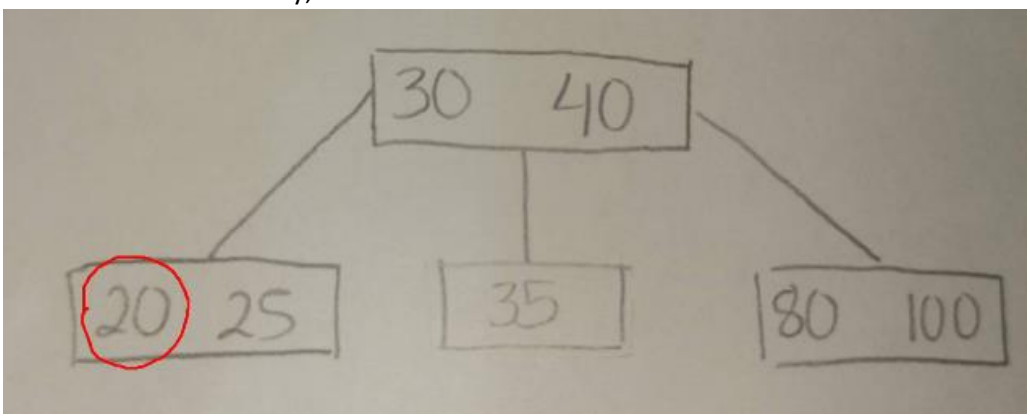
2. Because the root has an empty space, the largest value from the median (middle subtree) will move up to replace 50, in this case it is 40

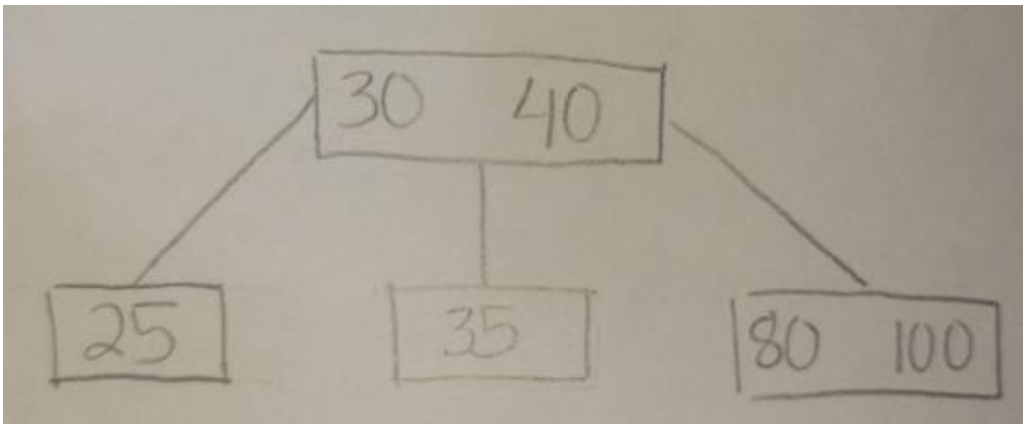


3. Because the root does not contain 20, search continues to the left subtree (20 < 30)

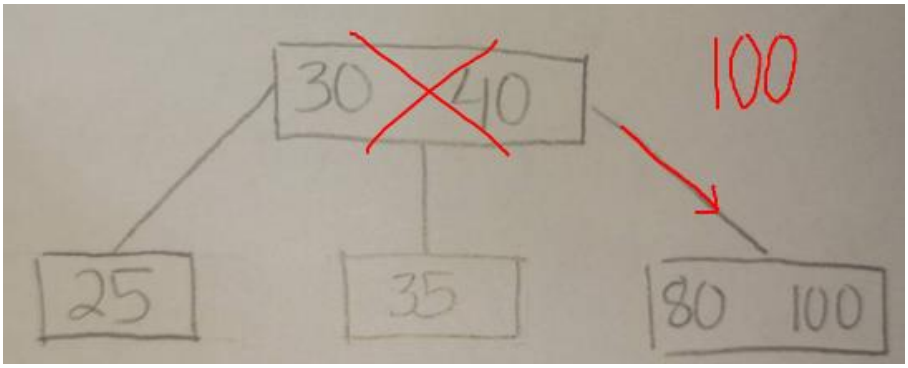


4. Found 20 as the 1st key, deleted





5. Because the root does not contain 100, search continues to the left subtree (100 > 50)



6. Found 100 as the 2nd key, deleted