The String data type

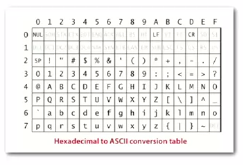
String: a sequence of characters.

Important fundamental abstraction:

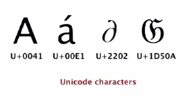
* Info processing
* Genomic sequences
* Programming systems

C char data type: typically and 8-bit integer.

* Supports 7-bit ASCII
* Can represent only 256 characters



* Most of the time now Unicode:  
  Unicode has many more characters (216  instead of 28)

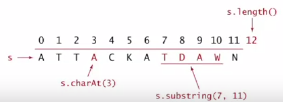


Java char type: 16-bit unsigned int

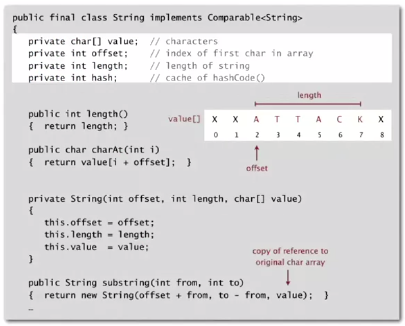
* Supports original 16-bit Unicode
* Supports 21-bit Unicode 3.0 (awkward)

Strings in Java:

String data type: sequence of characters (immutable)  
Length: number of characters  
Indexing: get the ith character  
Substring extraction: get a contiguous subsequence of characters  
String concatenation: append one character to end of another string

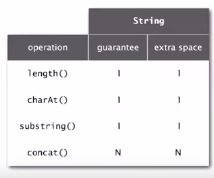


Java String implementation



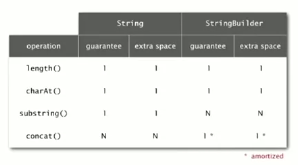
Performance of strings

Underlying implementation: immutable char[] array, offset and length

**

Memory: 40 + 2N bytes for a virgin string of length N.   
*Can use byte[] or char[] instead of String to save space, but lose convenience of String data type*

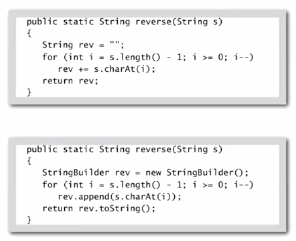
Another option for Strings in Java with different performance guarantees:



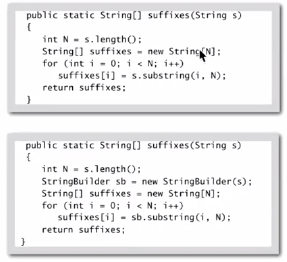
This is better when building a string from StandardInput. Constant time for concatenation (amortized), but linear time for substring method.

StringBuffer data type is similar, but thread safe and slower.

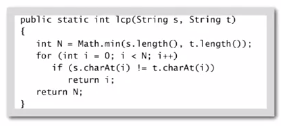
Reversing a String using different String ADTs:



Suffix array using different String ADTs:



Longest common prefix



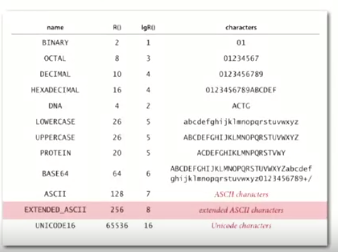
Linear time in worst case, but sublinear time in typical cases

Running time is proportional to length D of longest common prefix.  
*Also can compute compareTo() in sublinear time*

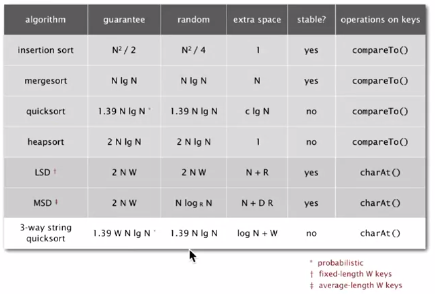
Alphabets

Digital key: sequence of digits over fixed alphabet

Radix: number of digits R in alphabet



Summary of performance of sorting algorithms



String sorting summary

We can develop linear time sorts

* Key compares not necessary for string keys
* Use characters as index in an array

We can develop sublinear time sorts

* Input size is amount of data in keys (not number of keys)
* Not all of the data has to be examined

3-way string quicksort is asymptotically optimal

* 1.39 N lg N chars for random data

Long strings are rarely random in practice

* Goal is often to learn the structure!
* May need specialized algorithms