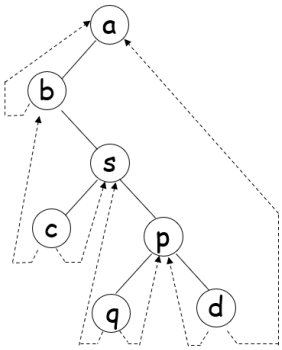
Department of Computer Science and Engineering
National Sun Yat-sen University
Data Structures - Final Exam., Dec. 19, 2022

1. Suppose that the preorder of a binary tree is *ABC*. Please give all possible postorder sequences with preorder *ABC*. (10%)

2. The following shows a threaded binary tree, where each dotted line is a thread pointer. Suppose that a new node *r* will be inserted as the right child of *s*. Please draw the threaded binary after *r* has been inserted. (10%)



3. Suppose that there are ten elements, whose hashing function is given in the left table. The directoryless dynamic hash table is used to insert the above elements, where each bucket has two slots. Initially, suppose that some elements have been inserted into the hash table (shown in the right table).

k	*h*(*k*)	*k*	*h*(*k*)
A0	100 000	B5	101 101
A1	100 001	C1	110 001
B0	101 000	C2	110 010
B1	101 001	C3	110 011
B4	101 100	C5	110 101

00	B4, A0
01	A1,B5
10	C2, -
11	C3, -

   (a) Please show the hash table after the next element C5 is inserted. (5%)
   (b) Please show the hash table after the next elements C5 and C1 are inserted. (5%)

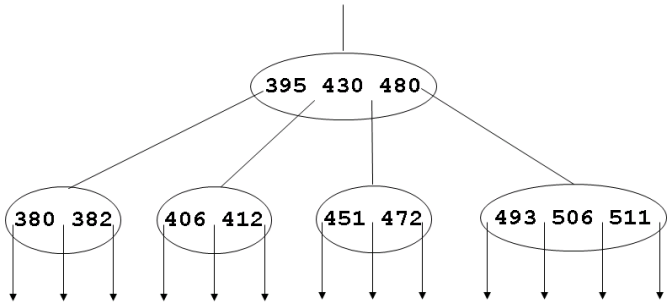
4. (a) Please describe the three properties for the colors of the nodes in a red-black tree (5%)
   (b) Assume that the initial red-black tree is empty. Please draw the tree after the numbers 25, 15, 10, 50, 70 are inserted into the tree sequentially, and indicate the node colors (R for red, and B for black). (5%)
   (c) Please draw the tree after the numbers 90, 6 are inserted into the above red-black tree obtained in (b), and indicate the node colors. (5%)

5. Explain each of the following terms. (16%)
   (a) priority queue
   (b) external sorting
   (c) linear probing in hashing

(d) splay tree

6. (a) For a B-tree of order *m* (*m*-way), how many children are there for each non-root node? How many children are there for the root node? (6%)
   (a) What is difference between a B-tree and B+-tree? (4%)

7. Please draw the tree after 485 and 496 are inserted into the following B tree of order 5. (5%)



8. Given a searched value *x* and a difference value *n*, we want to find the element *y* such that $| x - y | \le n$ (absolute value of the difference). Write a recursive C/C++ function to count the number of all such *y* stored in a *binary tree*, where each node stores one integer element (not a binary search tree). (12%)

```
class TreeNode {
   int data;      // the number stored in the node
   TreeNode *leftChild, *rightChild;
};
int Count(...) or void Count(...) // x: searched value;    n: difference value
{
```

Please write the body of the function.

```
} // end of Count( )
```

9. Write a recursive C++ function to perform *Quick Sort* with nondecreasing order. You can directly call swap(*x*,*y*) to exchange the values of *x* and *y*. (12%)

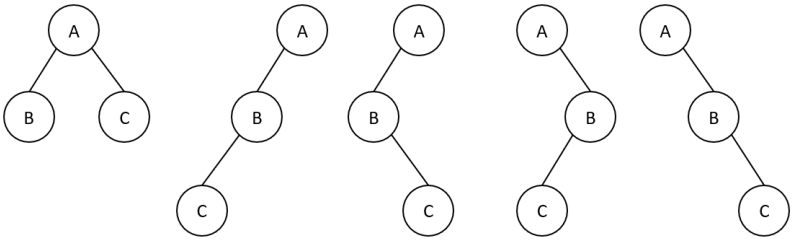
```
int a[100]; // global array for storing the elements to be sorted.
            // The sorted elements are still stored in a [ ].
void swap(int &c, int &d)    // exchange the values of c and d
{ int temp;
  temp=c;    c=d;    d=temp;
}
void qsort(int left, int right) // a recursive function
// left, right: the left and right indexes of a[ ] to be sorted
{
```

Please write the body of the function.

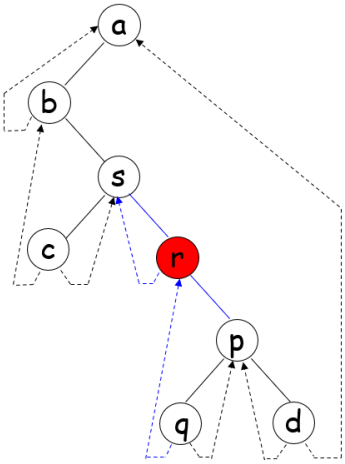
```
} // end of qsort ( )
```

Answer:

1. BCA, CBA



2. threaded binary tree



3. hash table

(a)

Overflow bucket

000	A0, -
001	A1, B5 → C5
010	C2, -
011	C3, -
100	B4, -

(b)

000	A0, -
001	A1, C1
010	C2, -
011	C3, -
100	B4, -
101	B5, C5

4. Red-black tree

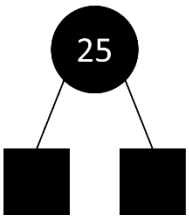
(a) RB1: The root and all external nodes are black.

RB2: No root-to-external-node path has two consecutive red nodes.

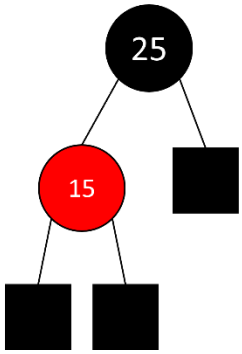
RB3: All root-to-external-node paths have the same number of black nodes.

(b)

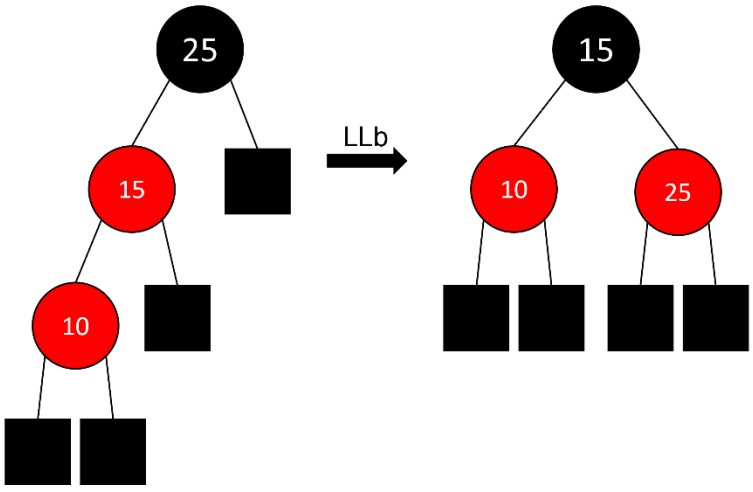
1. Insert 25



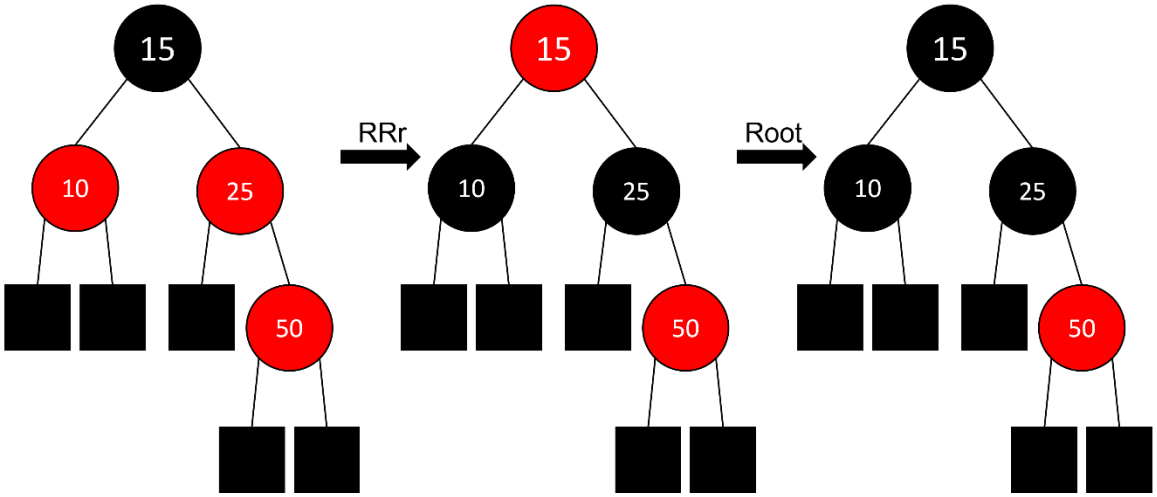
2. Insert 15



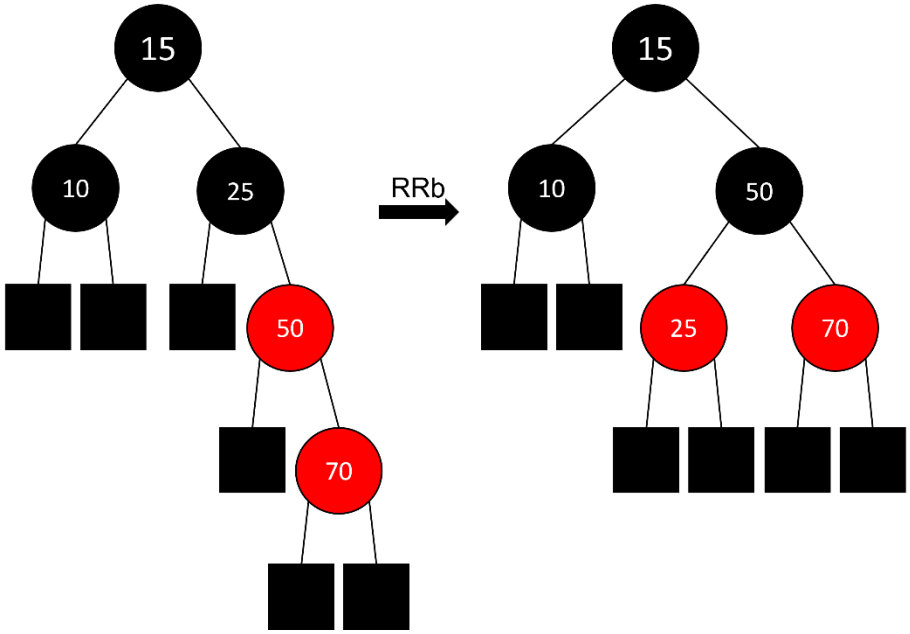
3. Insert 10



4. Insert 50

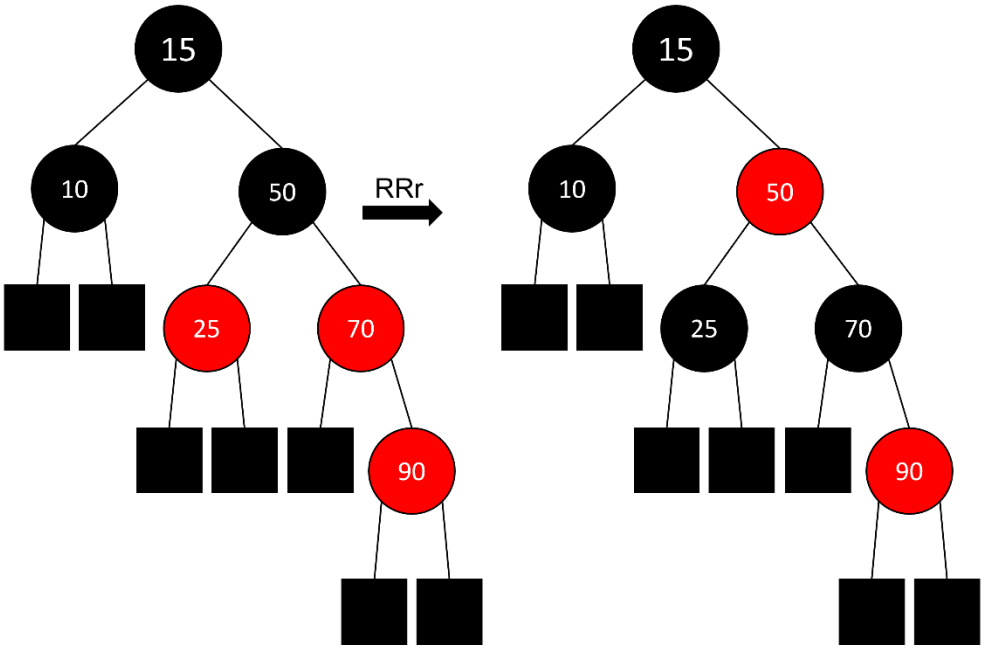


5. Insert 70

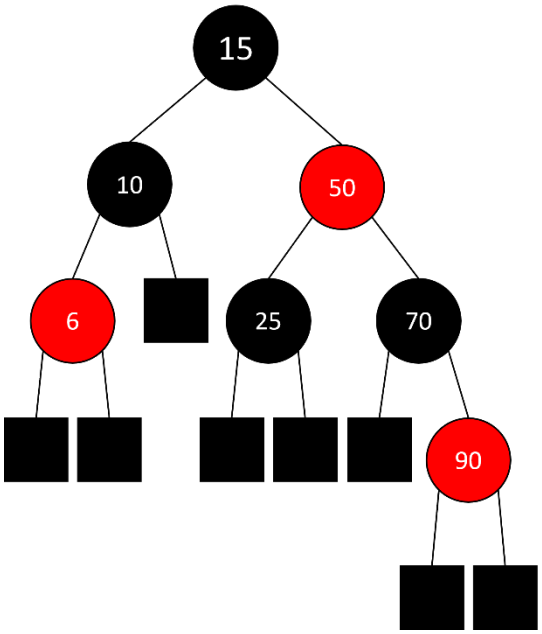


(c)

6. Insert 90



7. Insert 6



5.

(a) priority queue 是一種資料結構， 存放在 priority queue 的元素，其順序是根據元素的值，而非依據元素到達的先後順序。一般而言，可在 priority queue 中，進行 insertion, min(or max) finding, min (or max) deletion 的操作。

(b) 需要 sort 的資料量太大，在 sort 過程，無法將資料一次全部讀入主記憶體，而需要將資料存放於輔助記憶體。

(c) 在進行 hashing 時，若發生 overflow 的情形，則按順序找尋下一個可存入資料的 bucket。

(d) 沒有存放 no balanced information 的 binary search tree。 進行搜尋時，會將

被搜尋的 node 調整為 root，以方便往後可以較快搜尋到該 node。

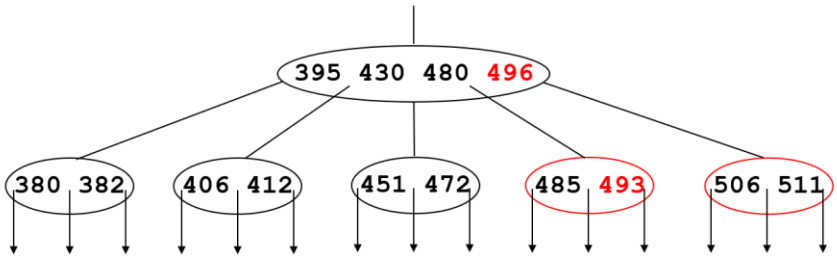
6. B-tree

(a) # of children of non-root node: $\left\lceil \frac{m}{2} \right\rceil \leq$ # children of non-root $\leq m$

# of children of root node: $2 \leq$ # children of root $\leq m$

(b) B tree 的每一個 node(包含 internal node 與 leaf node)均有儲存資料，但是 B+tree 的資料僅儲存於 leaf node(internal node 沒有儲存資料)。此外，B+tree 的 leaf node 間以 linked list 串接在一起。

7. B-tree



8.

```
class TreeNode {
   int data;       // the number stored in the node
   TreeNode *left, *right;
};
int Count(TreeNode *root, int x, int n)
{
int leftC, rightC;
if( root == 0 )
      return 0;      // Return 0 if the binary tree is empty.
leftC = Count(root->left );
rightC = Count(root->right );
if ( abs(root->data, x) <=n )       // the root is an answer
      return leftC + rightC +1;
else              // the root is not an answer
      return leftC + rightC
} // end of Count ( )
```

9.

```
void qsort(int left, int right){
    if (left < ritght) {
        int i = left, j = right + 1, pivot = a[left];
        do{
            do i++; while (a[i] < pivot);
            do j--; while (a[j] > pivot);
```

```
            if (i<j) swap(a[i], a[j]);
        } while (i < j);
        swap(a[left], a[j]);
        qsort(left, j–1);
        qsort(j+1, right);
    }
}
```