The UNIVERSITY of VERMONT

Benjamin Fenster
CS-104 (Lee)
10 October 2003

Linked Lists, Stacks, Queues

I. Multilists
   a. Multiple lists in one data structure
   b. Useful for matrices that are sparsely populated.
   c. If the matrix is completely full, just use a 2D array! If the matrix is sparsely populated that would waste space.
   d. Each node in multilist has two pointers: next row, next column (see slides)
   e. 2D Array vs. Multilist
      i. M = number of rows
      ii. N = number of columns
      iii. K = number of non-null elements in the list
      iv. Storage
         1. 2D Array = $\theta(MN)$
         2. Multilist = $\theta(M + N + K)$ *One cell for each row, column, and full cell.*
      v. Access Time
         1. 2D Array = $\theta(1) = O(1)$ *Direct access*
         2. Multilist = $O(M + N)$
      vi. Notice the tradeoff between time and space!

II. Generalized Queue Hierarchy
   a. ADT, Generalized Queue
      i. Encapsulates collection of items
      ii. Operations: insert, remove, size, isEmpty, many others
   b. Subtypes
      i. LIFO Queue (Stack)
      ii. FIFO Queue (Queue)
      iii. Random Queue (insert/remove wherever we want)
      iv. Deque (insert/remove at either end)
      v. Priority Queue (based on priority, not position in list). Very common. This is used repeatedly in CS-104
   c. Stack Example: Postfix Notation
      i. Example of Postfix: 1 3 + 16 12 - * 12 3 2 * / +
      ii. Infix: (1 + 2) * (4 – 2)
      iii. Prefix: * + 1 2 – 4 2
      iv. Postfix: 1 2 + 4 2 - *