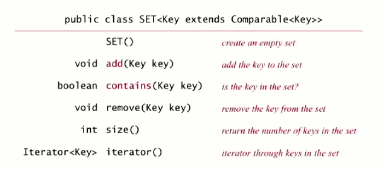
ST applications

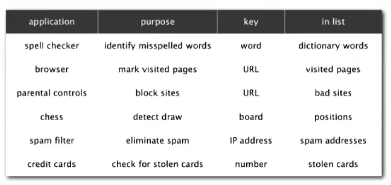
SETS

A mathematical set is just a collection of distinct keys

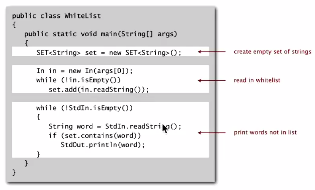


How to implement? Simply remove “value” from any ST implementation

You can use sets with exception filters to filter for data (only include) or filter out data (exclude all)



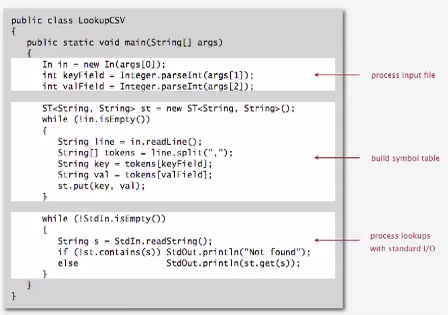
Implementation   
(only dif between white and blacklist is (set.contains(word)) becomes (!set.contains(word)):



DICTIONARY CLIENTS

Read in a file and store data organized by one category of info (e.g. store website with key as IP and value as address or vis versa), then search by selected key for the associated value(s)

Sample implementation:

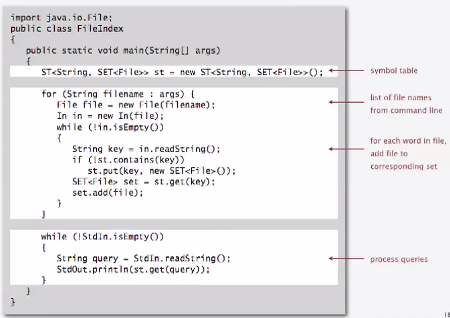


INDEXING

The goal: Given a list of files specified, create an index so that you can efficiently find all files containing a given query string.

Solution: key = query string, value = set of files containing that string

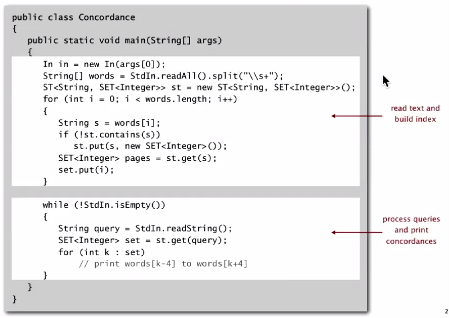
Sample implementation:



CONCORDANCE

The goal: Preprocess a text corpus to support concordance queries; given a word, find all occurrences with their immediate contexts

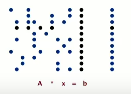
Sample implementation:



SPARSE VECTORS

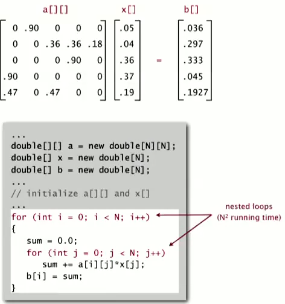
Problem: Sparse matrix-vector multiplication

Assumptions: matrix dimension is 10,000; average nonzeros per row ~10





Problem and standard implementation:



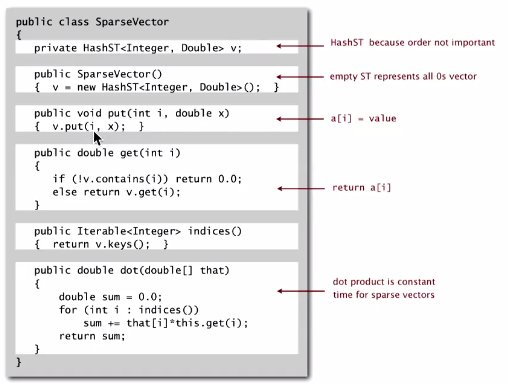
Vector representations

* 1D array  
  C:\Users\Zach\AppData\Local\Microsoft\Windows\INetCache\Content.Word\1darray.png
  + Constant time access
  + Space proportional to N
* Symbol table

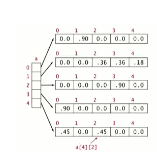


* + Key = index, value = entry
  + Efficient iterator
  + Space proportional to number of nonzeros

Sample implementation

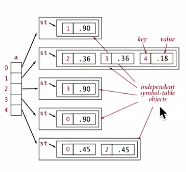


Matrix representations

2D array (standard) matrix rep: each row of matrix is an array  


* Constant time access to elements
* Space proportional to N2 (quadratic)

Sparse matrix rep: Each row of matrix is a sparse vector



* Efficient access to elements
* Space proportional to number of nonzero elements (plus N for ST overhead)

Due to space, much more efficient matrix multiplication method

Sparse matrix times a vector == running time is proportional to nonzero elements in each row (linear N)