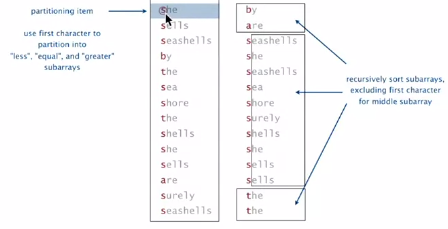
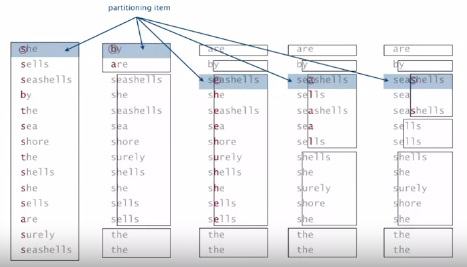
3-way Radix Quicksort

Do 3-way partitioning of the dth character.

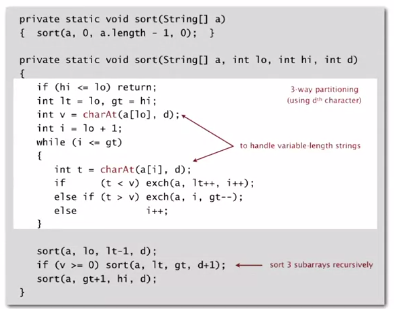
* Less overhead than R-way partitioning in MSD string sort
* Does not re-examine characters equal to the partitioning char  
  (but does re-examine characters not equal to the partitioning char)



Trace of recursive calls



3-way radix quicksort Java implementation



**Performance**

3-way string quicksort vs standard quicksort

Standard quicksort:

* Uses ~2 N ln N **string** compares on average
* Costly for keys with long common prefixes (and this is a common case)

3-way string (radix) quicksort

* Uses ~2 N ln N **character** compares on average for random strings
* Avoids re-comparing long common prefixes

3-way string quicksort vs MSD string sort

MSD String sort

* Is cache-inefficient
* Too much memory usage storing count[]
* Too much overhead reinitializing count[] and aux[]

3-way string quicksort

* Has a short inner loop
* Is cache-friendly
* Is in-place

Bottom line: 3-way quicksort is method of choice for string sorting