6.1 Presorting

Element Uniqueness

```
//Solves the element uniqueness problem by sorting the array first
//Input: An array A[0..n − 1] of orderable elements
//Output: Returns "true" if A has no equal elements, "false" otherwise
sort the array A
for i ← 0 to n − 2 do
    if A[i] = A[i + 1] return false
return true
```

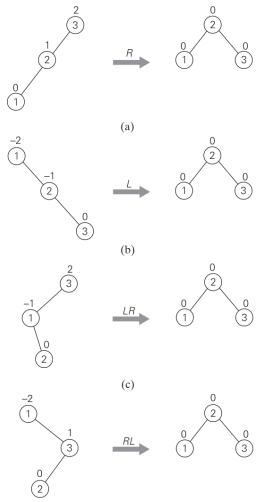
Computing mode

```
ALGORITHM PresortMode(A[0..n − 1])
    //Computes the mode of an array by sorting it first
    //Input: An array A[0..n − 1] of orderable elements
    //Output: The array's mode
    sort the array A
    i ← 0                          //current run begins at position i
    modefrequency ← 0              //highest frequency seen so far
    while i ≤ n − 1 do
        runlength ← 1;  runvalue ← A[i]
        while i + runlength ≤ n − 1 and A[i + runlength] = runvalue
            runlength ← runlength + 1
        if runlength > modefrequency
            modefrequency ← runlength;  modevalue ← runvalue
        i ← i + runlength
    return modevalue
```

Searching Problem

AVL Trees:

Rotation



Right rotation

Left rotation

Left right rotation

Right left rotation.

2-3 Trees:

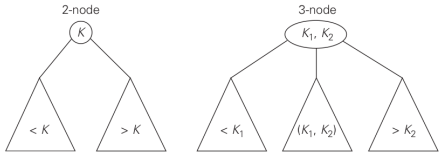


FIGURE 6.7 Two kinds of nodes of a 2-3 tree.